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

AUTO TAGGING EN CATÁLOGOS VIRTUALES DE VESTUARIO

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELÉCTRICO

DANIEL EDUARDO GUZMÁN PRADEL

PROFESOR GUÍA:
JUAN MANUEL BARRIOS NÚÑEZ

MIEMBROS DE LA COMISIÓN:
CAMILA ANDREA ÁLVAREZ INOSTROZA
ANDRÉS EDUARDO CABA RUTTE

SANTIAGO DE CHILE
2020

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERO CIVIL ELÉCTRICO
POR: DANIEL EDUARDO GUZMÁN PRADEL
FECHA: 2020
PROF. GUÍA: JUAN MANUEL BARRIOS NÚÑEZ

AUTO TAGGING EN CATÁLOGOS VIRTUALES DE VESTUARIO

A lo largo de los años, la industria de la moda ha mostrado un crecimiento constante que se ha visto acelerado por la venta de artículos por Internet, siendo hoy la categoría con más clientes del *ecommerce* en Chile.

Las tiendas exhiben sus productos en Internet usando los catálogos virtuales, que son herramientas que organizan los productos y permiten buscarlos y filtrarlos mediante texto, o por categorías y filtros. Una exploración a los catálogos del *ecommerce* nacional, deja en evidencia una deficiencia en la cantidad de filtros orientados en características visuales de las prendas, lo que impide una exploración eficiente para encontrar los productos deseados. Esto es especialmente importante en la industria de la moda, donde los artículos se basan fuertemente en estas características.

Los algoritmos de visión computacional desarrollados en los últimos años han mostrado ser capaces de automatizar una multitud de tareas. En este trabajo, se propone el uso de redes neuronales convolucionales, que son capaces de procesar imágenes, para generar etiquetas o *tags*, que representen características visuales de artículos de vestuario para realizar la búsqueda de estos.

Se estudia el desempeño de dos arquitecturas de redes neuronales convolucionales: ResNet50 e Inception-v3 en problemas de clasificación multi-etiqueta y clasificación binaria independiente de clases. Las clases utilizadas en este trabajo se definen en razón de aquellas disponibles en las bases de datos públicas *DeepFashion* y *DeepFashion* 2. Además, se definen 9 clases de color a identificar en las prendas, para lo cual se construye un sistema de clasificación de color basado en el uso de una red DeepLab-v3 de segmentación y el algoritmo de *clustering k-means*.

Se mide el desempeño de los modelos según los resultados de la búsqueda de artículos con los *tags* asignados por ellos. Para esto, se utiliza un *dataset* privado adicional con fotografías de catálogo de tres tiendas de vestuario, el cual es etiquetado para conducir un proceso de búsqueda por texto con cada etiqueta para recuperar sus imágenes, con lo que se valida el desempeño del trabajo en un ambiente real. El mejor desempeño en la recuperación de artículos se obtiene con las etiquetas generadas por ResNet50 multi-etiqueta, alcanzando un *mean average precision* (*mAP*) de 47 % en la recuperación de 15 artículos de cada clase. En particular, el mejor desempeño se obtiene para categorías de prendas y texturas, obteniendo un 55 % y 70 % de mAP con este mismo modelo.

Para mi mamá, quien con su amor hizo posible todo lo que soy.

Agradecimientos

Quiero agradecer en primer lugar a la persona más importante en todo este proceso, mi mamá, Claudia. Gracias por siempre preocuparse por mí, por aconsejarme y por permitirme crecer conociendo el amor durante toda mi vida. Gracias, gracias y gracias, nunca tendré palabras para agradecerle todo lo que ha hecho por mí. La amo mamá.

A mi familia, por su compañía y apoyo, especialmente este año tan complicado. Gracias Fabián y tía Paty por su ayuda y demostrar su apoyo incondicional. Gracias especiales a mi abuela, la Marta, por darnos de tu fortaleza para salir adelante, por tu compañía de todas las tardes y tu preocupación por mi. Gracias a ti Thomás, por crecer conmigo y ser el hermano que no tuve, con quien sé que puedo contar. Gracias a la Mami, mi bis-abuela, quien nos ayudó a crecer a todos y que hasta el día de hoy nos acompaña desde el cielo.

Gracias Lolita, por llegar a mi vida y alegrarla con todo lo que haces. Por acompañarme a madrugar todas esas noches de estudio y darme tranquilidad con tus besitos. Por acompañarme a dormir cada noche y ser mi regalona. Por darme todas esas risas con tus tonteras, tus mañas y todo lo que haces.

