UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

RECUPERACIÓN DE IMÁGENES BASADA EN DIBUJOS MEDIANTE REDES CONVOLUCIONALES

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELÉCTRICO

ANÍBAL IGNACIO FUENTES JARA

PROFESOR GUÍA:
JOSÉ SAAVEDRA RONDO

MIEMBROS DE LA COMISIÓN:
JORGE SILVA SÁNCHEZ
JUAN BARRIOS NÚÑEZ

SANTIAGO DE CHILE
2020

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL ELÉCTRICO
POR: ANÍBAL IGNACIO FUENTES JARA
FECHA: MARZO 2020
PROF. GUÍA: JOSÉ SAAVEDRA RONDO

RECUPERACIÓN DE IMÁGENES BASADA EN DIBUJOS MEDIANTE REDES CONVOLUCIONALES

La recuperación de imágenes basada en dibujos (en inglés *Sketch Based Image Retrieval* o SBIR) ha aumentado el interés de empresas e investigadores en los últimos años, debido a que representa una modalidad de búsqueda sencilla y a la vez poderosa, y a la proliferación de dispositivos móviles con pantalla táctil, que hacen que este tipo de consultas sea fácil de realizar. El problema de SBIR consiste en realizar consultas a través de dibujos a un catálogo o dataset de imágenes, de modo de ordenar estas imágenes acorde a su similitud con el dibujo realizado. En este trabajo se aborda el problema mediante redes convolucionales, en donde una red neuronal convolucional es entrenada para extraer características de dibujos e imágenes, las cuales luego son utilizadas en conjunto a una métrica de similitud para cuantificar la similitud entre entre la consulta y las imágenes del catálogo. Además, en este trabajo se extiende el alcance de SBIR para agregar color a los dibujos, y así recuperar imágenes de forma y color similar al dibujo realizado, este problema es llamado *Color Sketch Based Image Retrieval* (CSBIR); se propone así una nueva arquitectura basada en redes convolucionales para entrenar este tipo de sistemas.

Para abordar el problema de SBIR se implementan tres enfoques diferentes basados en la literatura, los cuales son **Deep SBIR**, **Siamese SBIR**, **Multi Stage Regression SBIR**. Estos enfoques utilizan distintas funciones de costo durante el entrenamiento, tal como *cross entropy loss*, *siamese loss* y *triplet loss*; en particular el último método es el que presenta mejores resultados, obteniendo un mAP de 0.553 en el dataset de evaluación *Flickr 15K*, resultado que alcanza el estado del arte.

Por otro lado para abordar el problema de CSBIR se proponen dos enfoques distintos, el primero llamado **CSBIR con histogramas de color** consistente en extraer por separado características de forma y de color de dibujos e imágenes, mientras que el segundo enfoque llamado **SBIR con *Quadruplet Networks*** utiliza una arquitectura con *Quadruplet Networks* y una nueva función de pérdida para extraer una representación que considere tanto forma como color. Para obtener un dataset de entrenamiento se propone una metodología para generar dibujos con color a partir de imágenes. Para la evaluación de estos métodos se realizan a mano 200 dibujos con color a partir de un dataset de fotografías de catálogo; los resultados muestran que el segundo método alcanza un desempeño superior, llegando a un MRR de 0.352, comparado con 0.216 del primer método.

Finalmente se concluye que los métodos desarrollados presentan un muy buen desempeño, logrando replicar el estado del arte en SBIR; esto debido a las metodologías empleadas en el entrenamiento y a la capacidad de las redes convolucionales de aprender y generalizar mediante grandes volúmenes de datos. Además, estos modelos pueden ser utilizados para implementar buscadores en plataformas de e-commerce, representando una modalidad de búsqueda novedosa y atractiva para el usuario.

A mi familia, pareja, amigos y profesores; que me han acompañado en este viaje y me inspiran para avanzar cada día.

Agradecimientos

Quisiera agradecer en primer lugar a mis padres Francisco y Verónica; ellos son quienes me han acompañado en este camino y sin ellos no podría haber llegado acá. Gracias a mi padre por brindarme su preocupación a diario, por brindarme su jazz en las tardes, su humor de padre, sus enseñanzas y su esfuerzo. Gracias a mi madre por mostrarme la definición de ser una gran mujer, por preocuparse de que nunca me haya faltado nada, por sentarse a estudiar tantas veces conmigo en mi infancia. Gracias a ambos por su amor de padres.

Me gustaría también agradecer a mi familia; a mis hermanos Luna y Alonso por acompañarme durante toda mi infancia, por jugar conmigo y enseñarme a crecer. A mis abuelitos, porque durante años me regalonearon. A mis tías, tíos, primas y primos por darme tantos momentos felices en familia.

A quien también quisiera agradecer es a mi acompañante de vida Natalia, por brindarme estos años de profundo aprendizaje, de crecimiento, de risas, de estudio y de amor. Me has inspirado a desarrollarme como persona y a ser una mejor versión de mi mismo; eres una mujer fuerte, inteligente y admirable y he aprendido mucho de ti.

Quisiera también dar las gracias a mis amigos de Again, a Oscar, Osvaldo, John y Cristian, quienes con su música, con su simpatía, con su gran amistad me han dado muchas sonrisas y buenos momentos. Espero conservemos esta amistad que tanto nos fortalece.

También agradezco a las ratitas, Baño, Cheru, More, Tomi, Vale, Cami, Azu, Cote y Octavio; quienes me acompañaron durante la universidad, dándome muchas alegrías y momentos felices, espero nuestros caminos se mantengan unidos.

A mis compañeros de Ingeniería Eléctrica y de Inteligencia Computacional, Simón, Hans, Giovanni, Miguel, Felipe, Daniel y muchos otros, por acompañarme en el estudio, por ayudarme a aprender y por brindarme tan buenos momentos juntos.

Quisiera agradecer por último a los profesores que he tenido a lo largo de mi vida, quienes me brindaron su conocimiento, me hicieron ver que los límites del saber se expanden mucho más lejos de lo que aparentan, me instaron a aprender, a estudiar, a crecer, a tomar una mayor iniciativa para desarrollar mi saber. En particular gracias a José, Juan y Jorge por apoyarme y guiarme con este trabajo.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

1.1. Motivación

En el contexto del vertiginoso avance de la tecnología, de una computación cada vez más asequible y con más poder de cómputo, de técnicas de aprendizaje de máquinas cada vez más sofisticadas y de la gran cantidad de datos disponibles; constantemente se crean nuevas aplicaciones que utilizan estas herramientas con el fin de facilitar labores cotidianas de las personas.

En este contexto un área que se encuentra en constante desarrollo son las herramientas de búsqueda por similitud, las cuales son cada vez más utilizadas en la industria, ya que facilitan a los usuarios la búsqueda y la recomendación de productos similares; un ejemplo destacable de este tipo de aplicación es *Shazam*, que permite identificar canciones a mediante muestras audio de la misma, a través de un sistema de recuperación de audio que utiliza búsqueda por similitud; o la empresa *Impresee*, cuya aplicación es capaz de buscar productos en el retail a través de dibujos o fotos, mediante sistemas de recuperación de imágenes.

Dentro de las motivaciones de este trabajo cabe mencionar el constante avance que han tenido las redes neuronales convolucionales para campos como el procesamiento de imágenes, debido a las numerosas aplicaciones que han surgido de estas, a los constantes avances en hardware, en particular en las unidades de procesamiento gráfico (GPU), que aumentan su velocidad y capacidad de manera muy acelerada, y a los destacables resultados de estas redes en una gran variedad de problemas. Así, algunas de las aplicaciones que tienen las redes convolucionales corresponden a sistemas de clasificación y regresión, detección, búsqueda por similitud, generación de imágenes, reducción de dimensionalidad, entre otros.

Se propone así realizar un sistema de recuperación (búsqueda) de imágenes basada en dibujos, el cual se extiende a dibujos con color; mediante el uso de redes convolucionales. Se decide utilizar redes convolucionales debido a su capacidad de ser entrenadas con grandes volúmenes de datos y a sus resultados del estado del arte en una enorme cantidad de problemas.

1.2. Definición y Relevancia del Problema

En el presente trabajo de memoria se pretende realizar un sistema de recuperación de imágenes basada en dibujos. Esto es, construir un sistema que reciba como entrada un dibujo con o sin color y a través de técnicas de *deep learning*, en particular redes convolucionales, permita ordenar las imágenes de un dataset de acuerdo a su similitud con el dibujo y de este modo encontrar las más parecidas.

A diferencia de una búsqueda por texto, la búsqueda basada en dibujo o *sketch* presenta ventajas tales como:

- Un *sketch* permite expresar en forma mucho más precisa y eficiente una búsqueda que un texto, debido a que con el sketch se pueden representar forma, pose, estilo y color de una fotografía.
- Las palabras no siempre son la forma más fácil de describir exactamente lo que una persona quiere buscar, sobre todo cuando hay detalles muy específicos.
- El creciente uso de pantallas táctiles y de dispositivos electrónicos está cambiando la forma en que la gente busca los productos, y facilita este tipo de búsquedas.

Existen sin embargo diversas dificultades asociadas a la implementación de este tipo de sistemas, algunas de ellas son:

- Los dibujos son representaciones abstractas cuya naturaleza es intrínsecamente diferente a las fotografías, por lo cual las características extraídas a partir de dibujos tienen distinta naturaleza respecto a las características de imágenes realistas.
- Las personas realizan dibujos sin tener una referencia real de los objetos, por lo que pueden resultar en variaciones en apariencia y estructura.
- Los dibujos tienen una gran variabilidad intra-clase, debido a que cada persona tiene distinta habilidad para dibujar, así, dos personas pueden realizar un dibujo muy distinto del mismo objeto.
- Actualmente no existen bases de datos públicas de dibujos con color.

Las potenciales aplicaciones de este tipo de sistemas están orientadas al retail y al comercio electrónico (*e-commerce*), en donde se pueden buscar los productos ofrecidos por una tienda a través de dibujos de estos mismos; eso facilita a los usuarios la búsqueda de productos, sobre todo en tiendas grandes con una amplia cantidad de productos diferentes, o para clientes que desconocen el nombre de productos, pero que tienen una imagen mental de los mismos. Además, las búsquedas a través de dibujos resultan atractivas y didácticas para el usuario, aumentando el interés del usuario en tiendas donde se haya implementado este tipo de sistemas.

1.3. Objetivo General

El objetivo general de este trabajo es desarrollar y comparar distintos modelos de recuperación de imágenes basados en redes convolucionales, que utilicen como consultas dibujos sin y con color. Como referencia se presenta en las figuras 1.1 y 1.2 un ejemplo de recuperación

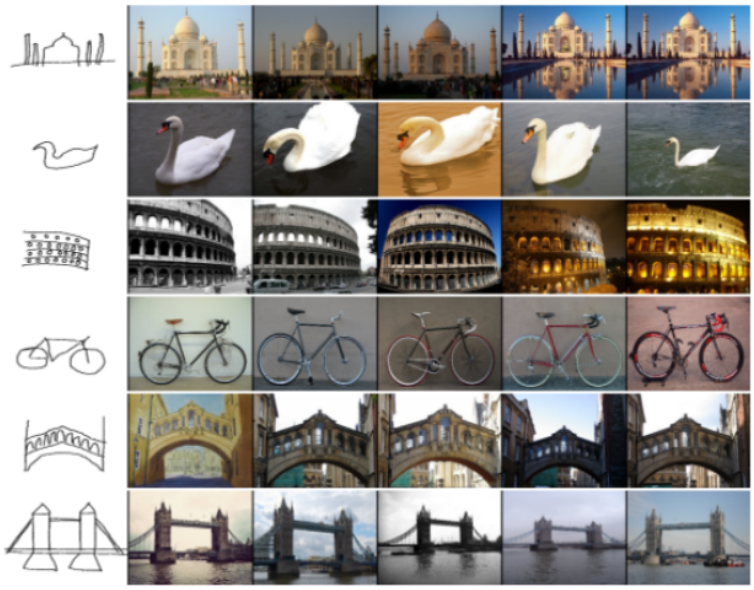

Figura 1.1: Ejemplo de recuperación de imágenes basada en dibujos sin color

de imágenes basadas en dibujos sin color y con color respectivamente, en donde se aprecia la modalidad de entrada y los resultados esperados.

1.4. Objetivos Específicos

El presente trabajo de memoria consta de los siguientes objetivos específicos.

1. Definir una arquitectura de red convolucional, esta será el núcleo (*backbone*) de los métodos a utilizar. Esta arquitectura de red convolucional estará encargada de la extracción de características de las imágenes y de los sketch, de forma de generar una representación que contenga la información semántica de las entradas.
2. Explorar distintas metodologías para la recuperación de imágenes basada en dibujos sin color, comparar sus desempeños y evaluar la mejor opción, tanto de modo visual como mediante el uso de métricas como *mean average precision*.
3. Construir un dataset para entrenamiento del sistema de recuperación basado en dibujos con color, este dataset se debe componer de fotografías de objetos y productos; en conjunto a dibujos de los mismos. Por otro lado se debe construir un dataset de evaluación, consistente en fotografías y dibujos con color realizados de forma manual por personas, basados en las fotografías.
4. Implementar un baseline para CSBIR, el cual será el punto de comparación del método propuesto. Este baseline consistirá en la extracción de características de forma y de color de manera separada.
5. Definir la metodología para el entrenamiento de la red de recuperación de imágenes basada en dibujos con color, de modo que la red sea capaz de distinguir características de forma y color de dibujos y fotografías.
6. Entrenar y evaluar el sistema de CSBIR, mediante métricas como mAP (*mean average precision*), *mean reciprocal rank* y *recall ratio*.

Figura 1.2: Ejemplo de recuperación de imágenes basada en dibujos con color

1.5. Hipótesis

Para la resolución del problema se consideran distintas hipótesis.

En primer lugar, los dibujos o sketches nos entregan una representación abstracta de los objetos, que sin embargo puede asociarse a una imagen real, esta hipótesis se postula debido a que un humano es capaz de realizar el proceso de asociar un dibujo a una fotografía de acuerdo a la similitud entre ambas. Esto nos lleva a la hipótesis de que el problema puede ser abordado de manera eficaz mediante redes convolucionales, ya que este tipo de redes basan su funcionamiento en la percepción visual de los seres vivos.

Una segunda hipótesis a comprobar es que la forma y el color son características que se pueden analizar por separado, así, uno puede utilizar redes convolucionales para extraer características de forma y un vector de características como histogramas de color para extraer estas características de color, y luego ponderar ambas características cambiando el peso relativo entre las características de forma y las de color.

Finalmente se postula como hipótesis que las redes convolucionales siamesas (o sus derivadas) son una forma adecuada de afrontar el problema, debido a que este tipo de redes tiene como objetivo encontrar una representación de las entradas tal que se minimice la distancia entre pares positivos (o pares similares de imágenes) y se maximice la distancia entre pares negativos (o pares diferentes); además este tipo de redes tiene la ventaja de que pueden encontrar una representación que haga comparables imágenes reales con dibujos. Se plantea que un sistema que considere de forma conjunta forma y color funcionará mejor que un sistema que lo haga por separado.

1.6. Alcances

Dentro de los alcances del proyecto se encuentran, en primer lugar, la implementación de distintos métodos de recuperación de imágenes basada en dibujos sin color, esto con el fin de comparar distintas metodologías de entrenamiento, funciones de costo y arquitecturas de redes, y evaluar cual de ellas es la mejor.

En segundo lugar se menciona la construcción de un dataset de imágenes, y dibujos con color basados en estas imágenes; para el dataset de dibujos con color se considera que los dibujos con color están conformados por bordes y por relleno (color), tanto bordes como color se almacenan de manera separada para facilitar distintas tareas.

Para comparar el sistema con color desarrollado se realizará primero un baseline para la recuperación de imágenes basada en sketch con color, este se basará en extracción de características de color y de forma por separado; así, las características de color serán extraídas mediante métodos clásicos de procesamiento de imágenes, mientras que la extracción de forma se realizará mediante los resultados de la primera sección de la memoria. También se consideran dentro de los alcances la construcción del sistema de CSBIR basado en redes convolucionales, para esto se definirá la arquitectura de la red convolucional y la forma de entrenamiento más adecuada para el problema, así, se debe definir una metodología de entrenamiento acorde al problema.

Finalmente se compararán los distintos sistemas desarrollados en un conjunto de test, utilizando métricas como mean average precision (mAP), precisión, recall ratio y mean reciprocal rank.

No se considera como alcance de la memoria la optimización del sistema de recuperación (por ejemplo mediante técnicas de búsqueda aproximada o reducción de dimensionalidad) o la implementación del frontend del sistema (interfaz de usuario).

1.7. Estructura de la memoria

El trabajo de memoria se estructura de la siguiente manera; se comienza con la presente introducción, indicando el tema a abordar, su motivación, relevancia, objetivos generales y específicos, hipótesis planteada y alcances del trabajo; a continuación se presenta el marco teórico, en el cual se explican los distintos conceptos necesarios para abordar el problema, tales como la definición del problema de recuperación de imágenes, redes convolucionales, redes siamesas, detección de bordes, espacios de color y métricas de evaluación; además se describen los principales métodos del estado del arte utilizados para resolver este problemas o problemas similares. Posteriormente se presenta la metodología utilizada para abordar el problema en este trabajo, explicando en detalle la forma en que se resuelve el problema, tanto conceptualmente como en términos de implementación. Luego se presentan los resultados de los distintos métodos propuestos; haciendo una comparación cuantitativa de estos resultados. Se continúa con la discusión y análisis de estos resultados explicando las ventajas y desventajas de los distintos métodos. Finalmente se termina con las conclusiones obtenidas y la bibliografía correspondiente.

Capítulo 2

Marco teórico y estado del arte

2.1. Marco Teórico

2.1.1. Recuperación de Imágenes

La recuperación de la información está encargada de la búsqueda de información en cualquier tipo de recurso digital tal como texto, audio, imágenes, videos, entre otros. El ejemplo más claro de sistemas de recuperación de la información corresponde a motores de búsqueda web como Google, en los cuales se les ingresa comúnmente texto y el motor busca en la web la información (como textos, imágenes, noticias) relacionadas a la búsqueda realizada.

En el contexto de recuperación de la información se enmarca lo que se conoce como recuperación de imágenes; un sistema de recuperación de imágenes corresponde entonces a un sistema que recibe como entrada una consulta (*query*) y a través del proceso de búsqueda y recuperación en un catálogo de imágenes nos regresa las más relacionadas con nuestra *query*. Pueden existir distintos tipos de consulta en un sistema de recuperación de imágenes, siendo la más usual la búsqueda por texto (*text based image retrieval*). En este problema se ingresa una consulta escrita y se deben retornar las imágenes que cumplen con la descripción entregada a la entrada; para ello los métodos tradicionales utilizan metadata, esto es, en las imágenes del dataset se añade información descriptiva de la imagen, como etiquetas de algún tipo, títulos, descripciones o palabras claves; el sistema entonces actúa buscando las coincidencias de la consulta con la metadata.