Gracias Valeria por todo el tiempo compartido. Gracias por darme tu amor y recibir el mío, escucharme y darme tus consejos, los que siempre recuerdo y guardo con amor.

A todos los compañeros y amigos que he tenido en mi vida, gracias por ser parte de ella. Gracias especiales a mi mejor amigo de la universidad, el Simón, por las tallas, los carretes y tu amistad.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

La industria de la moda es una de las más importantes y de mayor penetración a nivel mundial a través de todo tipo de mercado, como el *retail*, ferias de las pulgas, o mercados de redes sociales. A principios del año 2019, la industria de vestuario (excluyendo calzado) alcanzó un tamaño de 1.4 billones de dólares a nivel mundial y se proyecta que siga creciendo consistentemente en los próximos años.

Este mercado, al igual que la gran mayoría, ha crecido a través del comercio por Internet, tanto así, que el año 2019 las ventas *online* de este rubro significaron 545 miles de millones de dólares en ventas a nivel global, y se espera que crezca un 30 % hacia el año 2022, alcanzando los 713 miles de millones de dólares[1].

En Chile, al tercer trimestre del año 2019, el comercio electrónico de vestuario fue la categoría con el mayor porcentaje de clientes del *ecommerce* nacional [11], y a pesar de que sus ventas disminuyeron el año 2020 producto de la pandemia causada por el coronavirus, las visitas a portales *online* de moda aumentaron a lo largo del año [12], mostrando que es relevante poder ofrecer la posibilidad de vitrineo *online* a los potenciales clientes.

Para exhibir sus productos en Internet, las tiendas hacen uso de los catálogos virtuales. Estas son herramientas que permiten encontrar artículos mediante búsqueda por texto, y según determinados filtros de categorías y otros detalles. En la Figura 1.1, se ve un ejemplo de un catálogo donde se aprecia una barra de búsqueda por texto y las categorías que ofrece.

Un problema de los catálogos de tiendas de vestuario del *ecommerce* nacional, es que ofrecen filtros de prendas según criterios como marcas, precio, valoraciones de usuarios, y otros; dejando de lado características visuales que podrían ser más relevantes para encontrar artículos de vestuario, como se observa en la Figura 1.2. En un contexto económico, esta deficiencia de los catálogos se traduce en posibles oportunidades de venta perdidas para los negocios, y en una mayor inversión de tiempo de los clientes para encontrar un producto, debido a que se requiere un mayor esfuerzo para encontrar lo que se quiere. Esto empeora el problema, puesto que el motivo principal por el que los clientes deciden comprar *online* en Chile, es por el ahorro en tiempo que esta modalidad supone [11].

[1] `https://www.datafeedwatch.com/blog/ecommerce-and-the-fashion-industry`

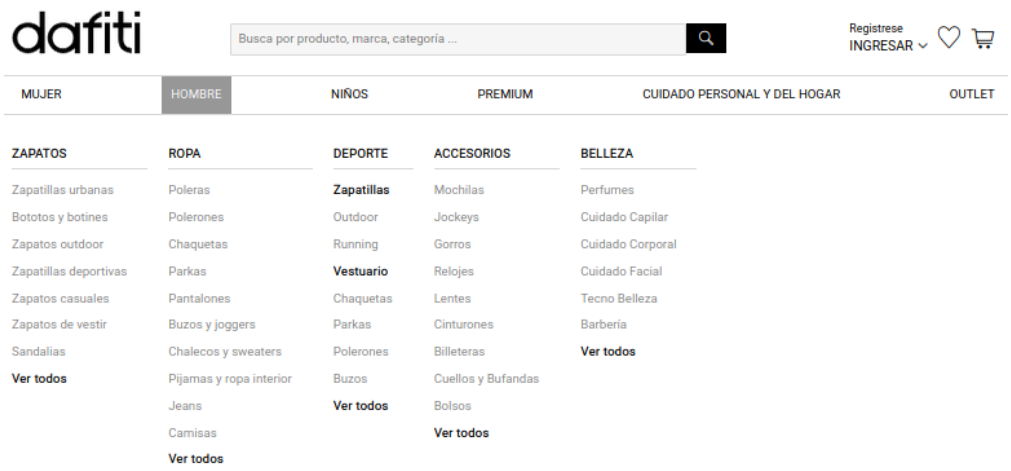


Figura 1.1: Ejemplo de las herramientas de búsquedas que ofrece un catálogo virtual.

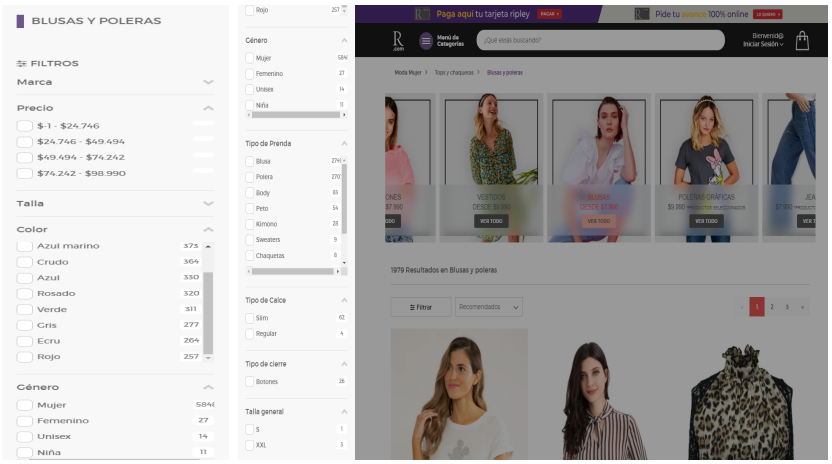


Figura 1.2: Muestra de los filtros ofrecidos para el filtrado de productos de vestuario según características específicas de la categoría blusas y poleras del portal de Ripley.

Los *tags* corresponden a palabras o frases cortas, que resumen el contenido semántico de las imágenes. Con ellos se puede realizar búsqueda de artículos por texto, midiendo la relevancia de algún término con los *tags* asociados a una imagen. Su principal ventaja, es que pueden ser obtenidos de muchas fuentes, como de anotaciones humanas, mediante técnicas de procesamiento de lenguaje, clasificación de imágenes, entre otros. Su contra, es que son costosos de obtener, sobre todo si se considera la gran cantidad de imágenes que componen los catálogos virtuales.

El desarrollo de métodos de procesamiento de imágenes, especialmente con el uso de redes neuronales convolucionales, desde hace varios años han mostrado desempeños cada vez mejores en distintos tipos de tareas, insertándose como una solución factible para la automatización de muchas ellas, mostrando en determinados problemas desempeños muy similares al de los humanos [4]. A pesar de esto, las aplicaciones con estos métodos han quedado detrás respecto a otras áreas, posiblemente por aspectos que complican esta tarea como la gran diversidad de prendas y características de ellas, su deformabilidad en las imágenes, y

a la dificultad de obtener datos bien etiquetados que permitan el entrenamiento de estos algoritmos.

La industria de la moda puede beneficiarse fuertemente de la utilización de *tags* para organizar catálogos con muchas categorías o con prendas con una alta cantidad de detalles. Por ejemplo, ya en etapa del inicio de desescalada de la pandemia causada por el Covid-19 en España, uno de los paises más afectados en el mundo, un estudio [18] muestra que las ventas *online* de moda y calzado representan la categoría con mayor crecimiento al empezar a retomar las actividades normales, con un aumento del 49,5 %. Esto refuerza la importancia de tener herramientas apropiadas para mostrarle productos a los clientes dado el gran uso que se les puede dar.

El presente trabajo tiene como objetivo implementar y evaluar métodos de clasificación de imágenes para asignarles etiquetas de manera automática (*auto-tagging*). Con estos *tags* se lleva a cabo un proceso de búsqueda de artículos por texto para filtrar prendas según distintas características visuales. Con el fin de evaluar el trabajo en un ambiente real, se lleva a cabo este proceso con imágenes de tres catálogos virtuales de tiendas reales.

1.1. Objetivos

1.1.1. Objetivo general

Desarrollar un sistema de etiquetado automático de imágenes que permita la implementación de filtros en catálogos virtuales, con el fin de mejorar las herramientas de búsqueda de artículos de vestuario que se usan actualmente en portales de *ecommerce* de este rubro.

1.1.2. Objetivos específicos

i Implementar y evaluar distintos métodos que permitan la asignación automática de *tags* a imágenes de vestuario.

ii Evaluar el sistema desarrollado mediante el filtrado de imágenes de catálogo de tiendas reales de vestuario, con el fin de darle validez al trabajo en un contexto real.

iii Determinar los mejores casos de uso del sistema desarrollado, en base a la evaluación realizada.

Capítulo 2

Marco teórico

En el presente capítulo se presenta una serie de conceptos relevantes a este trabajo. Estos involucran métodos de clasificación, espacios de color y métricas de evaluación utilizadas. Se muestran en detalle en las secciones a continuación.