Otro tipo de consultas puede ser a través de imágenes, con lo que el problema se conoce como *content based image retrieval*. En este caso la forma clásica de abordar el problema es extrayendo características de la *query* y características de las imágenes del dataset, estas características pueden corresponder a características de orientación, como *histogram of oriented gradients* (HOG) o *scale invariant feature transformation* (SIFT), características de textura como Local Binnary Patterns (LBP), descriptores de color, de forma, o lo más usado recientemente, características extraídas a partir de redes profundas (*deep features*). Una vez extraídas estas características tanto de la imagen de entrada como de las imágenes del dataset, se realiza el proceso de ordenamiento de las imágenes; en este proceso se compara la similitud entre la imagen de entrada con las imágenes del dataset, comparando la distancia

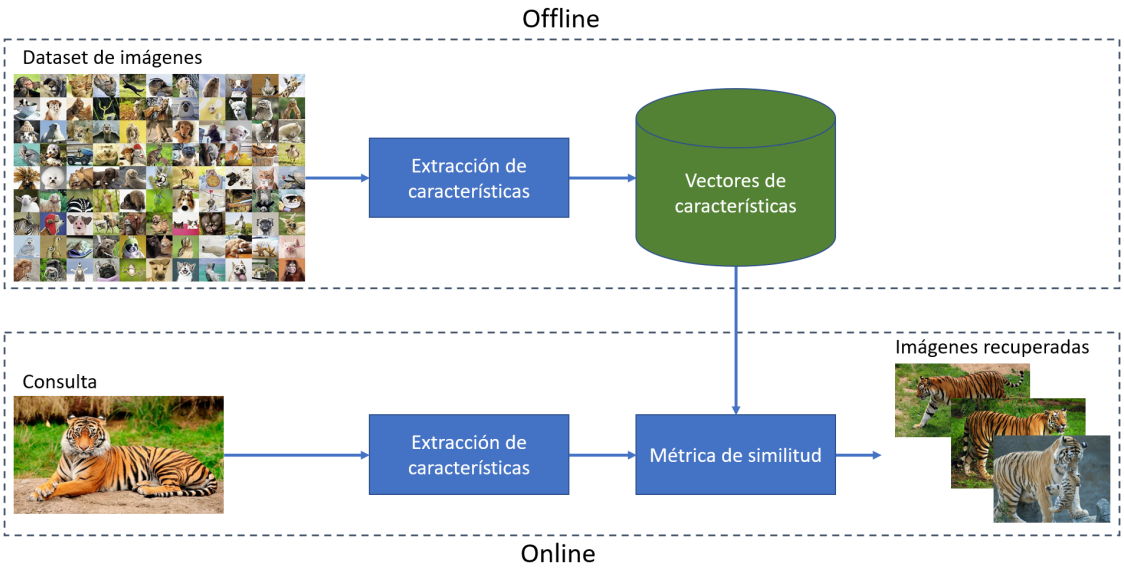

Figura 2.1: Ejemplo de recuperación de imágenes basada en contenido.

entre los vectores de características, así, las imágenes con mayor similitud serán las que presenten una menor distancia de sus vectores características con las de la query. Se destaca que el proceso de extracción de características del dataset se realiza de manera offline, mientras que la extracción de características de la consulta y la comparación con las características del dataset mediante la métrica de similitud se realiza de manera online. En la figura 2.1 se representa gráficamente este proceso.

Finalmente, si el problema de recuperación de imágenes recibe como consulta un dibujo el problema es conocido como *sketch based image retrieval* (SBIR). Este problema ha sido abordado de distintas maneras en la literatura, tanto con descriptores de bajo y medio nivel, como con redes convolucionales. Este problema presenta la dificultad de que la modalidad de búsqueda (dibujo) no corresponde a una imagen del mismo dominio que las imágenes del dataset (generalmente fotografías), es por esto que al aplicar una extracción de características en los dibujos y en las fotografías estas características tenderán a ser muy distintas. Para abordar este problema las técnicas más comunes dan un paso previo respecto al problema de *content based image retrieval*, este corresponde a la conversión de las imágenes del dataset a representaciones más similares a los dibujos, aplicando una técnica de extracción de bordes en las imágenes (como *Canny* [4] o *Sketch Tokens* [16]); y luego se procede de manera usual, extrayendo características y utilizando una métrica de similitud sobre las características extraídas; sin embargo este enfoque tiene el problema de que las imágenes pierden información de color y textura al quedarnos solo con los bordes. Se presenta en la figura 2.2 un ejemplo de este proceso.

2.1.2. Redes convolucionales, siamesas y triplets

Redes convolucionales

Las redes neuronales convolucionales son un tipo de red ampliamente utilizadas en procesamiento de imágenes debido a que son capaces de extraer características semánticas a

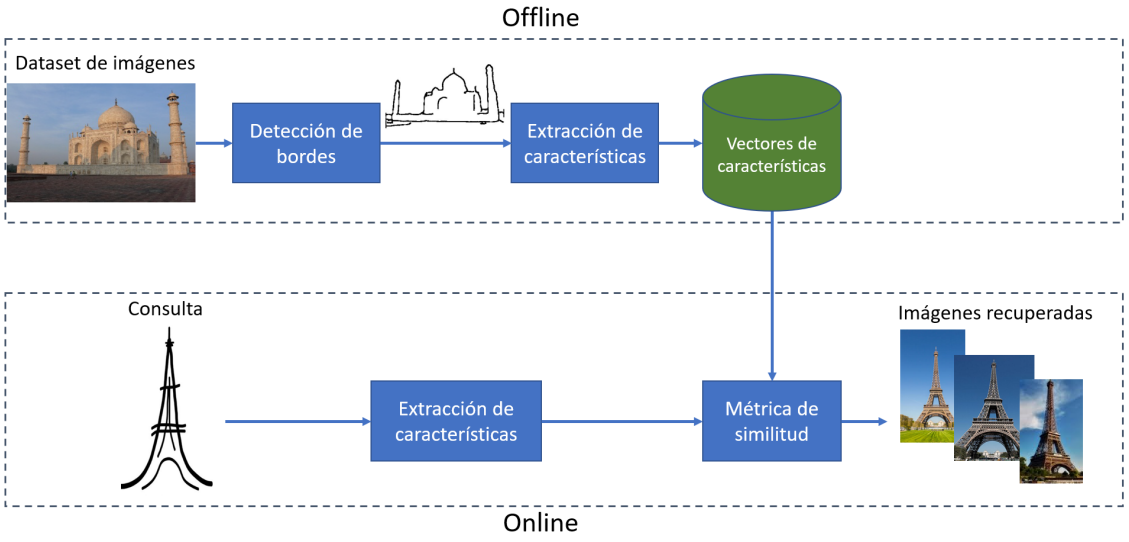

Figura 2.2: Ejemplo de recuperación de imágenes basado en dibujos.

partir de diversas capas de convolución o filtrado con parámetros entrenables, lo que permite implementar sistemas que requieran un procesamiento mínimo de las imágenes y de esta manera evitar la ingeniería de características, que es en la mayoría de los problemas el factor determinante a la hora de obtener buenos resultados. Estas redes son utilizadas en problemas de clasificación [14] [10], detección [21] [22], segmentación [9] [2], búsqueda por similitud [18] [30], entre otros. A continuación se realiza un repaso sobre las arquitecturas de distintas redes convolucionales [13].

Existen diversas arquitecturas de redes convolucionales; dentro de las más famosas se encuentran LeNet-5 (1998) [15], uno de los primeros trabajos en utilizar redes convolucionales para clasificación; luego **AlexNet** (2012) [14] atrajo nuevamente el interés hacia las redes convolucionales al ganar la competencia ImageNet, usando una red con 8 capas y 60 millones de parámetros, y cuya principal novedad fue el uso de nuevas funciones de activación llamadas *Rectified Linear Units*. Posteriormente, el año 2014 la red **VGG-16** [31] llamó la atención al mejorar los resultados de AlexNet introduciendo una red con una cantidad de capas mucho mayor (16 capas), sin embargo esta red presenta una cantidad de parámetros muy grande, con 138 millones de parámetros, lo cual la hace muy ineficiente para entrenar y evaluar. Se destaca que las tres arquitecturas mencionadas anteriormente siguen el esquema de capas convolucionales seguidas por capas de *max pooling*, mientras que utilizan capas fully connected a la salida para realizar clasificación. Se presenta en la figura 2.3 una visualización de la red AlexNet.

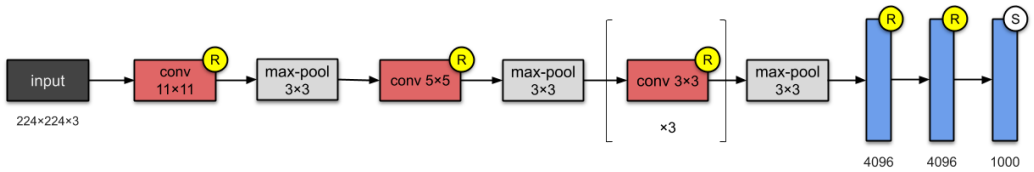

Figura 2.3: Arquitectura de la red *AlexNet*.

Posteriormente en el año 2015 se dio un salto en el desempeño de las redes convolucionales mediante los *bloques residuales*, con las redes **ResNet** [10], las cuales usan *skip connections* o atajos en la red, con el fin de hacer redes más profundas que no comprometieran la capacidad de generalización de la red (ya que estos atajos nos evitan el problema del *vanishing gradient* y además actúan como un ensamble de clasificadores), y además utilizaron *batch normalisation* para evitar el sobreajuste. Esta red cuenta con 26 millones de parámetros. En la figura 2.4 se muestra gráficamente la arquitectura de la red resnet 50, donde los *skip connections* se encuentran en los bloques *conv block* e *identity block*.

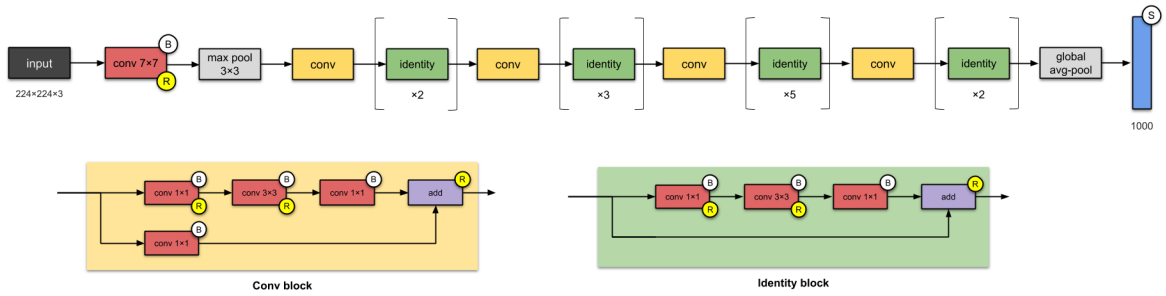

Figura 2.4: Arquitectura de la red *ResNet 50*.

Se tienen por otro lado las redes **inception**, la cual ha ido evolucionando desde inception v1 [33] el año 2014 a inception v4 [32] el año 2016. Estas redes se caracterizan por presentar bloques correspondientes a convoluciones en paralelo, con distintos filtros, las cuales se concatenan a la salida.

Finalmente, el año 2017 se plantea una mejora a la red ResNet, dando origen a la arquitectura **ResNext** [36]. La mejora planteada corresponde al uso de ramas en paralelo en cada módulo de convolución, idea también llamada *grouped convolution*. Se ha estudiado que esta idea produce grupos de filtros que se especializan en ciertas características de las imágenes como distintos colores, lineas blancas y negras, entre otros. Esta arquitectura presenta 25 millones de parámetros. Se presenta en la figura 2.5 la arquitectura ResNext 50, donde las *grouped convolutions* corresponden a los bloques *conv block* e *identity block*.

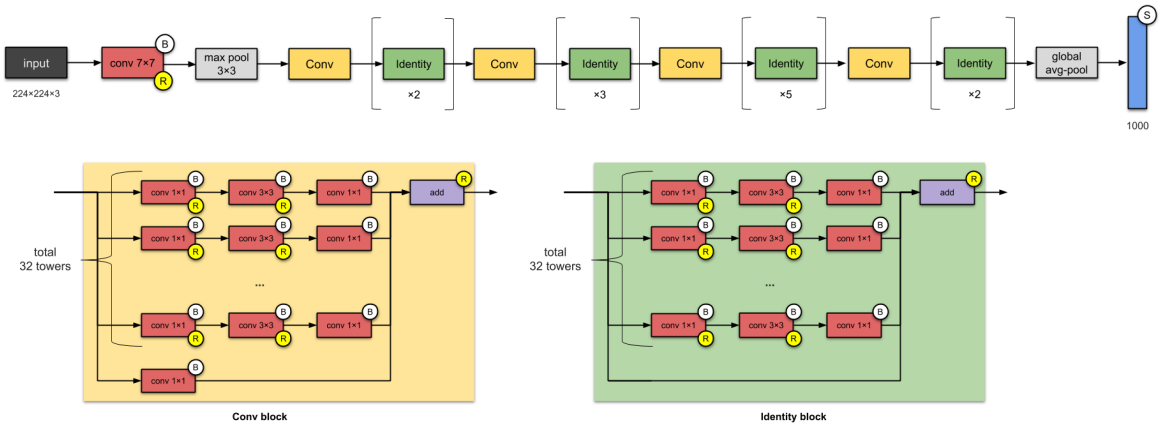

Figura 2.5: Arquitectura de la red *ResNext 50*.

Se puede recalcar además que existen diversas mejoras para aumentar el desempeño de las redes convolucionales, como *batch normalisation*, *dropout*, *skip connections*, entre otros. Una de estas mejoras corresponden a los bloques ***Squeeze and Excitation*** [11]. Estos introducen un cambio en las redes convolucionales que realizan el siguiente efecto: añaden parámetros a las capas convolucionales de modo que la red pueda ajustar de manera adaptativa el peso de cada *feature maps*. Se presenta en la figura 2.6 un bloque de *squeeze and excitation*, donde la entrada X corresponde a un feature map de dimensiones $[H \times W \times C]$, el cual es pasado por un global average pooling, obteniéndo un vector del mismo tamaño que el número de canales, este vector es pasado por capas fully connected, y finalmente multiplicado por el feature map original, obteniendo la salida *Scale*, correspondiente a la entrada ponderada de manera adaptiva en cada canal.

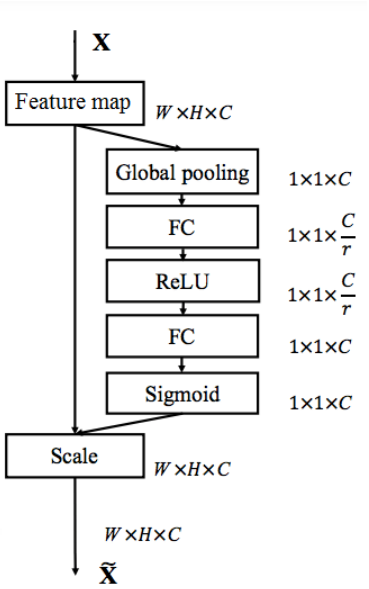

Figura 2.6: Bloque de *Squeeze and excitation*.

Redes siamesas

Las redes siamesas son un tipo de red neuronal que contienen dos o más redes idénticas, en el sentido que poseen la misma arquitectura y por lo general los mismos pesos, actualizando estos de manera simultánea en ambas subredes.

Las redes siamesas son ampliamente utilizadas en problemas donde se quiere encontrar similitud o una relación entre dos cosas comparables; por ejemplo, son utilizadas en el problema de reconocimiento de rostros; en este caso, se tiene un rostro de referencia el cual es procesado por una subred, y se tiene un rostro que se quiere verificar, procesado por la otra subred; las redes procesan ambos rostros encontrando una representación para ellos (*embedding*), luego, se compara la distancia entre ambas representaciones, verificando el rostro como correcto si la distancia es menor a cierto umbral o no verificándolo en caso contrario.

Se muestra en la figura 2.7 un ejemplo de red siamesa. El ejemplo de la figura muestra una arquitectura red convolucional siamesa, en que en la parte superior se muestra una subred y en la parte inferior se muestra otra subred. Las salidas de ambas subredes son comparadas para encontrar una métrica de similitud.

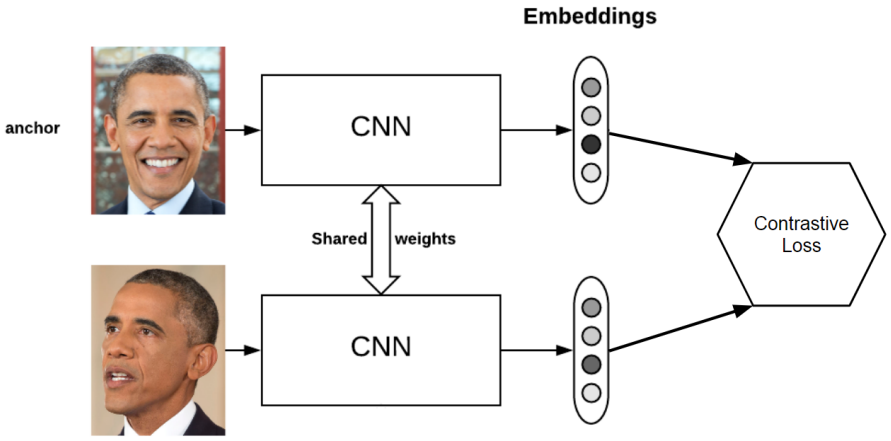

Figura 2.7: Ejemplo de red siamesa.

Cabe destacar que si ambas entradas a comparar son del mismo tipo (por ejemplo se trabaja solo con fotografías como en el problema de content based image retrieval), entonces ambas subredes comparten los mismos pesos (o en otras palabras es la misma red que se utiliza dos veces), sin embargo puede existir casos en que las entradas a comparar sean de tipos distintos, como en el caso de recuperación de imágenes a partir de dibujos, donde una entrada corresponde a fotografías y la otra a dibujos; si este es el caso entonces ambas subredes pueden tener distintos pesos.

Para entrenar una red siamesa el entrenamiento se hace con pares de imágenes. Denotemos cada par como (x_i^1, x_i^2), los cuales producen representaciones $(f_w(x_i^1), f_w(x_i^2))$. Se tiene que cada par tiene asociada una etiqueta binaria $y \in \{0, 1\}$, donde la etiqueta corresponde a $y = 1$ si los pares son pares positivos (ejemplos que están correctamente asociados o que son similares), mientras que la etiqueta es $y = 0$ si los pares son pares negativos (ejemplos

diferentes). El modelo entonces tiene como objetivo encontrar representaciones que minimicen la distancia entre ejemplos similares y la maximice entre ejemplos diferentes. Para esto se utiliza la función de pérdida contrastiva, definida como:

$$L_{\mathrm{i}} = y_{\mathrm{i}} D_{w,\mathrm{i}}^2 + (1 - y_{\mathrm{i}})\{max(0, \lambda - D_{w,\mathrm{i}})\}^2 \tag{2.1}$$

Con:

$$D_{w,\mathrm{i}} = L_2(f_w(x_{\mathrm{i}}^1), f_w(x_{\mathrm{i}}^2)) \tag{2.2}$$

Donde $f_w(x_{\mathrm{i}})$ es la representación (embedding) producido por la red, $D_{w,\mathrm{i}}$ es la medida de distancia entre ambas representaciones, por lo general igual a la norma L_2, y λ corresponde al margen (distancia mínima permitida entre las representaciones para pares negativos).

Interpretando en detalle la función de pérdida, se tiene que cuando dos muestras son similares entonces $y_{\mathrm{i}} = 1$, por lo que la función de pérdida se reduce a:

$$L_{\mathrm{i}}|_{y=1} = D_{w,\mathrm{i}}^2 \tag{2.3}$$

Por lo que se está minimizando la distancia entre las representaciones de ambas muestras. Por otro lado, si ambas muestras son diferentes, entonces $y_{\mathrm{i}} = 0$, por lo que la función de pérdida se reduce a:

$$L_{\mathrm{i}}|_{y=0} = \{max(0, \lambda - D_{w,\mathrm{i}})\}^2 \tag{2.4}$$

Por lo que se maximiza la distancia entre las representaciones si estas tienen una distancia menor a λ.

Triplet networks

Las *triplet networks* son una extensión de las redes siamesas, en la cual en vez de utilizar un par de imágenes se utiliza un trío de imágenes a la entrada de la red, donde la primera imagen corresponde a una imagen de referencia (*anchor*), la segunda imagen corresponde a un par positivo o imagen similar y la tercera imagen corresponde a un par negativo o imagen diferente. Se puede ver un ejemplo de esto en la figura 2.8.

Formalmente, se tiene un anchor, un par positivo y un par negativo, denotados como x_{i}^a, x_{i}^+, x_{i}^-, los cuales producen *embeddings* $f_w(x_{\mathrm{i}}^a)$, $f_w(x_{\mathrm{i}}^+)$, $f_w(x_{\mathrm{i}}^-)$ respectivamente. La función de pérdida utilizada al entrenar una red siamesa corresponde a:

$$L_{\mathrm{i}} = \frac{1}{2} max\{0, D_{w,\mathrm{i}}^+ - D_{w,\mathrm{i}}^+ + \lambda\} \tag{2.5}$$

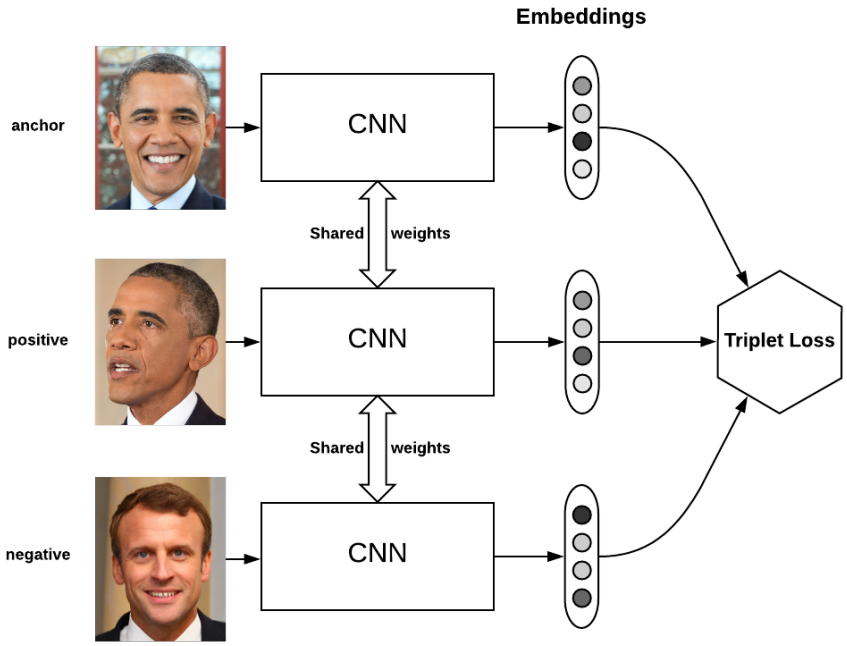

Figura 2.8: Ejemplo de *triplet network*

Con:

$$D^{+}_{w,\mathrm{i}} = L_2(f_w(x^a_\mathrm{i}), f_w(x^+_\mathrm{i})) \tag{2.6}$$

$$D^{-}_{w,\mathrm{i}} = L_2(f_w(x^a_\mathrm{i}), f_w(x^-_\mathrm{i})) \tag{2.7}$$

Esta función de pérdida nace a partir de que el objetivo de la red es cumplir la desigualdad:

$$D^{+}_{w,\mathrm{i}} + \lambda < D^{-}_{w,\mathrm{i}} \tag{2.8}$$

Es decir, que la distancia entre los anchor y sus pares negativos sea mayor por un margen λ a la distancia entre los anchor y los pares positivos.

2.1.3. Algoritmo de Canny de detección de bordes

Como se vio en la sección de recuperación de imágenes, algunos métodos de recuperación de imágenes a partir de dibujos utilizan técnicas de detección de bordes para transformar fotografías en representaciones similares a dibujos, haciendo que la comparación entre las imágenes con los dibujos de query sea más fácil.

Uno de los métodos utilizados para esto corresponde al algoritmo de Canny de detección de bordes [4]; este algoritmo está basado en los 3 criterios siguientes: buena detección, es decir, el algoritmo debe marcar el mayor número real en los bordes de la imagen como sea

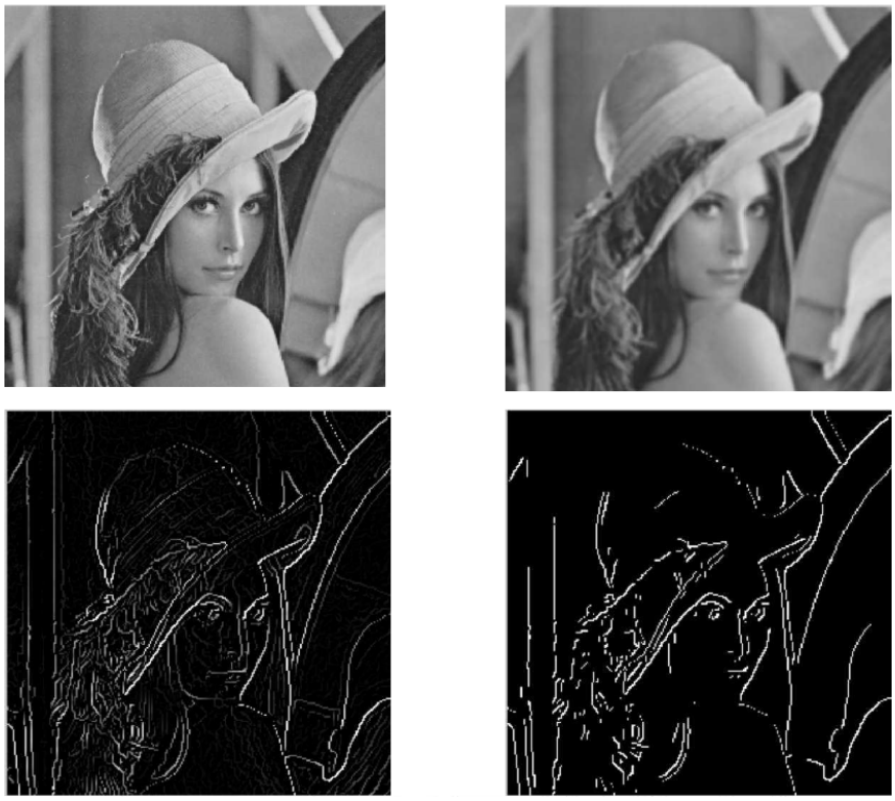

Figura 2.9: Ejemplo de detección de bordes con Canny.

posible; buena localización, los bordes deben estar lo más cerca posible del borde de la imagen real; respuesta mínima, el borde de una imagen sólo debe ser marcado una vez, y siempre que sea posible, el ruido de la imagen no debe crear falsos bordes.

Para esto, el algoritmo se compone de los siguientes pasos:

- Cálculo del gradiente: se aplica un filtro gaussiano y posteriormente se calcula la magnitud y orientación del gradiente en cada pixel de la imagen.
- Supresión no máxima: se adelgaza el ancho de los bordes encontrados con el gradiente, hasta obtener bordes del ancho de un pixel.
- Histéresis de umbral: se aplica un doble umbral para reducir la aparición de falsos bordes.

Se muestra en la figura 2.9 un ejemplo de detección de bordes con el algoritmo de Canny. En la imagen superior izquierda se presenta la fotografía original, en la imagen superior derecha se presenta la figura luego de aplicar filtro gaussiano, en la inferior izquierda está la fotografía luego de aplicar cálculo del gradiente y supresión no máxima, mientras que en la figura inferior derecha se presenta la salida de aplicar histéresis de umbral.

2.1.4. Modelos de color

Los modelos de color son modelos matemáticos utilizados para representar el color de forma numérica, es decir, un modelo de color asocia un vector numérico a un color. Estos modelos

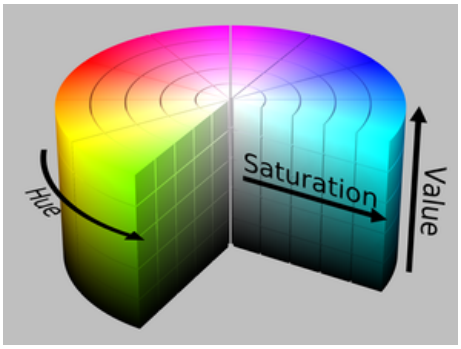

Figura 2.10: Espacio de color HSV.

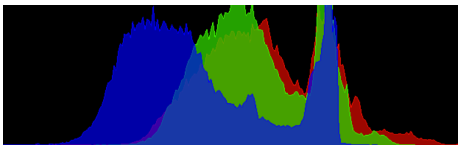

Figura 2.11: Histograma de color.

de color usualmente se representan por 3 o 4 componentes, también llamados canales. Los modelos de color más usuales utilizados corresponden a RGB y CMYK. El modelo de color RGB permite representar el color mediante tres canales, estos son el rojo, el verde y el azul; mientras que el modelo CMYK representa el color mediante cuatro canales, correspondientes a cian, magenta, amarillo y negro.

Existe también el modelo de color HSV (Hue, Saturation, Value), este corresponde a una representación alternativa del modelo de color RGB; y se acerca más a la forma en que el humano percibe los colores. Tal como se muestra en la figura 2.10, el espacio de color hsv se puede representar en un cilindro, donde en el eje central están los colores neutrales desde el blanco al negro, mientras que más hacia los extremos se encuentran los colores más saturados. Si uno se mueve alrededor del eje central (canal Hue) recorre los colores de distintos matices, mientras que el canal value representa mezclas de colores con distinta cantidad de negro.

Histogramas de color

Un descriptor clásico en el ámbito de procesamiento de imágenes corresponde a los histogramas de color. Estos son histogramas formados a partir de las intensidades de los pixeles de cada uno de los canales de color de una imagen, tal como se observa en la figura 2.11. Estos histogramas tienen la particularidad de que se pueden discretizar en una cantidad distinta de bines y se pueden concatenar, formando un vector de características que presenta una buena descripción de color de una imagen.

2.1.5. Métricas de evaluación

Mean Average Precision (mAP)

Mean average precision (mAP) [34] es una métrica de evaluación utilizada en el contexto de *object detection* y de *information retrieval*, teniendo una definición distinta para cada

(a) Query: flor

(b) Query: perro

Figura 2.12: Ejemplo de consultas e imágenes recuperadas.

contexto. En el caso de *information retrieval*, el mAP es un valor entre 0 a 1 (mayor es mejor) que nos indica en promedio si las respuestas correctas se encuentran más cercana a la primera posición, respecto al total de posibles respuestas. Supongamos se tiene un dataset con M imágenes y $|Q|$ consultas. El mAP se calcula promediando el *average precision* (AP) para todas las consultas del dataset, tal como en la ecuación 2.9.

$$mAP = \frac{1}{|Q|} \sum_{\mathrm{i}=1}^{|Q|} AP_{\mathrm{i}} \tag{2.9}$$

Se destaca que cada una de las $|Q|$ imágenes de consulta debe tener etiquetadas todas las imágenes correctas en el dataset; por ejemplo, si mi consulta es un dibujo de un perro, puedo considerar todas las fotografías de perros como correctas. Se tiene además que el AP_{i} de cada consulta i se calcula mediante la ecuación 2.10. Para su cálculo se ordenan todas las imágenes del dataset de acuerdo a su similitud con la consulta i, de mayor a menor similitud; luego se recorren todas estas respuestas ordenadas, desde $j = 1$ hasta M, sumando el término correspondiente de la sumatoria. En la ecuación, GTP_{i} corresponde al total de imágenes correctas dentro del dataset para la consulta i; $\mathbb{1}_{j=correct}$ es una función indicatriz que vale 1 cuando la respuesta j que se está mirando es correcta y 0 en caso contrario, $TP_{seen,j}$ es la cantidad de respuestas correctas vista hasta el momento j, mientras que j es el ranking de la respuesta que se está mirando actualmente.

$$AP_{\mathrm{i}} = \frac{1}{GTP_{\mathrm{i}}} \sum_{j=1}^{M} \frac{\mathbb{1}_{j=correct} \cdot TP_{seen,j}}{j} \tag{2.10}$$

Para entender mejor el AP y el mAP se plantea el siguiente ejemplo, supongamos tenemos un dataset que tiene 12 imágenes de 3 clases: 3 imágenes de flores, 4 de perros y 5 de autos; sobre este dataset se realizan dos consultas; la primera de una flor y la segunda de un perro, y se ordenan las imágenes de acuerdo a la similitud encontrada, tal como se muestra en la figura 2.12. En la figura las imágenes marcadas en verde corresponden a imágenes positivas (de la misma clase).

Tenemos para la primera consulta que $GTP_1 = 3$, luego aplicamos la fórmula de la ecuación

2.10, obteniendo:

$$AP_1 = \frac{1}{3} \cdot (0 + \frac{1}{2} + 0 + \frac{2}{4} + \frac{3}{5} + 0 + 0 + ...) = 0{,}533 \tag{2.11}$$

Mientras que para la consulta del perro se tienen 4 imágenes correctas en el dataset, así $GTP_2 = 4$; además las imágenes se recuperan todas correctamente en las primeras posiciones, se tiene entonces que el AP_2 es:

$$AP_2 = \frac{1}{4} \cdot (\frac{1}{1} + \frac{2}{2} + \frac{3}{3} + \frac{4}{4} + 0 + 0 + ...) = 1 \tag{2.12}$$

Finalmente, el mAP se calcula como el promedio del AP para todas las consultas, esto es:

$$mAP = \frac{1}{2}(0{,}533 + 1) = 0{,}767 \tag{2.13}$$

Mean Reciprocal Rank (MRR)

El *Mean Reciprocal Rank* [5] es una métrica de evaluación que toma valores entre 0 y 1 (1 es mejor) y nos indica en promedio para todas las consultas en que valor se encuentra la primera respuesta correcta. La definición matemática es:

$$MRR = \frac{1}{|Q|} \sum_{\mathrm{i}=1}^{|Q|} \frac{1}{rank_{\mathrm{i}}} \tag{2.14}$$

Donde $|Q|$ es la cantidad de consultas, y $rank_{\mathrm{i}}$ es la posición de la primera respuesta correcta para la consulta i.

Para ejemplificar el MRR consideremos las mismas consultas y respuestas de la figura 2.12, se tiene que para estas consultas el MRR se calcula como:

$$MRR = \frac{1}{2} \cdot (\frac{1}{2} + \frac{1}{1}) = 0{,}75 \tag{2.15}$$

Recall ratio

El recall ratio [26] $R(n)$ es una curva que nos indica para que porcentaje de las queries se recupera correctamente la imagen objetivo (*groundtruth*), si se mira dentro de las primeras n imágenes recuperadas. Si se tiene más de una imagen objetivo (por ejemplo todas las imágenes de la misma clase que la consulta) entonces se considera la imagen objetivo como la primera de estas imágenes en ser recuperada. Formalmente, el recall ratio se expresa en la ecuación 2.16.

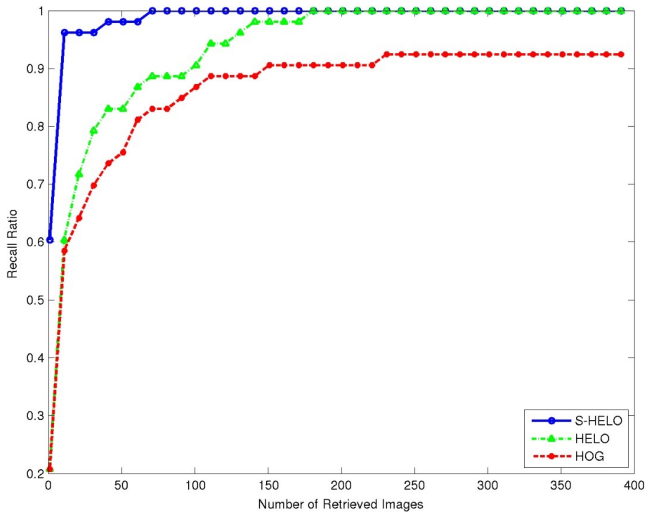

Figura 2.13: Ejemplo recall ratio [23]

$$R(n) = \frac{T(n)}{|Q|} \tag{2.16}$$

Siendo $|Q|$ la cantidad de total de queries y $T(n)$ la cantidad de queries en que la imagen objetivo se recupera dentro de las primeras n respuestas. Un ejemplo gráfico del recall ratio se presenta en la figura 2.13, donde se compara el recall ratio de 3 métodos en el contexto de SBIR. En la curva el eje x representa el número de imágenes recuperadas n, mientras que el eje y representa el recall ratio $R(n)$. Se debe recalcar que el recall ratio es una curva creciente y que siempre alcanza el valor 1.

2.2. Estado del Arte

Los sistemas de recuperación de imágenes basados en dibujos se han desarrollado desde comienzo de los años 90 y se pueden dividir en dos grandes grupos: SBIR con características clásicas o *handcrafted features* y SBIR con *deep learning*, el primero se caracteriza por el uso de descriptores tradicionales sobre los sketch, como pueden ser descriptores basados en gradientes, texturas, color o una combinación de ellos; mientras que el segundo usa características extraídas a partir de redes neuronales profundas o redes convolucionales.