2.1. Neurona artificial

La neurona artificial es la unidad base de las redes neuronales artificiales (ANN por sus siglas en inglés). Como unidades de una red permiten a las ANN construir una serie de hiperplanos que separen un espacio multi-dimensional, con el fin de ajustarse a una función objetivo.

Los parámetros de una neurona artificial son tres: un vector de pesos $\vec{W}$ (de las mismas dimensiones que el vector de entrada esperado $\vec{X}$), un sesgo o *bias* b y una función de activación (o de transferencia) ϕ. Una representación visual de una neurona con sus parámetros se muestra en la Figura 2.1.

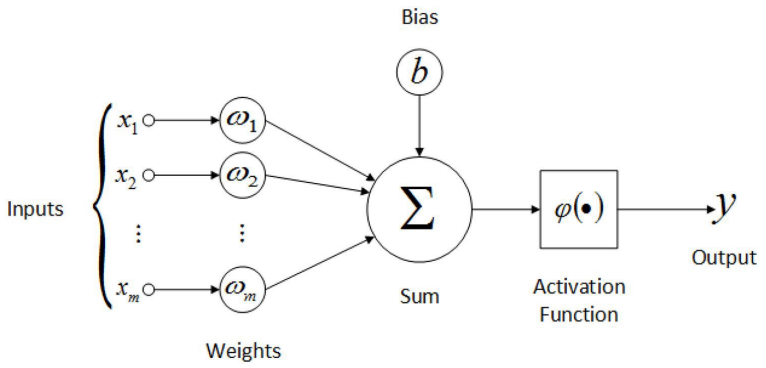


Figura 2.1: Modelo de una neurona artificial.

Matemáticamente, la salida $\vec{y}$ de una neurona es:

$$y = \phi(\langle \vec{X}, \vec{W} \rangle + b) = \phi(\sum_{\mathrm{i}}^{M} (w_{\mathrm{i}} x_{\mathrm{i}}) + b)$$

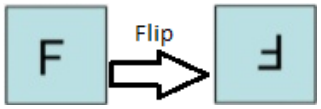
La función de transferencia le entrega la mayor parte de sus propiedades a la neurona y puede ser cualquier función matemática, sin embargo, se eligen generalmente funciones no lineales. Dentro de las más relevantes para este trabajo, se encuentran:

- Sigmoide: $sig(x) = \frac{1}{1+\mathrm{e}^{-x}}$
- Softmax: $softmax(X) = \frac{x_{\mathrm{i}}}{\sum_{\mathrm{i}=1}^{N} x_{\mathrm{i}}}$

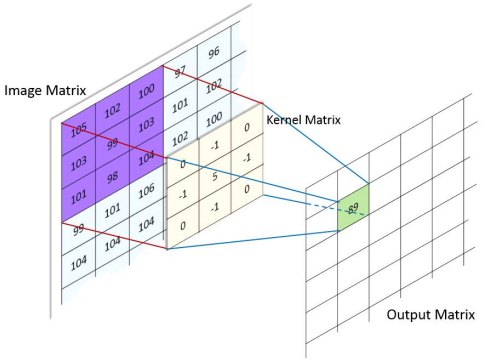
Ambas funciones son utilizadas a las salidas de las redes neuronales. Una característica importante de ellas, es que entregan un espacio de probabilidades válido de las clases, es decir, que su salida se puede interpretar como la probabilidad de encontrar cada una en la entrada. La diferencia entre ambas, es que la sigmoide calcula esta probabilidad independientemente para cada clase, mientras que la softmax asume la presencia exclusiva de las clases, restringiendo a que la suma de probabilidades de todas las clases deba sumar 1. Por este motivo, la sigmoide se usa cuando se desea encontrar varias clases a la vez (clasificación multi-etiqueta); y la softmax, cuando se desea encontrar la clase predominante (clasificación multi-clase).

2.2. Convolución

En procesamiento de imágenes una convolución es una operación realizada entre una matriz pequeña, típicamente de 1x1, 3x3, 5x5 ó 7x7, llamada *kernel* y una imagen, cuyo fin es filtrar características de la imagen como bordes, esquinas, colores, entre muchas otras. El resultado de esta operación disminuye el tamaño de la imagen (excepto para *kernels* de 1x1), logrando de esta manera resumir la información presente en la imagen.



(a) *Flip* de una matriz/imagen.



(b) Operación entre el kernel y una zona de la imagen.

Figura 2.2: Filtrado de una imagen con un *kernel* mediante convolución.