2.2.1. SBIR con *handcrafted features*

En su mayoría estos métodos están basados en características de bajo nivel, y en particular en características basadas en gradiente. En esta área Eitz [7] extendió el concepto de *Bag Of Features* a SBIR, método basado en un proceso de agregación de características locales, usando una variante del descriptor HoG llamada SHoG.

La mayoría de estos métodos usa *Canny* para extraer los bordes de las imágenes y así adaptarlas a un dominio más similar al de los dibujos, sin embargo Lim et al. [16] presenta

una alternativa a este método de detección de bordes llamada *Sketch Tokens*, la cual toma en cuenta la semántica de la imagen, y a través de aprendizaje supervisado clasifica si un pixel corresponde o no corresponde a un borde.

Posteriormente, Saavedra mostró que HOG no enfrenta de manera apropiada la condición de *sparsity* en los sketches, por lo que propuso HELO [26] (*Histogram of Edge Local Orientations*), donde el histograma de orientaciones se forma por orientaciones locales que son estimadas agrupando pixeles en celdas y determinando solo una orientación representativa para cada celda. Luego HELO fue mejorada a SHELO [23] usando *square root normalization* y una estimación suave de los histogramas de orientación. También en el contexto de características de nivel medio Saavedra y Barrios propusieron los métodos de *KeyShapes* [27] y posteriormente *Learned KeyShapes* [25], donde se utiliza un proceso de *clustering* para aprender estas características de nivel medio.

2.2.2. SBIR con características profundas

Actualmente, el estado del arte en *Sketch Based Image Retrieval* es alcanzado utilizando características obtenidas a partir de redes convolucionales.

En este contexto, Saavedra et al. [28] utiliza características profundas extraídas a partir de una red convolucional pre-entrenada en el contexto de clasificación de sketches, y luego utilizó estas características para SBIR. Para extraer las características de imágenes primero debe transformar el dominio de estas imágenes calculando sus bordes, para esto utiliza *Sketch Tokens*. Para mejorar el desempeño del sistema se utiliza *Square Root Normalization*.

Qi et al. [19] utiliza una red convolucional siamesa para SBIR, donde en una rama se ingresan los sketch y en la otra rama se ingresan los bordes de las fotografías; además ambas redes comparten sus pesos. Este trabajo sin embargo se encuentra por debajo del desempeño de los métodos del estado del arte; esto se debe principalmente a que utiliza la misma red (con los mismos pesos) para adaptar diferentes dominios.

En el contexto de redes siamesas, Patsorn et al. [29] propone el dataset *sketchy*, consistente en fotografías y dibujos realizados en base a estas fotografías. Además utilizan una arquitectura de redes convolucionales siamesas para el aprendizaje de un *embedding* común entre fotografías y dibujos. Para ello utilizan una red para los dibujos y otra para las fotografías; mientras que a la salida de ambas redes se utiliza el *contrastive loss* entre ambas redes, en conjunto con el *cross-entropy loss* de cada red, con el fin de guiar el entrenamiento.

Más adelante, Xu. et al [37] propone en el ámbito de los sketch, un sistema de recuperación de sketch (tanto el dataset como la query corresponden a sketch), el cual realiza la extracción de un embedding a partir de una red convolucional, que extrae información de forma de las imágenes, y una red recurrente, que nos permite extraer información temporal de la secuencia de trazos con que se dibujó el sketch, además proponen discretizar el embedding en un vector de características binarias mediante una capa de discretización, de modo de utilizar menos espacio en memoria para las características extraídas y disminuir el tiempo de recuperación.

Finalmente, Bui et al. [3] utiliza distintas arquitecturas de redes convolucionales y utiliza un entrenamiento de varias etapas con dificultad creciente para aprendizaje entre dominios

(dibujo-fotografía), en las cuales se utilizan múltiples funciones de pérdida (*cross-entropy*, *contrastive loss* y *triplet loss*) en conjunto con redes con pesos parcialmente compartidos, logrando así adaptar el dominio dibujo-fotografía y encontrar una representación que se adapte a ambos dominios. Finalmente logran alcanzar desempeño del estado del arte en diversos datasets de evaluación.

2.2.3. Recuperación de imágenes y dibujos con color

Actualmente existen algunos trabajos de recuperación de imágenes usando características de color, sin embargo estos métodos consideran de forma separada características de color y forma.

Nugrowati et al. [17] implementa un sistema de búsqueda por similitud para *batik*, estas corresponden a imágenes simbólicas, las cuales cuentan con una diversidad de patrones influenciadas por una variedad de culturas; para abordar este problema se extraen por separado características de color, esto mediante histogramas de color 3d extraídos en el espacio RGB, y características de forma extraídas mediante *Hu Moments*.

Reddy et al. [20] propone un sistema de recuperación de imágenes basada en dibujos con color, en el cual se discretiza el espacio de color HSV para extraer características de color y *gray-level co-occurrence matrix* (GLCM) para extraer características de textura, finalmente la distancia se calcula ponderando la distancia de forma y de color. Un problema de este trabajo es que para evaluar el sistema se utilizan dibujos generados sintéticamente a partir de las imágenes y no dibujos hechos a mano.

Xia et al. [35] implementa un sistema de color SBIR de grano fino para calzado, en el cual la modalidad de las consultas son dibujos, a los cuales se les ingresa el color en los trazos. Para esto primero se extraen las características de forma mediante una *triplet network*, luego los primeros 10 resultados recuperados se ordenan de acuerdo a la distancia de color. En este trabajo sin embargo el color parece no marcar una gran diferencia, debido a que solo es utilizado para ordenar los diez mejores resultados encontrados de acuerdo a forma, por lo que finalmente el usuario estaría viendo los mismos 10 productos recuperados, solo que en distinto orden.

Capítulo 3

Metodología

3.1. Formalización del problema

El problema a resolver se formaliza de la siguiente manera.

Dado un dataset de imágenes (principalmente fotografías), y una *query* (consulta) consistente en un dibujo con o sin color; se debe encontrar y retornar las imágenes del dataset que sean similares al dibujo realizado. Para esto la metodología seguida consiste en definir una medida de similitud entre imágenes y dibujos y regresar las imágenes ordenadas de acuerdo a la similitud con el dibujo. Se puede observar de manera gráfica este proceso en la imagen 3.1, para una consulta correspondiente a un dibujo con color.

En las siguientes secciones se presenta la forma de adquisición de los datos y el sistema propuesto para la recuperación de imágenes.

3.2. Adquisición de los datos y datasets utilizados

Para implementar este tipo de sistemas se debe contar con una gran cantidad de datos que nos permitan entrenar los parámetros de una red convolucional. En la literatura existen diversos datasets de sketch based image retrieval, las cuales **no presentan dibujos con color**; sin embargo estos datasets pueden ser utilizadas para abordar el problema sin color y realizar un *baseline* para el problema con color.

En las siguientes subsecciones se describen los datasets utilizados y la metodología de procesamiento de estos datos, tanto para el problema de recuperación de imágenes basada en dibujos sin color y el problema con color.

3.2.1. Recuperación de imágenes basada en dibujos sin color

Para esto se propone utilizar datasets de imágenes y *sketch* presentes en la literatura; en la tabla 3.1 se presenta un resumen de los distintos datasets.

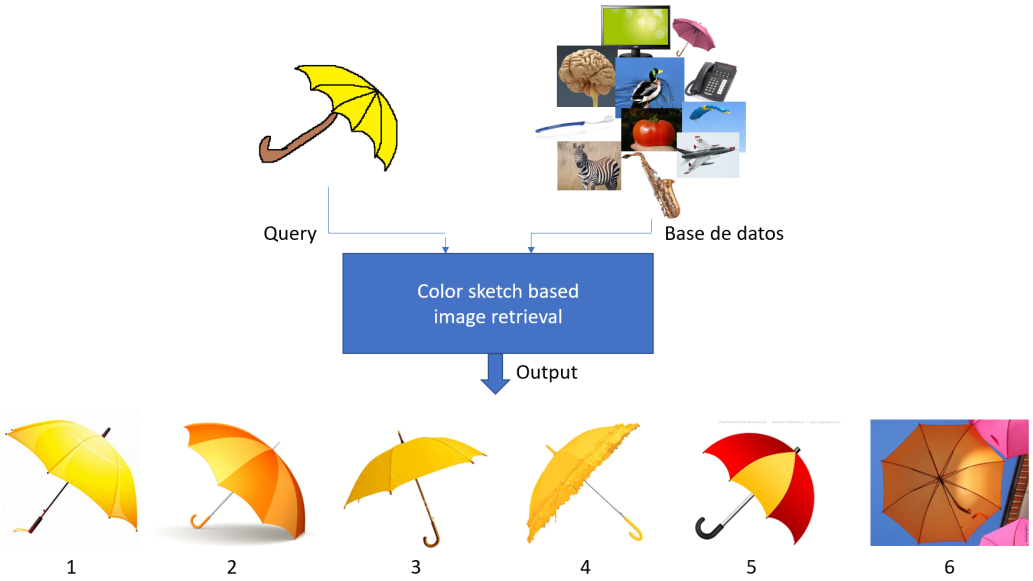

Figura 3.1: Esquema del problema a abordar.

Dataset	Número de clases	Número de imágenes	Número de dibujos	Conjunto
Flickr 25K	250	25.000	20.000	Entrenamiento
Sketchy	125	12.500	75.471	Entrenamiento
Flickr 15K	33	14.660	330	Test
Saavedra	52	1.326	53	Test

Tabla 3.1: Resumen de los datasets para SBIR.

A continuación se describen en detalle estos datasets:

Flickr25K

Este dataset propuesto por Bui et al. [3], es una extensión del dataset de sketch de Eitz [6], el cual contiene 20.000 sketch de 250 clases (con 80 dibujos por clase), y se complementa con 25.000 imágenes de las mismas clases (100 imágenes de cada clase), extraídas de Flickr, Google y Bing. Algunos ejemplos de clases son hormiga, hacha, plátano, libro, calculadora, helicóptero, entre otras. En la figura 3.2 se presentan ejemplos de imágenes y sketches de las clases anteriormente mencionadas.

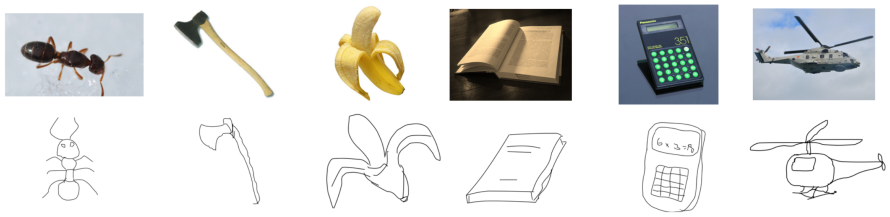

Figura 3.2: Ejemplos de imágenes y sketches del dataset Flickr25K.

Sketchy

El dataset sketchy [29] cuenta con 12.500 imágenes de 125 clases, y para cada una de estas imágenes se presentan en promedio 5 dibujos realizados en base a la imagen, dando un total de 75.471 dibujos. Este dataset está enfocado en entregar pares imagen-dibujo para abordar el problema *fine grained SBIR*. Dentro de algunas clases se encuentran alarma, dirigible, guitarra, hongo, zapato, tetera; tal como se muestra en la figura 3.3.

Figura 3.3: Ejemplos de pares de imágenes y sketches del dataset Sketchy.

Flickr15K

Flickr15k [12] consiste en 14660 imágenes, separadas en 33 clases, entre las cuales hay lugares históricos, objetos, animales, plantas. Además se cuenta con 330 consultas (dibujos sin color), en las cuales hay 10 consultas para cada clase. Algunos ejemplos de clases son aeroplano, bicicleta, big ben, coliseo, pajaro, estrella de mar, entre otras. En la figura 3.4 se presentan ejemplos de imágenes y sketches asociados a estas clases.

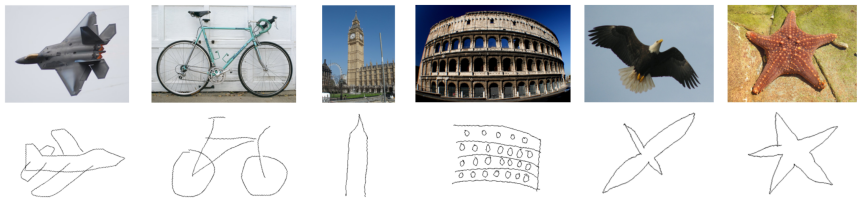

Figura 3.4: Ejemplos de imágenes y sketches del dataset Flickr15K.

Dataset de Saavedra

El dataset de Saavedra [24] esta enfocado en *fine grained SBIR* y cuenta con 1326 imágenes y 53 dibujos, con un total de 52 clases, entre las cuales se encuentran acordeón, barril, brontosaurio, motocicleta, revólver, langosta, entre otros; cada dibujo tiene además una imagen asociada. Se presenta en la figura 3.5 imágenes y dibujos de las clases anteriormente mencionadas.

Figura 3.5: Ejemplos de imágenes y sketches del dataset de Saavedra.

3.2.2. Recuperación de imágenes basada en dibujos con color

Para la recuperación de imágenes basada en dibujos con color no existen datasets de dibujos con color en la literatura, por ello se generan sintéticamente dibujos con color a partir de datasets de fotografías. Además, debido a que se quiere comparar el desempeño de métodos recuperando imágenes de colores similares, se generan imágenes de diversos colores a partir de imágenes fuente. Los datasets utilizados para este propósito son:

Dataset de artículos de hogar

Este dataset privado se utiliza para entrenamiento y corresponde a un dataset de catálogo, con productos de hogar, que cuenta con cerca de 6.000 imágenes y 132 clases; dentro de los productos se tiene accesorios de cocina, de baño, de habitaciones, de living, entre otros. Se presentan ejemplos de imágenes en la figura 3.6. Se utiliza este dataset debido a que las fotografías son de buena calidad y se pueden procesar fácilmente para generar otras imágenes.

Figura 3.6: Ejemplos de imágenes del dataset de artículos de hogar.

Generación de imágenes de distintos colores

Debido a que el sistema a implementar debe ser capaz de distinguir color, se crean nuevas imágenes de diversos colores a partir de las imágenes del dataset; esto tomando las imágenes originales, convirtiéndolas al espacio de color HSV, y sumando una componente constante al canal Hue; además se generan imágenes en escala de grises convirtiendo la imagen fuente a escala de grises y aplicando operaciones que modifican el contraste de la imagen, de modo de

Figura 3.7: Imágenes con distinto color generadas a partir de una imagen de referencia.

hacerlas más claras u obscuras. Se puede ver un ejemplo de las imágenes generadas en la figura 3.7. Una vez aplicada esta metodología el dataset resulta con cerca de 20.000 fotografías.

Generación de dibujos con color

Ya que no existe un dataset de dibujos con color, se propone una metodología de generación de dibujos con color a partir de fotografías; este proceso tiene como objetivo reducir la cantidad de detalles, reducir el número de colores y resaltar los bordes en las fotografías. El primer paso consiste en aplicar un filtro de suavizado con preservación de bordes [8] obteniendo una imagen suavizada, luego se aplica *Canny* sobre esta imagen suavizada y se obtienen los bordes de la imagen, posteriormente se aplica *k-means clustering* sobre la imagen suavizada con el fin de reducir la cantidad de colores y se obtiene el color de la imagen, finalmente se unen bordes y color para formar el sketch sintetizado. Se presenta en la figura 3.8 el diagrama de este proceso. Este proceso se aplica para las 20.000 imágenes del dataset, presentando resultados en la figura 3.9, donde en la parte superior se presentan las imágenes originales y en la parte inferior se presentan los sketch generados a partir de estas imágenes.

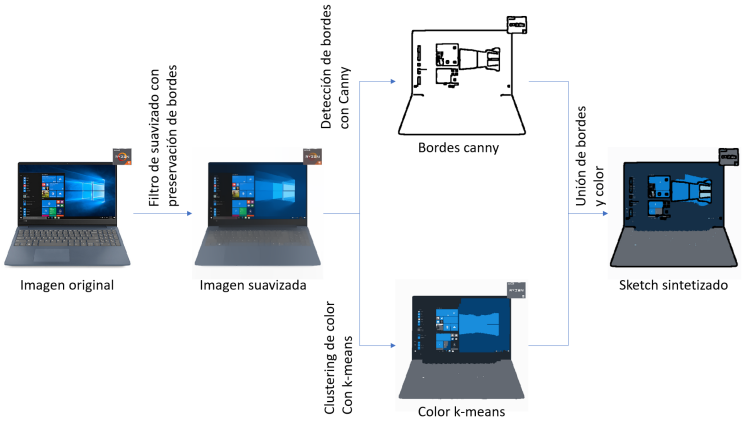

Figura 3.8: Diagrama del proceso de generación de sketches con color.

Figura 3.9: Dibujos con color generados y sus imágenes de referencia.

Dataset de retail y juguetería

Este dataset privado se utiliza como conjunto de test, originalmente consiste en 286 clases y 11.000 imágenes de productos de juguetería, ropa, artículos para bebés, hogar, electrodomésticos, deportes, entre otros. Se presenta en la figura 3.10 un ejemplo de productos del dataset. Al igual que con el dataset de artículos de hogar, se generan imágenes de distintos colores a partir de las imágenes originales, alcanzando así un total de aproximadamente 46.000 imágenes de test.

Figura 3.10: Imágenes del dataset de retail y juguetería.

Adquisición de dibujos de test

Para la adquisición de dibujos con color se plantea que estos dibujos deben estar separados en dos componentes, una componente de color y una componente de forma (bordes), esto con el fin de que ambas componentes puedan ser analizadas de manera separada o de manera conjunta en el sistema de recuperación de imágenes.

Para realizar esto se implementa un sistema de adquisición de dibujos que funcione en dos pasos, el primer paso corresponde a dibujar los bordes de la imagen (con color negro) sobre fondo blanco, obteniendo así la imagen de bordes. A continuación se debe pintar esta imagen, para ello se debe dejar la imagen de bordes fija y pintar sobre ella, de modo que los bordes se mantengan estáticos en primer plano y el color se dibuje atrás de los bordes. Cuando el usuario realice el dibujo se le mostrará en pantalla la imagen completa (bordes más color), sin embargo al guardar el dibujo se guardará por separado la componente de bordes y la componente de color. Este sistema se desarrolla en python con el uso de la librería OpenCV.

Para realizar la validación y test de los sistemas entrenados se dibujan a mano 100 sketches para el dataset de artículos de hogar, y 200 sketches para el dataset de retail y juguetería, se presentan ejemplos de estos sketches en la figura 3.11.

Figura 3.11: Sketches dibujados a mano para test, dataset de retail y juguetería.

3.3. Sistema de recuperación de imágenes basada en dibujos sin color (SBIR)

Para la implementación de este sistema se plantea comparar las propuestas de Saavedra [28], de Qi [19] y de Bui [3], descritas en la sección 2.2.2 del estado del arte. Se comparan las tres metodologías debido a que las dos primeras corresponden a métodos más estándar de redes convolucionales, mientras que la tercera corresponde a un método más reciente y que reporta mejores resultados. A continuación se describe en detalle la implementación de estas metodologías.

3.3.1. Deep SBIR

Para la extracción de características se entrena una red convolucional para clasificación de sketches, para esto utilizan los sketches del dataset Flickr25k, en conjunto con las fotografías del dataset, a las cuales se le aplica previamente detección de bordes de Canny. La red convolucional utilizada corresponde a una red Alexnet, preentrenada en el dataset Imagenet, mientras que la función de pérdida utilizada en el entrenamiento corresponde al *cross entropy loss*. Una vez entrenada la red, para la extracción de características de sketches se evalúan las deep features entregadas por la capa fc7 de la red.

Además, sobre estas características se aplica una normalización del tipo **square root normalization**, definida como:

$$sqrt\_norm(x) = unit(sign(x) \cdot \sqrt{(|x|)}) \tag{3.1}$$

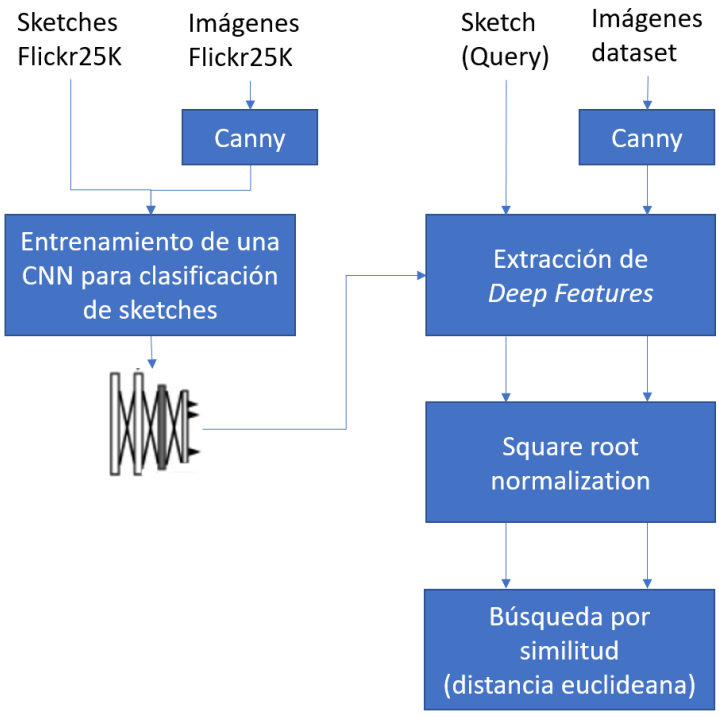

Figura 3.12: Esquema del proceso implementado en Deep SBIR.

Donde $sign(x)$ es un vector que toma valores 1 o -1 dependiendo del valor de x en cada posición, y *unit* hace referencia a normalización l2. Esta normalización evita el *bursting effect*, que es cuando una gran diferencia en unas pocas dimensiones del vector de características lleva a una gran distancia entre vectores, lo cual ocurre cuando hay unas pocas dimensiones que alcanzan un valor muy elevado, con lo que el resto de las dimensiones tienden a ser ignoradas.

Finalmente, la similitud se calcula como la distancia euclideana entre las características del sketch y de las imágenes. Se puede ver el diagrama de este método en la figura 3.12.

Con el fin de evaluar distintas arquitecturas de redes, esta metodología se implementó con *AlexNet*, *ResNet 50*, *Inception v4* y *SE-ResNext 50*.

3.3.2. SBIR a través de redes convolucionales siamesas

Esta metodología plantea abordar la extracción de características mediante redes convolucionales siamesas. La red siamesa recibirá en una de sus ramas la *query* (dibujo) y en la otra de sus ramas los bordes de las imágenes del dataset, extraídos mediante *Canny*; además, ambas ramas comparten todos sus pesos. En la figura 3.13 se presenta un diagrama del entrenamiento del sistema.

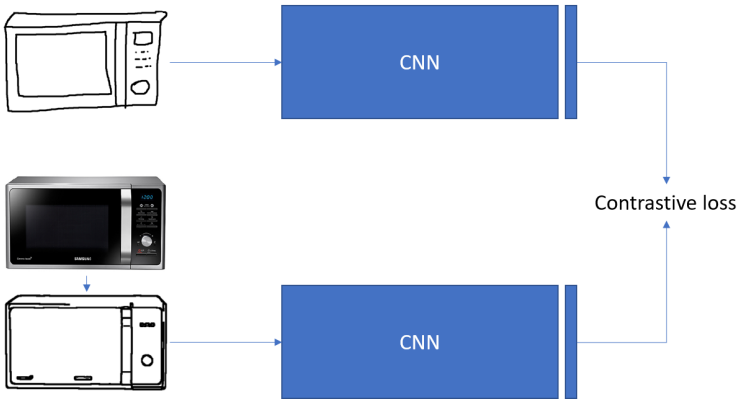

Figura 3.13: Diagrama del entrenamiento de red siamesa.

La red se entrena con el dataset Flickr25K, utilizando como función de pérdida el *contrastive loss*, donde se considera que dos imágenes son similares si pertenecen a la misma clase y son diferentes si pertenecen a clases distintas, y los pares positivos y negativos son elegidos con igual probabilidad. Este método se implementó solo con la red AlexNet, con el fin de comparar su desempeño con *Deep SBIR*, y evaluar las diferencias de ambas funciones de pérdida durante el entrenamiento.

3.3.3. SBIR usando CNN con entrenamiento en múltiples etapas

Bui et al. [3] plantea diversas mejoras para abordar el problema de SBIR. En primer lugar se plantea la utilización de dos redes para extracción de características, una para fotografías y la otra para sketches; estas redes poseen la misma arquitectura pero tienen pesos independientes en las primeras capas, con el fin de aprender distintos filtros para imágenes y sketches, de acuerdo a sus dominios, y comparten los pesos de las últimas capas con el fin de extraer características comunes a ambos dominios; además se plantea un entrenamiento en múltiples etapas, donde se va aumentando la complejidad de la función de pérdida y del entrenamiento con el fin de que las redes aprendan incrementalmente.

El procedimiento del paper es modificado con el fin de simplificar el entrenamiento; así, en el paper original se tenían 4 etapas de entrenamiento, pero la implementación realizada mezcla la segunda y la tercera, reduciendo entonces el entrenamiento a los tres pasos presentados en la figura 3.14, los cuales son descritos a continuación.

Etapa 1, entrenamiento de pesos no compartidos: En esta primera etapa se tiene por objetivo aprender los pesos de las primeras capas (pesos no compartidos), los cuales serán específicos de cada dominio; para esto se entrena tanto la red de imágenes como la red de sketches por separado, usando como función de pérdida el *cross entropy loss*. Los datos para entrenar corresponden a las imágenes de Flickr25K (directamente, sin aplicar Canny) y a los sketches de Flickr25K.

Etapa 2, entrenamiento de toda la red: Esta etapa tiene por objetivo aprender un embedding que sea común para las imágenes y los dibujos, a través de los pesos compartidos, en esta parte el entrenamiento se realiza en conjunto para ambas redes, como si fuera una red

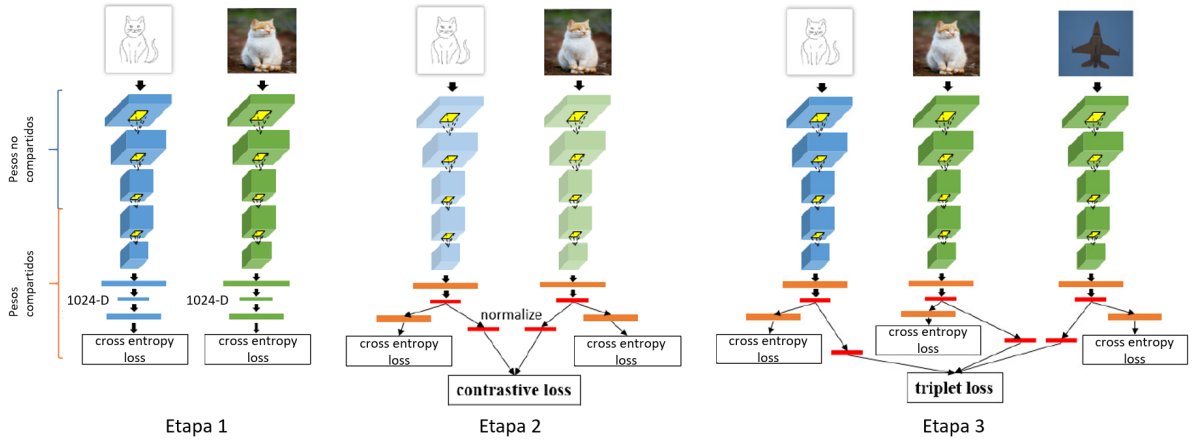

Figura 3.14: Etapas de entrenamiento del método *multi stage regression*

siamesa, así, el entrenamiento se realiza con pares de imágenes, donde un par es positivo si el sketch y la imagen pertenecen a la misma clase y es negativo en caso contrario. La función de pérdida utilizada corresponde a la suma del contrastive loss junto al cross entropy de ambas redes (con el fin de guiar el entrenamiento). El embedding aprendido es la capa mostrada en rojo en la figura 3.14, la cual se normaliza antes de utilizar el contrastive loss.

Etapa 3, fine tuning: En esta etapa se realiza un fine-tuning de la red, con el fin de que el sistema sea capaz de recuperar instancias de objetos (*fine grained SBIR*) y de mejorar el desempeño del sistema.

Para esto se utiliza el dataset Sketchy (que contiene dibujos basados en las fotografías del dataset), y se entrenan las redes usando *triplet networks*, usando una rama para dibujos (*anchor*) y las otras dos ramas para imágenes, donde la primera corresponde a imágenes positivas, que serán imágenes de la misma instancia que el dibujo *anchor*, y la segunda red corresponde a imágenes negativas, que pueden ser tanto imágenes de distinta clase, como imágenes de la misma clase pero de una distinta instancia.

La función de pérdida utilizada en esta sección corresponde a:

$$loss_{part3} = CE_1 + CE_2 + CE_3 + \alpha \cdot triplet\ loss \tag{3.2}$$

Donde CE corresponde al *cross entropy loss* de cada una de las ramas, para guiar el entrenamiento; *triplet loss* corresponde al loss de las *triplet networks* y α corresponde a un ponderador que comienza en 2, y que en cada época se aumenta en 0.5 para aumentar la relevancia del triplet loss en la función de pérdida a minimizar.

Arquitectura

Respecto a la arquitectura utilizada para esta metodología, esta corresponde a SE-ResNext 50, que cuenta con 5 grandes bloques de operaciones de convolución, de los cuales los tres primeros poseen pesos independientes para ambas redes (marcados en la figura 3.14 como

pesos no compartidos), mientras que los dos últimos bloques comparten sus pesos para la red de sketches y de imágenes. A la salida de la red se realiza un *flatten* de las capas convolucionales, luego se utiliza una capa *fully connected* de dimensión 2048 con función de activación *leaky relu*; a continuación de esta capa se tiene una capa *fully connected* que será nuestro *embedding* o vector de características, que por defecto es de dimensión 1024 y posee una función de activación lineal a la salida. Finalmente, para entrenar la red mediante clasificación este embedding será conectado a una capa de salida de dimensión igual al número de clases, y para las redes siamesas o triplets este vector de salida será normalizado mediante norma L2 y luego se utiliza el vector normalizado en el contrastive loss o triplet loss. Se destaca que finalmente **el vector de características corresponde al *embedding* normalizado**.

Dentro de los experimentos realizados para este método se encuentra verificar el comportamiento del sistema ante distintas dimensiones del vector de características, y visualizar el embedding entregado por la red.

3.4. Sistema de recuperación de imágenes basada en dibujos con color (CSBIR)

Para implementar el sistema de recuperación de imágenes basada en dibujos con color se deben extraer tanto características de forma y de color de las imágenes y de los dibujos, estas características se pueden extraer de manera independiente (lo cual se presentará a continuación como el *baseline*) o de manera conjunta. **Para testear este sistema se utiliza el dataset de retail y juguetería**, y sobre los dibujos con color realizados a mano, como los presentados en la figura 3.11 se evaluarán distintas métricas de desempeño, como el Mean Reciprocal Rank o el Recall Ratio. A continuación se describen los dos métodos implementados para abordar este problema:

3.4.1. CSBIR con histogramas de color

Se propone para este sistema realizar una extracción de características de forma y de color de manera separada, tanto para el dibujo de entrada como para las imágenes del dataset. Formalmente:

Dada una query (sketch con color) S, formada por la imagen de bordes S_{e} y la imagen de color S_c, e imágenes del dataset X_{i}; se deben extraer características de forma de la componente de bordes $f_s(S_{\mathrm{e}})$ y características de color de la componente de color $g_s(S_c)$, y también características de forma y de color de las imágenes X_{i} del dataset, las cuales llamaremos $f_x(X_{\mathrm{i}})$ y $g_x(X_{\mathrm{i}})$ respectivamente (ver figura 3.15); de modo que la distancia $l_2(f_s(S_{\mathrm{e}}) - f_x(X_{\mathrm{i}}))$ entre las características de forma de la query y de las imágenes sea pequeña si ambas imágenes tienen forma similar y grande si ambas imágenes tienen forma diferente; y la distancia $l_2(g_s(S_c) - g_x(X_{\mathrm{i}}))$ entre las características de color del dibujo y las imágenes sea pequeña si el color es similar y grande si tienen colores distintos. Finalmente, la métrica de similitud D entre la query S y una imagen X_{i} del dataset estará dada por:

$$D = \gamma \cdot l_2(f_s(S_{\mathrm{e}}) - f_x(X_{\mathrm{i}})) + (1 - \gamma) \cdot l_2(g_s(S_c) - g_x(X_{\mathrm{i}})) \tag{3.3}$$

Es decir, la medida de similitud entre la query S y una imagen X_i estará dada por una suma ponderada de la norma l_2 de la diferencia entre las características de forma de la query y la imagen, y la norma l_2 de la diferencia entre las características de color de la query y la imagen. Finalmente, para retornar las imágenes más similares se debe calcular la distancia D_i entre S y X_i $\forall$i en el dataset, ordenar de menor a mayor distancia y retornar las imágenes que presenten una menor distancia.

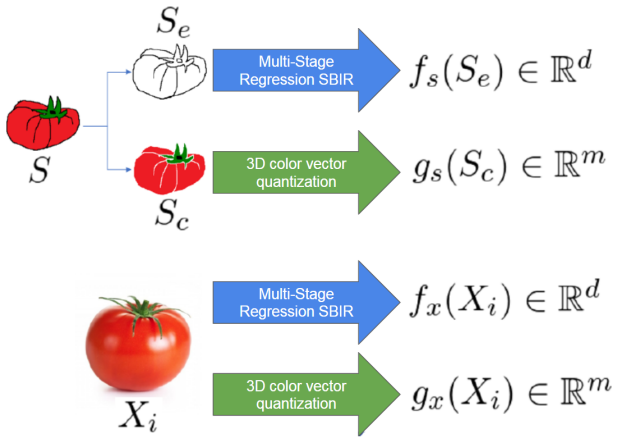

Figura 3.15: Extracción de características del dibujo y de una imagen del dataset.

La extracción de características de forma se realiza mediante el método de entrenamiento en múltiples etapas, detallado en la sección anterior, mientras que la extracción de características de color se explica en la siguiente sección.

Extracción de características de color

Para la obtención de características de color de las imágenes se utilizarán histogramas de color. Estos histogramas se evaluarán en el espacio de color RGB.

El método a utilizar corresponde en particular a *3D color vector quantization* [1]. La idea principal de este método es discretizar el espacio de color, correspondiente a un espacio 3D (RGB), de manera que la discretización sea uniforme. Esto se realiza con el objetivo de disminuir la complejidad del espacio de color y agrupar colores cercanos en este espacio. Se puede observar en la figura 3.16 esta discretización. Si se utilizaran directamente histogramas de color sobre la imagen se obtendrían 256x256x256 posiciones en el histograma, por lo cual la dimensionalidad de este espacio sería muy grande. El método propone entonces realizar una cuantización del espacio de color en una grilla de 5x5x5, de modo de tener un total de 125 características.

Para incorporar información espacial a las características de color se propone además discretizar la imagen en una grilla de tamaño 2x2; así, se tendrían en total $2 \cdot 2 \cdot 5 \cdot 5 \cdot 5 = 500$ características de color en total para cada imagen. Esto se observa en la figura 3.17, donde se usa una grilla de tamaño 4x4 en la imágen.

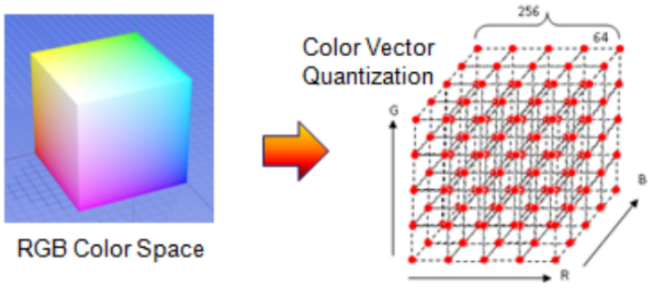

Figura 3.16: Cuantización del espacio de color RGB.

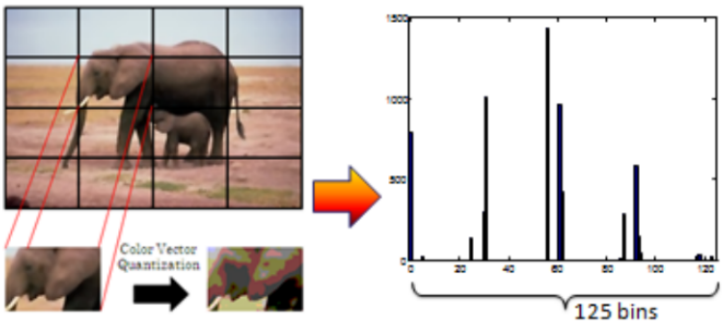

Figura 3.17: Cuantización de la imagen en grillas.

3.4.2. CSBIR con *Quadruplet Networks*

Se propone esta nueva metodología para abordar el problema de CSBIR a partir del método de entrenamiento en múltiples etapas (descrito en la sección de SBIR), pero adaptándolo para incorporar información de color; este método además se caracteriza por extraer en forma conjunta características de forma y de color de las imágenes y de los dibujos.

Para comprender la hipótesis a cumplir por el método se plantea en la figura 3.18 un sketch de consulta q y tres fotografías distintas, la primera p_+ de la misma instancia y color que el sketch, la segunda p_{+-} de la misma instancia y distinto color que la consulta y la tercera p_- de distinta clase. La hipótesis planteada es la siguiente:

$$dist(q, p_+) < dist(q, p_{+-}) < dist(q, p_-) \tag{3.4}$$

El objetivo es entonces implementar un método que sea capaz de extraer un embedding que cumpla con las relaciones planteadas en la ecuación 3.4; para ello recordamos que las *triplet networks* tienen como objetivos cumplir la desigualdad de la ecuación 2.8, que plantea que la distancia entre el anchor y el par positivo debe ser menor a la distancia entre el anchor y el par negativo, separadas por un margen λ. Esta idea se puede aplicar al separar nuestra hipótesis 3.4 en dos desigualdades y agregando un margen, obteniendo las ecuaciones 3.5 y 3.6:

$$dist(q, p_+) + \alpha \cdot \lambda < dist(q, p_{+-}) \tag{3.5}$$

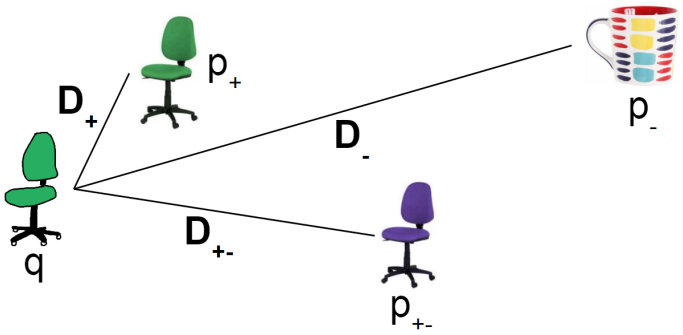

Figura 3.18: Hipótesis del método de CSBIR con *Quadruplet Networks*

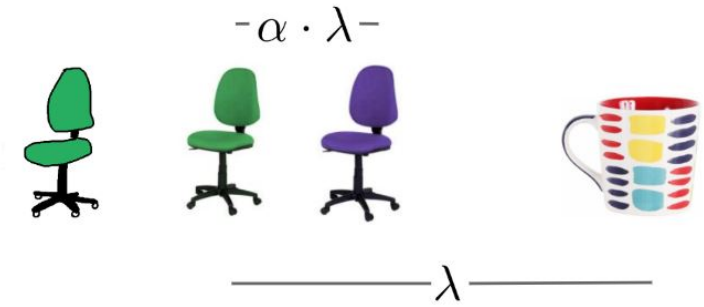

Figura 3.19: Proyección de las distancias objetivo a una dimensión y visualización del efecto de λ y α.

$$dist(q, p_{+-}) + (1 - \alpha) \cdot \lambda < dist(q, im_{-}) \tag{3.6}$$

Juntando las ecuaciones 3.5 y 3.6 se obtiene la desigualdad:

$$dist(q, im_{+}) + \lambda < dist(q, im_{-}) \tag{3.7}$$

En las expresiones anteriores se tiene que λ es el margen que existirá para un ejemplo de la misma instancia y color con un ejemplo de distinta clase. El ponderador α nos indica que tan cercano debe ser una imagen de la misma instancia pero distinto color a una imagen de la misma instancia y color que la query; se eligen valores de α cercanos a 0.25, lo que implica que la imagen p_{+-} estará más cercana a la imagen p_{+} que a la imagen p_{-}. Esto se puede observar gráficamente en la figura 3.19 al proyectar las distancias a una dimensión, donde se puede ver el efecto de α y λ.

Para llevar esto a un contexto de redes neuronales, las desigualdades 3.5 y 3.6 pueden convertirse en funciones objetivo de la siguiente manera:

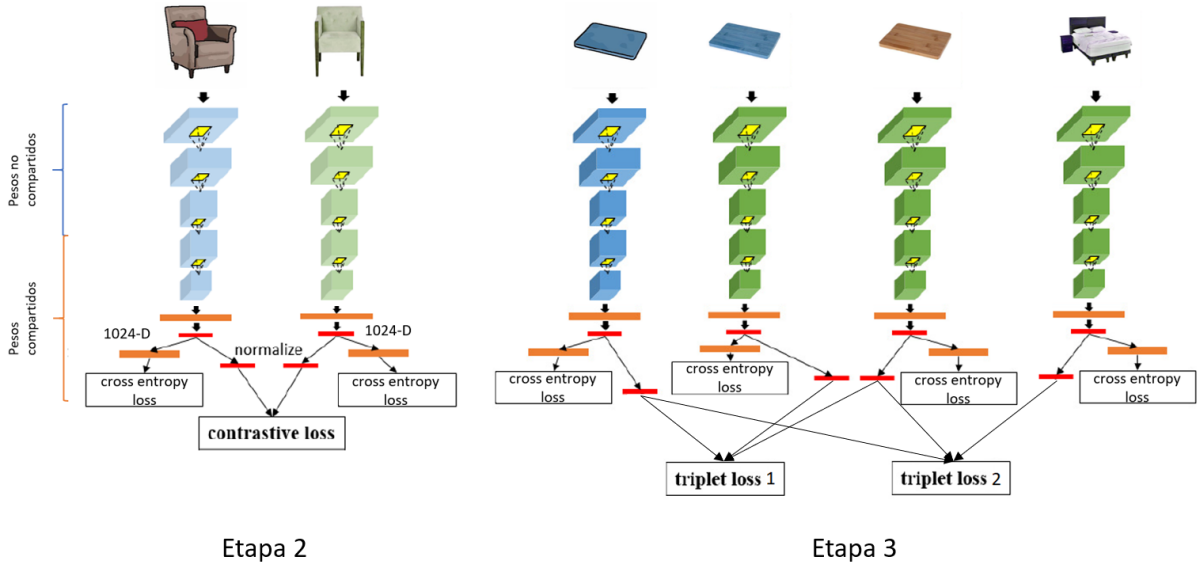

Figura 3.20: Etapas de entrenamiento del método *Quadruplet Networks*.

$$triplet\ loss_1 = max\{0, dist(F(q), G(p_+)) + \alpha \cdot \lambda - dist(F(q), G(p_{+-}))\} \tag{3.8}$$

$$triplet\ loss_2 = max\{0, dist(F(q), G(p_{+-})) + (1-\alpha) \cdot \lambda - dist(F(q), G(p_-))\} \tag{3.9}$$

Donde F es el embedding formado por la red de dibujos y G es el embedding formado por la red de imágenes.

Con esto en mente, el proceso seguido para entrenar el sistema se presenta en la figura 3.20 (exceptuando la etapa 1 que se mantiene igual al método de entrenamiento en múltiples etapas presentado en la figura 3.14).

Las etapas del entrenamiento consisten en lo siguiente:

Etapa 1, entrenamiento de pesos no compartidos: Esta etapa tiene por objetivo entrenar los pesos de las primeras capas convolucionales (no compartidos), que serán distintos para fotografías y para dibujos con color; para ello se entrenan ambas redes por separado, considerando el problema como un problema de clasificación, donde la función objetivo a minimizar corresponde al *cross entropy loss*.

Etapa 2, entrenamiento de toda la red: El objetivo de esta etapa es aprender un *embedding* común para las imágenes y dibujos, a través de los pesos compartidos. El entrenamiento se realiza en ambas redes en conjunto, alimentando las redes con pares fotografía-dibujo, donde un par es positivo si ambas imágenes pertenecen a la misma clase y negativo en caso contrario, y tanto pares positivos como negativos son seleccionados con la misma probabilidad. La función de pérdida corresponde a la suma del *cross entropy loss* de ambas ramas, junto al *contrastive loss*; esto con el fin de formar *embeddings* en que las imágenes estén agrupadas por clase. En esta etapa aún no se considera el color.

Etapa 3, aprendizaje conjunto de color y forma: El objetivo de esta etapa es que la red sea capaz de aprender características de color y de forma a nivel de detalle, con el fin de alcanzar las desigualdades de la ecuación 3.4 para todas las imágenes del dataset. Para esto se entrena una *quadruplet network*, tal que la primera rama contenga los sketches (*anchors*), y las otras tres ramas contengan fotografías, donde la primera de estas ramas de fotografías tenga la foto de referencia para el sketch, la segunda tenga la misma foto pero de un color diferente y la tercera rama tenga una foto de distinta clase.

La función de pérdida a minimizar es la siguiente:

$$loss_{part3} = CE_1 + CE_2 + CE_3 + CE_4 + \beta \cdot (triplet\ loss_1 + triplet\ loss_2) \tag{3.10}$$

Que corresponde a la suma del cross entropy para las cuatro redes, con el fin de guiar el entrenamiento, más un ponderador β multiplicado por la suma de los *triplet loss* de las ecuaciones 3.8 y 3.9, que estarán orientados a que la red ordene las imágenes de acuerdo a la desigualdad objetivo. El ponderador β utilizado parte en 2 y aumenta 0,5 en cada época. El método se entrena por 25 épocas.

3.5. Ajustes experimentales

Los múltiples experimentos con redes convolucionales presentaron los siguientes ajustes experimentales:

- Cada método o etapa fue entrenado por un máximo de 10 épocas (a menos que se haya indicado lo contrario) y se utilizó un conjunto de validación para elegir la mejor época.
- Se utilizó el optimizador ADAM con un learning rate de 10^{-4}.
- Las imágenes de entrada de las redes son de tamaño $224x224$.
- Los sketch son normalizados antes de entrar a la red, de modo de centrarlos, y utilizar un 90 % del tamaño de la imagen en su eje más largo.
- Al realizar *resize* de las imágenes se mantiene el *aspect ratio*, con el fin de no deformar las imágenes.
- Se aplica *data augmentation* con las siguientes operaciones: *random horizontal flip*, *color augmentation* (solo para imágenes y dibujos con color, consiste en cambios de saturación, pequeñas alteraciones en el canal Hue, cambios de contraste), *resize* (consistente en agrandar o achicar la imagen en un margen de 10 %) y *random rotation* (en un margen de -20 a 20 grados).

Capítulo 4

Resultados y discusión

4.1. Resultados

A continuación se presentan los principales resultados y métricas de desempeño para las distintas metodologías desarrolladas. Para evaluar el desempeño de los sistemas se utilizan las métricas mean average precision (mAP), mean reciprocal rank (MRR), recall ratio, y se muestran ejemplos gráficos del desempeño del sistema.

4.1.1. Sistema de recuperación de imágenes basada en dibujos sin color (SBIR)

En esta sección se presentan los resultados de los métodos de recuperación de imágenes basados en dibujos sin color. **La evaluación se realiza sobre los dataset *Flickr15k* y de *Saavedra*.**

Para comparar visualmente los tres métodos desarrollados entre sí, se presentan en las figuras 4.1, 4.3 y 4.5 resultados de recuperación de imágenes en el dataset *Flickr15k* para los tres métodos, esto utilizando 5 dibujos seleccionados al azar del dataset, y mostrando las 10 imágenes más similares de acuerdo a cada método. Además se presentan en las figuras 4.2, 4.4 y 4.6 resultados para el dataset de *Saavedra*. Se destaca que **en cada imagen se marca en rojo las fotografías incorrectas recuperadas**, es decir, aquellas que no corresponden a la misma clase del dibujo.

Deep SBIR

El primer resultado extraído corresponde a la metodología *Deep SBIR*. Para comparar los resultados se calcula el mAP para los dataset de test, correspondientes a *Flickr 15K* y *Saavedra*. Además, con el fin de elegir la mejor arquitectura se comparan las arquitecturas *AlexNet*, *ResNet 50*, *Inception v4* y *SE ResNext 50*. Los resultados se presentan en la tabla 4.1.

Implementación	Accuracy validación	mAP Flickr 15k	mAP Saavedra
Paper Original	-	0.282	0.326
AlexNet	0.515	0.172±0.020	0.243±0.066
ResNet 50	0.722	0.244±0.025	0.370±0.060
Inception v4	0.736	0.070±0.008	0.140±0.041
SE ResNext 50	0.686	**0.326±0.028**	**0.424±0.063**

Tabla 4.1: Resultados DeepSBIR

Junto a estos resultados, se presenta en las figuras 4.1 y 4.2 ejemplos de imágenes recuperadas en ambos *datasets* de test, usando la arquitectura *SE ResNext 50*.

Figura 4.1: Ejemplos de imágenes recuperadas con Deep SBIR para el dataset Flickr 15K.

Figura 4.2: Ejemplos de imágenes recuperadas con Deep SBIR para el dataset de Saavedra.

SBIR a través de redes convolucionales siamesas

Para esta metodología se presenta en la tabla 4.2 el mAP de este sistema dado por el paper original y el de la implementación propuesta.

Implementación	mAP Flickr 15k	mAP Saavedra
Paper Original	0.195	-
AlexNet	0.137±0.017	0.172±0.046

Tabla 4.2: Resultados redes siamesas

Además, se presenta en las figuras 4.3 y 4.4 ejemplos de imágenes recuperadas con este sistema para los datasets Flickr 15K y Saavedra respectivamente.

Figura 4.3: Ejemplos de imágenes recuperadas con Siamese SBIR para el dataset Flickr 15K.

Figura 4.4: Ejemplos de imágenes recuperadas con Siamese SBIR para el dataset de Saavedra.

SBIR usando CNN con entrenamiento en múltiples etapas

Se presenta en la tabla 4.3 el mAP de este sistema reportado por el paper en sus distintas etapas de entrenamiento y el de la implementación propuesta.

Implementación	mAP Flickr 15k	mAP Saavedra
Paper Original (Etapa 2)	0.348	0.572
Paper Original (Etapa 3)	0.411	0.633
Paper Original (Etapa 4)	0.511	0.659
Implementación (Etapa 2)	0.523±0.034	0.487±0.073
Implementación (Etapa 3)	0.553±0.033	0.506±0.065

Tabla 4.3: Resultados Multi Stage Regression

Se presenta además en las figuras 4.5 y 4.6 ejemplos de imágenes recuperadas con este sistema para el dataset Flickr 15K y de Saavedra respectivamente.

Figura 4.5: Ejemplos de imágenes recuperadas con SBIR mediante entrenamiento en múltiples etapas para el dataset Flickr 15K.

Figura 4.6: Ejemplos de imágenes recuperadas con SBIR mediante entrenamiento en múltiples etapas para el dataset de Saavedra.

Junto a estos resultados, se experimentó cambiando la dimensionalidad del vector de características, con el fin de observar su efecto en el desempeño y en el tiempo de búsqueda. Para esto se realiza un fine-tuning (etapa 3 del entrenamiento) del sistema cambiando esta dimensionalidad. Se presenta en la tabla 4.4 el desempeño del sistema (mAP) y el tiempo de búsqueda para ambos dataset, considerando una distinta dimensionalidad del espacio de características. El tiempo de búsqueda hace referencia al tiempo en ordenar todas las imágenes del catálogo por su distancia con una query, es decir, realizar una búsqueda lineal.

		Dimensionalidad del embedding			
		256	**512**	**1024**	**2048**
Flickr 15K	mAP	0.502	0.497	0.553	0.534
	Tiempo de búsqueda [ms]	83.67	90.11	97.07	107.24
	Tamaño en disco [MB]	18.4	32.9	61.9	119.9
Saavedra	mAP	0.404	0.465	0.506	0.534
	Tiempo de búsqueda [ms]	8.26	8.62	8.70	10.34
	Tamaño en disco [MB]	1.7	3.1	5.7	11.0

Tabla 4.4: mAP y tiempo de búsqueda para distinta dimensionalidad del embedding.

Por último, se presenta en la figura 4.7 una visualización del embedding generado por la red en la etapa 3 del entrenamiento, para el dataset Flickr 15K. Para realizar la proyección a dos dimensiones se utiliza tSNE.

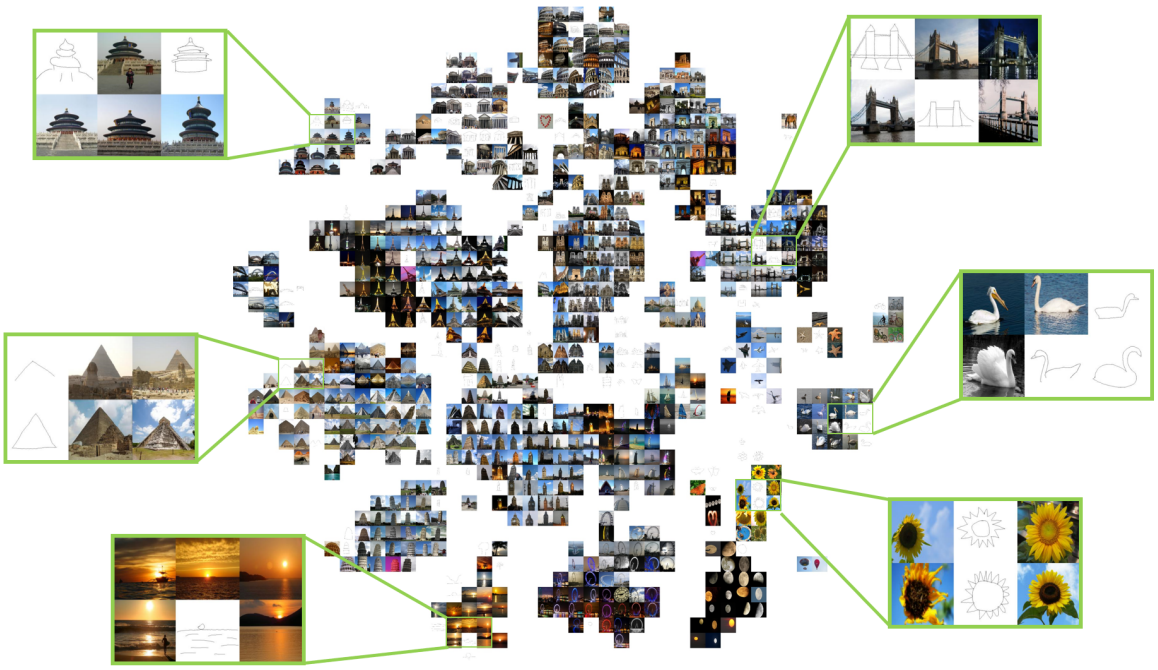

Figura 4.7: Visualización del embedding generado por la etapa 3 del entrenamiento, para el dataset Flickr15k.

4.1.2. Sistema de recuperación de imágenes basada en dibujos con color (CSBIR)

En esta sección se presentan los resultados obtenidos para ambos métodos propuestos de recuperación de imágenes basada en dibujos con color. Para comparar los resultados obtenidos, se presenta el mean Average Precision (mAP) por clase, el mean reciprocal rank y el recall ratio, en conjunto a diversos ejemplos de imágenes recuperadas.

El conjunto de test utilizado en ambos casos corresponde al dataset de retail y juguetería, consistente en 200 dibujos con color realizados a mano y 46000 fotos (las 11000 originales en conjunto a las fotos con color generadas), con un total de 286 clases. Para calcular el mRR y el recall ratio se considera que cada dibujo tiene una sola imagen correcta, correspondiente a la imagen desde la cual fue realizada el dibujo.

CSBIR con histogramas de color

Para este método, el primer parámetro a analizar corresponde a γ, esto es, el *trade-off* entre las características de forma y de color, así, un valor de $\gamma = 0$ implica solo considerar características de forma, mientras que $\gamma = 1$ implica utilizar solo características de color. Para realizar la selección de este parámetro se presenta en la figura 4.8 el MRR y el mAP al considerar distintos valores del mismo. De acuerdo a estos resultados, se opta por seleccionar $\gamma = 0{,}6$ como el valor más adecuado para este parámetro, debido a que alcanza el mayor MRR a costa de una leve disminución respecto al mayor mAP. Las métricas de desempeño obtenidas con $\gamma = 0{,}6$ corresponden a $MRR = 0{,}216$ y un $mAP = 0{,}1324$.

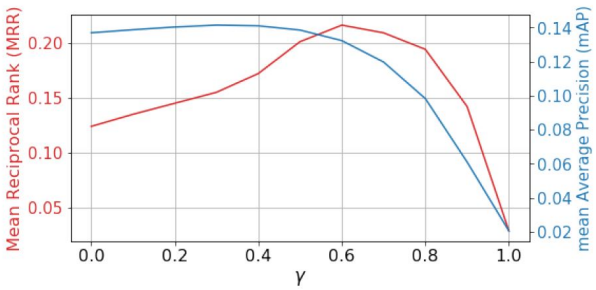

Figura 4.8: Métricas de desempeño para distintos valores de γ, dataset de retail y juguetería.

Para complementar, en la figura 4.9 se presentan ejemplos de imágenes recuperadas para dos consultas, al considerar distintos valores de γ, en esta figura se aprecia claramente el efecto de este parámetro sobre las imágenes recuperadas, mencionado en el párrafo anterior.

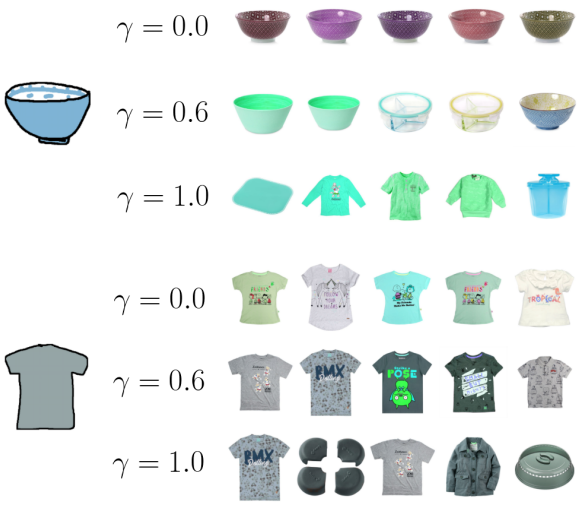

Figura 4.9: Imágenes recuperadas para dos consultas, al considerar distinto γ.

Además, en la figura 4.10 se presentan ejemplos de las 20 primeras imágenes recuperadas para diversas consultas, utilizando $\gamma = 0.6$. **En la figura se enmarca en verde la fotografía correcta**.

CSBIR con *Quadruplet Networks*

Para evaluar el desempeño de este método se realizan experimentos entrenando la parte 3 con distintos valores del parámetro α, utilizando un valor de λ fijo igual a 1,5.

En la tabla 4.5 se presenta el MRR y el mAP en el conjunto de test; para la etapa 2 del entrenamiento, y para la etapa 3 al considerar distintos valores de $\alpha \in \{0.1, 0.25, 0.50, 0.75\}$. De acuerdo a la tabla, se escoge el parámetro $\alpha = 0{,}25$ como el más adecuado, ya que con este valor se presenta un MRR muy superior al resto, manteniendo un buen mAP. Los siguientes resultados son extraídos usando este valor de α.

	MRR	**mAP**
Etapa 2	0.2478	**0.1601**
Etapa 3 $\alpha = 0{,}10$	0.2987	0.1217
Etapa 3 $\alpha = 0{,}25$	**0.3519**	0.0977
Etapa 3 $\alpha = 0{,}50$	0.3228	0.0671
Etapa 3 $\alpha = 0{,}75$	0.2987	0.0552

Tabla 4.5: mAP y MRR para el método de *Quadruplet Networks* en distintas etapas del entrenamiento.

Para evaluar visualmente el resultado del método se presenta en las figuras 4.11 y 4.12

Figura 4.10: Ejemplos de imágenes recuperadas para distintas consultas para el método CS-BIR con histogramas de color utilizando $\gamma = 0{,}6$.

ejemplos de las 20 primeras imágenes recuperadas ante distintas consultas, para las etapas 2 y 3 del entrenamiento. En las figuras se marca en verde la fotografía correcta.

Para entender mejor el comportamiento de la red, se realiza una proyección del *embedding* extraído para los datos a 2 dimensiones, utilizando el método t-SNE; se presenta en la figura 4.13 uno de los *clusters* formados con esta proyección, correspondiente a coches para bebé.

Una de las principales ventajas del método propuesto es que además de poder realizar consultas a través de dibujos permite realizar consultas a través de fotografías, pudiendo recuperar imágenes de color similar; esto se realiza extrayendo características mediante la red de fotos. Se presenta así en la figura 4.14 ejemplos de imágenes recuperadas al realizar una búsqueda por imágenes.

Finalmente, para realizar una comparación más en detalle de ambos métodos desarrollados, se presenta en la figura 4.15 el *recall ratio* para el método de CSBIR con histogramas de color y para las etapas 2 y 3 del método *Quadruplet Networks*. Para la extracción del *recall ratio* se considera que para cada consulta solo existe una respuesta correcta, que corresponde a la imagen de referencia con la que fue dibujada la consulta.

4.2. Discusión

En esta sección se discuten los resultados obtenidos para los distintos métodos implementados; contrastando las métricas con los resultados visuales, y analizando que métodos son mejores, junto a las ventajas y desventajas de cada uno.

4.2.1. Sistema de recuperación de imágenes basada en dibujos sin color (SBIR)

Deep SBIR

Respecto a los resultados extraídos con la metodología *Deep SBIR*, el primer resultado interesante corresponde a la comparación entre distintas **arquitecturas de redes convolucionales**, para ello se observa en la tabla 4.1 que la arquitectura que consigue un mejor desempeño corresponde a Squeeze and Excitation ResNext 50, obteniendo un desempeño mejor que el paper original, y mejor versus a otras arquitecturas como ResNet 50 e Inception v4.

Sin embargo, se debe ser cuidadoso con la comparación, ya que la implementación del paper original utiliza una arquitectura *AlexNet*, obteniendo resultados mucho mejores que la implementación propia con esta arquitectura. Esto se asocia a diversos factores, enumerados a continuación:

- El paper original entrena la red con el dataset *QuickDraw*, este dataset contiene una cantidad mucho mayor de dibujos que el dataset utilizado durante el entrenamiento de la implementación propia, correspondiente a *Flickr 15K*.
- La implementación original utiliza *Sketch Tokens* para extraer bordes de las imágenes, consistente en un método supervisado para la extracción de bordes. Este método

Figura 4.11: Ejemplos de imágenes recuperadas ante distintas consultas para el método *Quadruplet Networks* en la etapa 2 del entrenamiento.

Figura 4.12: Ejemplos de imágenes recuperadas ante distintas consultas para el método *Quadruplet Networks* en la etapa 3 del entrenamiento, usando $\alpha = 0{,}25$.

Figura 4.13: *Cluster* de coches para bebé, visualizado a través de t-SNE sobre el dataset completo. En cuadros rojos se marcan dibujos de consulta.

Figura 4.14: Imágenes recuperadas al realizar búsqueda mediante fotos.

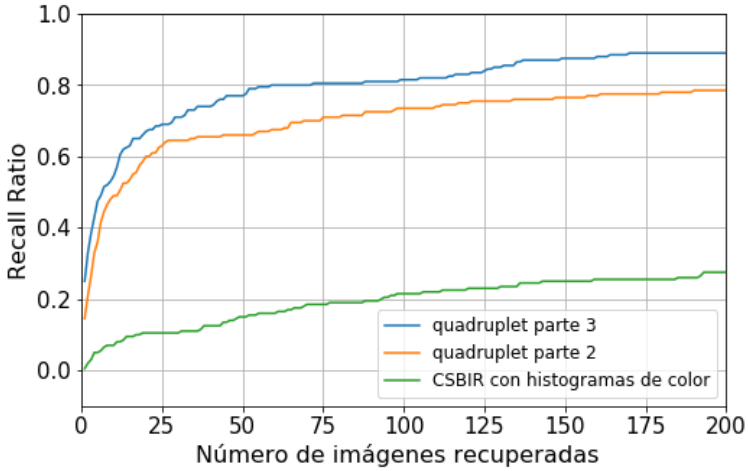

Figura 4.15: Recall ratio comparando los distintos métodos implementados.

presenta resultados bastante superiores al utilizado en la implementación propia, correspondiente a *Canny*, sin embargo, debido a su alto costo computacional se descarta su uso.

Se plantea que utilizar estas mejoras debiese aumentar el desempeño de la red.

Otro punto de los resultados que llama la atención es el comportamiento de la arquitectura *inception v4*, la cual presenta un mayor *accuracy* de clasificación en el conjunto de validación (el cual corresponde a un subconjunto del dataset *Flickr15k*), sin embargo posee un menor mAP en ambos conjuntos de test. Esto se puede interpretar como que la red tiende a sobreajustar su extracción de características a las clases del conjunto de entrenamiento, así, se debiese experimentar extraer características de una capa de la red más cercana a la entrada, con el fin de que las características extraídas dependan en mayor cantidad de los filtros aprendidos y no tanto de la salida de la red (que dependerá de las clases vistas durante el entrenamiento).

De los ejemplos visuales mostrados en la figura 4.1, se aprecia que el método recupera con mucha precisión las imágenes del dataset Flickr15k, mientras que de la figura 4.2 se extrae que para el dataset de Saavedra funciona bien, pero tiende a cometer más errores dentro de las 10 primeras imágenes recuperadas.

SBIR a través de redes convolucionales siamesas

Respecto a la implementación de SBIR a través de redes convolucionales siamesas, se muestra en la tabla 4.2 que la implementación propia del método tiene un peor desempeño respecto a la implementación del paper original. Esto puede ser debido a que en el paper original dividen el dataset *Flickr15k* la mitad para entrenamiento y la mitad para test, por lo que es muy posible que el método se haya sobreajustado a las clases vistas durante el entrenamiento, las cuales son las mismas que se utilizan para evaluar, aumentando así el desempeño del sistema. Por otro lado la implementación propia utiliza el dataset *Flickr25k* durante el entrenamiento y el dataset *Flickr15k* completo para test, ambos con clases distintas entre sí, por lo que es menos probable que se presente sobreajuste.

Se plantea que el mal desempeño de este método respecto a los otros métodos entrenados es el uso de la función de pérdida *contrastive loss* durante el entrenamiento, debido a que al realizar el entrenamiento esta función solo es capaz de comparar entre sí las clases presentadas en un batch; y no presenta una referencia global de todas las clases del dataset. Para entender mejor este punto, supongamos que se cuentan con 20 clases y se tiene un batch donde solo están presente las clases 1, 3 y 10; al momento de entrenar, la función de pérdida tiende a alejar estas clases entre sí, pero al hacer esto se puede estar acercando las características de la clase 1 con las características de una clase que no esté presente (por ejemplo de la clase 4); por otro lado, si se tiene otra función de pérdida como el *cross entropy loss* entonces se sabe el total de clases y tiene una referencia global de todas las clases presentes durante el entrenamiento (aún cuando el batch no presente todas las clases), por lo que será más fácil separar las clases entre si. Este problema se hace aún más notorio cuando se tienen muchas clases de entrenamiento y los batch son de tamaño pequeño, como es en nuestro caso, en que se presentan 250 clases de entrenamiento y batch de tamaño 30.

De las figuras 4.3 y 4.4 es posible visualizar que los métodos tienen un desempeño bastante pobre, recuperando imágenes que no tienen mayor similitud entre sí.

SBIR usando CNN con entrenamiento en múltiples etapas

Los principales resultados de este método son presentados en la tabla 4.3.

Lo primero que se recalca es que el paper original posee 4 etapas de entrenamiento, mientras que en la implementación propia solo se tienen 3 etapas (las etapas 2 y 3 se mezclaron en una única etapa); esto debido a que se experimentó implementar las etapas 2 y 3 del paper original por separado, sin embargo se comprobó experimentalmente que esto no tenía mayores ventajas, y aún más, que al realizar el entrenamiento de estas etapas por separado la etapa 3 disminuye el desempeño respecto a la etapa 2. Así, en la implementación propia las etapas 2 y 3 del paper original se mezclan en una única etapa (etapa 2), mientras que la etapa 4 del paper original pasó a ser la etapa 3 en la implementación propia. De esta manera el entrenamiento se simplifica y mejora su desempeño.

De los resultados presentados en la tabla 4.3 no es claro si la implementación propia es mejor que la del paper original, debido a que la implementación propia presenta un mayor mAP en el dataset *Flickr15k* pero un menor mAP en el dataset de *Saavedra*. Se plantea como hipótesis que la implementación propia funciona mejor que la implementación original cuando los dibujos de consulta no presentan un gran nivel de detalle; esto ya que la implementación realizada presenta un mayor mAP en el dataset *Flickr15k*, cuyos dibujos son simples. Por otro lado se plantea que la implementación del paper original funciona mejor cuando los dibujos presentan un gran nivel de detalle, pues alcanzan un mayor mAP en el dataset de *Saavedra*, cuyos dibujos contienen muchos detalles y se asemejan más a las fotografías a partir de las cuales se dibujaron.

De los resultados gráficos presentados en las figuras 4.5 y 4.6, y del mAP obtenido; se extrae que el método presenta mejores resultados respecto a los otros dos métodos probados, pudiendo recuperar dentro de los primeros 10 resultados en su gran mayoría imágenes pertenecientes a la misma clase y muy similares visualmente.

Un parámetro muy importante del método implementado es la dimensionalidad del vector de características, ya que vectores de características de mayor dimensión nos permiten representar con mayor flexibilidad los atributos de las imágenes, sin embargo hacen que el tiempo de búsqueda y que el espacio en disco utilizado por las características aumente, tal como se presenta en la tabla 4.4. De la tabla se extrae que el tiempo de búsqueda no varía mucho respecto al tamaño del vector de características, sino que varía más respecto a la cantidad de imágenes en el dataset; para afrontar este problema se puede experimentar el uso de algoritmos de búsqueda aproximada.

Un punto que es más crítico corresponde al tamaño en disco utilizado por los *embeddings* del dataset, el cual tiende a escalar mucho con la cantidad de imágenes en el dataset y con la dimensionalidad del vector de características; para ello, la solución más adecuada no parece ser el reducir la dimensionalidad del *embedding* directamente en la red, debido a que tiende a disminuir el mAP. Para afrontar este problema se puede experimentar el considerar un *embedding* de dimensionalidad alta (como 1024 o 2048) durante el entrenamiento y aplicar técnicas de reducción de dimensionalidad sobre los vectores de características, como PCA.

Por último, en la figura 4.7 se muestra una visualización del *embedding* generado por el método, donde se aprecia que tanto las fotografías como los dibujos se proyectan a un espacio en común y se agrupan en *clusters*, donde cada *cluster* agrupa imágenes visualmente similares, y *clusters* cercanos son similares entre sí; para ejemplificar, el *cluster* de girasoles, con el *cluster* de lunas y de ruedas de la fortuna son cercanos entre sí, ya que todas estas imágenes tienen forma circular.

Análisis general de SBIR

Se evaluaron tres métodos de recuperación de imágenes basada en sketch, correspondientes a **DeepSBIR**, **SBIR mediante redes siamesas** y **SBIR con entrenamiento en múltiples etapas**. Algunos análisis generales de estos métodos son:

- El entrenamiento usando *cross-entropy loss* presenta un mayor desempeño que el entrenamiento utilizando *contrastive loss*; así, para la arquitectura *AlexNet* y el mismo dataset de entrenamiento (Flickr 25k) el método *Deep SBIR* (que utiliza *cross entropy loss*) presenta un mAP de 0.172 en Flickr15k y de 0.243 en el dataset de Saavedra, mientras que el método con redes siamesas (que utiliza *contrastive loss*) alcanza un mAP de 0.137 y 0.172 respectivamente. Se concluye que esto ocurre porque el *cross entropy loss* posee una mayor información global de las clases que el *contrastive loss*, debido a que el *contrastive loss* solo nos permite tomar la información de un *batch* al entrenar, y por lo tanto una menor cantidad de información respecto a las clases; mientras que al utilizar *cross entropy loss* tenemos información de todas las clases presentes en el entrenamiento.
- Distintas arquitecturas de redes convolucionales pueden mejorar mucho los resultados, así, para el mismo método *Deep SBIR*, cuando se utiliza una red más actual (como *SE ResNext 50*) se alcanzan resultados del doble de desempeño respecto a *AlexNet*.
- Al procesar las fotografías extrayendo bordes antes de ingresarlas a una red convolucional se pierde mucha información (de color y de texturas por ejemplo), por lo que los métodos disminuyen mucho su desempeño respecto a ingresar directamente las imáge-

nes. Así, en el método de entrenamiento en múltiples etapas, los resultados son muy superiores en comparación a los otros métodos. Esto se puede interpretar como que las redes convolucionales son mejores extrayendo características y procesando las imágenes respecto a métodos tradicionales como *Canny*, además, al utilizar *Canny* se pueden extraer bordes con mucho ruido, lo que dificulta la tarea de la red convolucional. Se evidencia así la ventaja del método de *entrenamiento en múltiples etapas* respecto a los otros dos, debido a que se ahorra el tiempo de extracción de bordes y además presenta mejores resultados.

- La dimensionalidad del espacio de características es relevante en términos de la calidad de los sistemas de recuperación de imágenes, así, en general vectores de características de mayor dimensión funcionan mejor al poder contener una mayor cantidad de información, sin embargo esto tiene un costo asociado de memoria y de velocidad de búsqueda. Para afrontar estos problemas se sugiere experimentar con técnicas de reducción de dimensionalidad como PCA, lo que nos permitiría disminuir la memoria utilizada por los vectores de características y disminuir el tiempo de búsqueda, o técnicas de búsqueda aproximada para la reducción del tiempo de búsqueda.

4.2.2. Sistema de recuperación de imágenes basada en dibujos con color (CSBIR)

CSBIR con histogramas de color

El método de CSBIR con histogramas de color representa la forma clásica de abordar el problema de recuperación de imágenes considerando atributos de forma y color, en que por un lado se extraen características de forma y por el otro lado se extraen características de color.

Lo primero que es necesario recalcar respecto a este método corresponde a la elección del parámetro γ, que representa el trade off entre características de forma y de color. En la figura 4.9 se presenta el *mean reciprocal rank* y el *mean average precision* del sistema para distintos valores del parámetro γ. Se debe recordar que el *mean reciprocal rank* mide en que posición se encuentra la imagen correcta respecto a todas las imágenes recuperadas, y por lo tanto es un buen indicador de que tan bien el sistema es capaz de recuperar imágenes considerando forma y color; por otro lado el mAP mide que las imágenes de la misma clase se encuentren en las primeras posiciones, y por lo tanto es un buen indicador de que tan bien el sistema puede recuperar imágenes considerando la forma y clase de los objetos, pero no es un buen indicador para el color.

Considerando esto, el comportamiento del sistema es como se debería esperar, por ejemplo, al usar un valor de γ pequeño significa que se está dando una mayor ponderación a las características de forma, y por lo tanto el mAP es alto, pero el MRR no tanto, debido a que en las primeras posiciones tienden a aparecer objetos de forma similar, pero que pueden tener color totalmente distinto. A medida se tiende a aumentar el valor de γ comienza a aumentar el MRR, ya que los objetos de color similar tienden a aparecer más adelante en las imágenes recuperadas, sin embargo llega un punto en que ambas métricas comienzan a disminuir ($\gamma > 0{,}6$) esto debido a que se comienza a considerar solo la información de color para recuperar las imágenes, la cual no es suficiente para describir bien a la imagen. Una

representación gráfica de esto se muestra en la figura 4.9, donde se aprecia como el valor $\gamma = 0{,}6$ recupera imágenes de forma y color similar. A partir de esta experimentación se establece que un valor adecuado de este parámetro es 0.6, ya que presenta el mayor MRR a costa de una disminución pequeña en el mAP.

Por último en la figura 4.10 se muestran diversos ejemplos de imágenes recuperadas con el valor anteriormente mencionado de γ, donde se ve que para algunas consultas como la fuente blanca o la polera rosada el método funciona muy bien, sin embargo para otras consultas como el auto o la botella el sistema no considera muy bien el color al recuperar las imágenes.

CSBIR con quadruplet networks

Este método plantea una estrategia de recuperación de imágenes basada en dibujos con color que solo utiliza redes convolucionales (sin necesidad de extracción de características de color de manera separada); esto plantea una ventaja, ya que la extracción de características se realiza de forma mucho más rápida y considera el sketch completo (con color), lo cual nos provee una cantidad de información mayor respecto a cuando se extraen características usando el sketch sin color y se le agregan las descripciones de color de manera separada.

En la tabla 4.5 se muestra el MRR y el mAP para la etapa 2 y 3 del entrenamiento. Se recalca que en la etapa 2 del entrenamiento el MRR y el mAP son mayores que el método con histogramas de color, y el valor del mAP es el más alto alcanzado. Esto ocurre debido a que la etapa 2 tiene por objetivo formar un buen *embedding* para las clases, así, el mAP alcanzado es alto; sin embargo esta etapa no considera información de color, por lo que no es útil como método de recuperación de imágenes basada en dibujos con color.

Por otro lado en la etapa 3 del entrenamiento el parámetro α produce un efecto similar en el mAP y el MRR que el parámetro γ del método con histogramas de color. Esto ocurre debido a que valores grandes de α hacen que imágenes de distinto color se alejen mucho, por lo que se tenderá a formar *clusters* del mismo color sin considerar información de las clases, así, el mAP comenzará a disminuir. Se tiene sin embargo, que en esta etapa el mAP disminuye respecto a la etapa 2, debido a que ahora el sistema debe ordenar las imágenes de acuerdo al color y a la forma. De acuerdo a los resultados se escoge $\alpha = 0{,}25$ como el mejor valor para este parámetro, ya que alcanza el mayor valor para el MRR, y un buen valor del mAP.

Al analizar las imágenes recuperadas en las figuras 4.11 y 4.12 es posible notar gráficamente como la etapa 2 del método es capaz de recuperar imágenes basada en la información de forma, sin embargo no considera el color al retornar las imágenes; mientras que la etapa 3 considera color y forma al momento de recuperar las imágenes, a costa de que tiende a recuperar imágenes de otras clases en algunas ocasiones.

Una buena visualización de como actúa la red se presenta en la figura 4.13, donde se aprecia que el sistema forma *clusters* basados en la forma de las imágenes, y dentro de cada *cluster* ordena las imágenes de acuerdo a su color, lo cual es el comportamiento esperado para el sistema. Otra de las grandes ventajas del sistema es que nos permite recuperar imágenes considerando color y forma usando como query una fotografía, así, el sistema se podría adaptar para realizar *content based image retrieval* considerando características de color.

Finalmente, en la figura 4.15 se analiza el *recall ratio* del método de histogramas de color y de la etapa 2 y 3 de *Quadruplet Networks*, donde se aprecia la superioridad de este último método, y en particular de la etapa 3. Además, de la figura se puede establecer que un buen número de imágenes retornadas corresponde a 50 imágenes; haciendo que en el 80 % de las consultas se retorne la imagen correcta entre las 50 imágenes retornadas.

Análisis general de CSBIR

Se evaluaron dos métodos para recuperación de imágenes basada en dibujos con color, el primero mediante el uso de histogramas de color y el segundo con *Quadruplet Networks*. Algunos análisis de estos métodos son:

- El método de CSBIR con histogramas de color presenta la ventaja de que es flexible, en el sentido de que se puede cambiar la ponderación entre color y forma de manera fácil y ajustable por el usuario. Además no requiere un dataset de dibujos con color para ser entrenado.
- El método de *Quadruplet Networks* presenta la ventaja de que produce un solo *embedding* que codifica información de forma y color; además la extracción de características se realiza solo haciendo un *forward* por la red convolucional y no requiere extraer histogramas ni otros descriptores de la imagen o dibujo, lo cual hace al método mucho más rápido.
- El método de *Quadruplet Networks* presenta mejores resultados en términos del MRR, ya que es capaz de extraer información detallada de color y forma en conjunto, mientras que el método de histogramas de color realiza la extracción por separado, lo cual no tiene un desempeño óptimo.
- El método de *Quadruplet Networks* presenta la ventaja de que permite además realizar recuperación de imágenes a través de otras imágenes, extrayendo a través de la misma red información de color y forma, lo que resulta ser una novedad (ya que no existen métodos que aborden la extracción de características de color a través de redes convolucionales).
- Si bien el método de *Quadruplet Networks* marca una mejora sustancial en el problema de recuperación de imágenes basada en dibujos con color, existen muchas mejoras que se les puede hacer, como el uso de un dataset con mayor cantidad de imágenes y clases mejor definidas, o un mejor método para sintetizar dibujos a partir de las fotografías.

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

A lo largo de este trabajo se realizó un estudio exhaustivo de métodos de recuperación de imágenes basadas en dibujos sin y con color usando redes convolucionales; los cuales son cada vez más atractivos por sus aplicaciones en *e-commerce*, debido a que representan una herramienta de búsqueda muy sencilla y a la vez poderosa.

Dentro del estudio de los métodos de *Sketch Based Image Retrieval* (SBIR) sin color se exploraron tres alternativas, enumeradas a continuación:

- **DeepSBIR:** consistente en entrenar una red convolucional para clasificación de dibujos, para luego extraer características de dibujos y de fotografías, convirtiendo las fotografías a representaciones más cercanas a dibujos a través de la extracción de bordes, y luego extrayendo características a través de la red convolucional.
- **Siamese SBIR:** realiza el mismo proceso anterior, pero entrena la red convolucional a través de la función de pérdida *contrastive loss*.
- **MultiStage Regression SBIR:** utiliza dos redes convolucionales para extraer una representación común entre dibujos y fotografías, entrenando las redes en múltiples etapas con una dificultad creciente.

De la realización de estos resultados se concluye que la mejor metodología entre las estudiadas corresponde a la tercera, ya que nos permite realizar un preprocesamiento mínimo de las fotografías, al poder ingresarlas directamente a una red convolucional (sin extraer bordes primero), y además logra un desempeño mucho mayor, lo cual se asocia a su estrategia de entrenamiento, a su forma de compartir pesos y a que al no realizar el paso previo de extracción de bordes cuenta con una mayor cantidad de información para poder realizar la extracción de características.

Respecto a los métodos de *Color Sketch Based Image Retrieval* (CSBIR) se compararon dos implementaciones, enumeradas a continuación:

- **CSBIR con histogramas de color:** consiste en extraer por separado características

de forma y de color de fotografías y dibujos, utilizando el método *MultiStage Regression SBIR* para la extracción de características de forma e histogramas de color para describir las características de color, finalmente, la distancia se calcula mediante una ponderación entre las distancias de las características de forma y color entre las consultas y las fotografías del dataset, que nos permite decidir si al momento de recuperar las imágenes nos importa más la forma o el color de los resultados, de acuerdo a nuestra consulta.

- **CSBIR con Quadruplet Networks**: es una manera novedosa de abordar el problema de CSBIR, al implementar una red convolucional capaz de extraer en conjunto información de forma y color de dibujos y fotos; teniendo la ventaja de que el procesamiento es mucho más rápido al ahorrar el tiempo de extracción de histogramas de color, y los resultados son mejores en métricas y visualmente, ya que las características codifican en conjunto forma y color.

Se concluye que el primero de los métodos presenta un desempeño que funciona bien respecto a algunas consultas, pero que no es la manera óptima de abordar el problema, debido a que al realizar una extracción por separado de características de forma y color es difícil encontrar una ponderación que entregue buenos resultados para el general de los casos, así, a veces los resultados entregados pueden ser buenos en forma y no en color, o en otros casos puede suceder lo contrario dependiendo de la ponderación utilizada. Por otro lado, se concluye que el segundo método funciona mejor al ser más rápido y presentar mejores resultados en término de métricas de desempeño y visualmente. Además, a la hora de implementar estos métodos es necesario seleccionar adecuadamente los parámetros, lo cual se puede realizar midiendo las métricas de desempeño del sistema con distintos valores de los mismos, esto sobre un conjunto de validación.

Respecto a los objetivos específicos planteados en este trabajo, se cumplieron satisfactoriamente cada uno de ellos:

- Se experimentaron distintas arquitecturas de redes convolucionales y se optó por utilizar la arquitectura SE-ResNext 50, debido a que presentaba un mejor desempeño frente a otras arquitecturas a la hora de realizar la recuperación de imágenes basada en dibujos.
- Se exploraron y evaluaron tres metodologías para la recuperación de imágenes basada en dibujos sin color, en particular, la metodología de entrenamiento en múltiples etapas presentó resultados superiores, alcanzando resultados del estado del arte.
- Se construyó un dataset para entrenamiento del sistema de recuperación de imágenes basada en dibujos con color, con cerca de 20.000 fotografías y sus respectivos dibujos, generados sintéticamente a partir de las fotografías. Además se construyó un dataset para la evaluación del sistema, con aproximadamente 46.000 imágenes y 200 dibujos con color.
- Se implementó un baseline para el problema de recuperación de imágenes basada en dibujos con color, consistente en la extracción por separado de características de forma y de color.
- Se definió una arquitectura de entrenamiento para el problema recién mencionado, la cual utiliza *quadruplet networks*, un entrenamiento en múltiples etapas y una nueva función de pérdida.
- Se evaluó el baseline y la nueva metodología propuesta, obteniendo mejores resultados

con la metodología propuesta.

Finalmente, se concluye que se pudieron implementar de manera exitosa los sistemas de recuperación de imágenes basados en dibujos con y sin color; dando cuenta que el método de entrenamiento y la modalidad de entrada de las imágenes a la red son muy relevante a la hora de implementar este tipo de sistemas. También es necesario destacar que este tipo de sistemas tiene potenciales aplicaciones en plataformas de *e-commerce* para realizar búsqueda por similitud y ya se encuentra implementado en algunas tiendas como *Casa Ideas* en Chile o *Sodimac Corona* y *Pepeganga* en Colombia, provisto por la empresa *Impresee*.

5.2. Trabajo futuro

El trabajo realizado en SBIR y color SBIR representan un importante avance; presentando resultados que replican el estado del arte en el problema de SBIR y un nuevo punto de comparación para CSBIR, que se espera se pueda mejorar a medida se cuente con una mayor cantidad de datos y mejor calidad de los mismos, y con nuevas técnicas de entrenamiento o arquitecturas de redes. Con esto en mente se mencionan los siguientes aspectos dentro del trabajo futuro:

- **Entrenar el modelo de *Quadruplet Networks* con un dataset más grande y de mayor calidad**: para esto se propone que el modelo puede ser entrenado con la utilización de un **dataset de segmentación**, que tenga imágenes con fondo, así, para generar objetos de diferentes colores se puede modificar el espacio HSV usando la máscara de segmentación para no cambiar el color del fondo. Además los dibujos con color pueden ser generados utilizando la misma máscara, de modo de hacer un dibujo que no incluya fondo.
- **Idear y entrenar un modelo de SBIR que incluya información semántica de las clases**: la idea es poder entrenar un modelo de recuperación de imágenes basado en dibujos que sea capaz de incorporar información semántica de los objetos, a modo de ejemplo, si la consulta corresponde a una bicicleta, que se entreguen en primer lugar bicicletas, y luego otras categorías similares como pueden ser un scooter, una motocicleta scooter, una patineta, etc.
- **Extender el modelo de CSBIR con *Quadruplet Networks* para color *content based image retrieval*:** es decir, realizar un sistema de búsqueda a través de imágenes que sea capaz de recuperar imágenes considerando tanto la forma como el color de los objetos.
- **Probar los modelos propuestos con arquitecturas más recientes:** constantemente surgen nuevas publicaciones científicas con avances en arquitecturas de redes convolucionales, de modo de lograr un mayor accuracy o menores tiempos de evaluación. Se propone entrenar los modelos de múltiples etapas para SBIR y de *Quadruplet Networks* para CSBIR con arquitecturas del estado del arte, de modo de mejorar el desempeño de los sistemas.

Bibliografía

[1] Ali Barakbah and Yasushi Kiyoki. 3d-color vector quantization for image retrieval systems. 09 2008.

[2] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. YOLACT: real-time instance segmentation. *CoRR*, abs/1904.02689, 2019.

[3] Tu Bui, Leonardo Ribeiro, Moacir Ponti, and John Collomosse. Sketching out the details: Sketch-based image retrieval using convolutional neural networks with multi-stage regression. *Computers Graphics*, 71:77 – 87, 2018.

[4] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, Nov 1986.

[5] Nick Craswell. *Mean Reciprocal Rank*, pages 1703–1703. Springer US, Boston, MA, 2009.

[6] Mathias Eitz, James Hays, and Marc Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012.

[7] Mathias Eitz, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. Sketch-based image retrieval: benchmark and bag-of-features descriptors. *IEEE Transactions on Visualization and Computer Graphics*, 2010.

[8] Eduardo S. L. Gastal and Manuel M. Oliveira. Domain transform for edge-aware image and video processing. *ACM TOG*, 30(4):69:1–69:12, 2011. Proceedings of SIGGRAPH 2011.

[9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[11] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2017.

[12] Rui Hu and John Collomosse. A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *Computer Vision and Image Understanding*, 117:790–806, 07 2013.

[13] Raimi Karim. Illustrated: 10 cnn architectures, 2019.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012.

[15] Yann Lecun, Leon Bottou, Y. Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324, 12 1998.

[16] J. J. Lim, C. L. Zitnick, and P. Dollár. Sketch tokens: A learned mid-level representation for contour and object detection. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3158–3165, June 2013.

[17] Ayuninda Dwi Nugrowati, Ali Ridho Barakbah, Nana Ramadijanti, and Yuliana Setiowati. Batik image search system with extracted combination of color and shape features. 2014.

[18] Mohamed Ouhda, Khalid Elasnaoui, Mohammed Ouanan, and B. Aksasse. *Content-Based Image Retrieval Using Convolutional Neural Networks*, pages 463–476. 01 2019.

[19] Y. Qi, Y. Song, H. Zhang, and J. Liu. Sketch-based image retrieval via siamese convolutional neural network. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 2460–2464, Sep. 2016.

[20] N Reddy, Gundreddy Reddy, and Dr.M. Narayana. Color sketch based image retrieval. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 03:12179–12185, 09 2014.

[21] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[22] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[23] J. M. Saavedra. Sketch based image retrieval using a soft computation of the histogram of edge local orientations (s-helo). In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 2998–3002, Oct 2014.

[24] Jose Saavedra and Benjamin Bustos. An improved histogram of edge local orientations for sketch-based image retrieval. pages 432–441, 09 2010.

[25] Jose M. Saavedra and Juan Manuel Barrios. Sketch based image retrieval using learned keyshapes (lks). In Mark W. Jones Xianghua Xie and Gary K. L. Tam, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 164.1–164.11. BMVA Press, September 2015.

[26] Jose M. Saavedra and Benjamin Bustos. An improved histogram of edge local orientations for sketch-based image retrieval. In *Pattern Recognition*, pages 432–441, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[27] Jose M. Saavedra and Benjamin Bustos. Sketch-based image retrieval using keyshapes. *Multimedia Tools and Applications*, 73(3):2033–2062, Dec 2014.

[28] Jose M. Saavedra and Camila Álvarez. Deepsbir: Sketch based image retrieval using deep features. 2016.

[29] Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. The sketchy database: Learning to retrieve badly drawn bunnies. *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 2016.

[30] Rishab Sharma and Anirudha Vishvakarma. Retrieving similar e-commerce images using deep learning. *CoRR*, abs/1901.03546, 2019.

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[32] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning, 2016.

[33] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014.

[34] Ren Jie Tan. Breaking down mean average precision (map), 2019.

[35] Yu Xia, Shuangbu Wang, Yanran Li, Lihua You, Xiaosong Yang, and Jian Jun Zhang. Fine-grained color sketch-based image retrieval. In Marina Gavrilova, Jian Chang, Nadia Magnenat Thalmann, Eckhard Hitzer, and Hiroshi Ishikawa, editors, *Advances in Computer Graphics*, pages 424–430, Cham, 2019. Springer International Publishing.

[36] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks, 2016.

[37] Peng Xu, Yongye Huang, Tongtong Yuan, Kaiyue Pang, Yi-Zhe Song, Tao Xiang, Timothy M. Hospedales, Zhanyu Ma, and Jun Guo. Sketchmate: Deep hashing for million-scale human sketch retrieval. *CoRR*, abs/1804.01401, 2018.