La operación de convolución se ilustra en la Figura 2.2, donde al kernel se le aplica una reflexión en el eje horizontal y vertical (*flip*) y luego se barre por toda la imagen aplicando un

producto punto con cada una de las submatrices del mismo tamaño del kernel de la imagen original, con lo que se va construyendo una imagen filtrada o mapa de características.

La forma en que el *kernel* es barrido por la imagen depende de tres parámetros: el tamaño del filtro, el *stride* y el *padding*, los cuales son explicados a continuación:

- Tamaño del filtro: Como su nombre indica, corresponde a las dimensiones del filtro, que es típicamente cuadrado. Ya que con la convolución se resume toda la zona operada a un único valor, según el tamaño del filtro, el mapa de características puede tener algunos pixeles menos que la entrada si no se aplica *padding*.
- Stride: Corresponde a cuantas unidades se desplaza el filtro entre la posición actual y la siguiente tanto en orientación vertical como horizontal. Al igual que el tamaño del filtro, mientras más grande es este valor, más pequeño es el resultado.
- Padding: Corresponde a la cantidad de pixeles que se añaden artificialmente a los bordes de la imagen para garantizar que el kernel no deje zonas de esta sin barrer en algún desplazamiento. La técnica más común de *padding* corresponde a *zero-padding*, que es rellenar con ceros. Contrario a los otros dos parámetros, aumentar el *padding* aumenta el tamaño del mapa de características.

2.3. Redes neuronales convolucionales

Las redes neuronales convolucionales (CNN) son capaces de procesar imágenes y realizar distintas tareas con ellas. Dentro de las tareas más relevantes para este trabajo se encuentran la clasificación y la segmentación. La clasificación se basa en indicar la presencia de clases predefinidas en una imagen, mientras que la segmentación además construye una imagen con máscaras, de las mismas dimensiones que la de entrada, donde se encuentran los objetos presentes de cada clase.

Las CNN llevan a cabo sus tareas realizando extracción de características visuales utilizando capas de *kernels* de distintos tamaños aplicados secuencialmente, denominadas capas convolucionales, como se muestra en la Figura 2.3. Estas capas se encargan de extraer las características más relevantes de las imágenes, para que las neuronas puedan discriminar entre los objetos presentes en ellas.

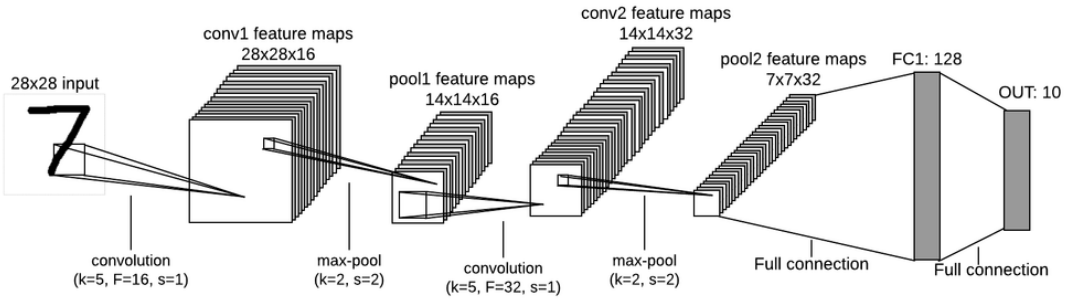


Figura 2.3: Estructura de una CNN.

Para realizar clasificación, los mapas de características son alimentados a capas de neuronas que se encargan de predecir la presencia de las clases a la salida de la red. En segmentación

en cambio, las CNN pueden terminar con una capa convolucional, donde el mapa de características entregado es interpretado como una máscara que indica la pertenencia de un pixel a una clase, para definir la zona que objetos de estas clases ocupan en una imagen.

Las capas convolucionales, al igual que las neuronas, adaptan sus parámetros para maximizar la diferenciación entre clases, en este caso, buscando extraer las características que resulten más distinguibles entre una y otra. Para esto, se lleva a cabo el mismo proceso de ajuste de parámetros que con las neuronas, asignándole un error a cada peso de cada capa en la etapa de entrenamiento, y ajustando su valor según esto.

Los parámetros de una capa convolucional que definen la manera en que se aplica la convolución, son:

- Tamaño del filtro: Define el tamaño del filtro usado en la capa. Típicamente se usan filtros cuadrados de tamaño 1, 3, 5 y más raramente de 7, 9 y 11.
- Número de filtros: Es el número de filtros usado en la capa. La salida de la capa, denominada mapa de características (*feature map*), tiene tantas dimensiones como número de filtros emplee. Así, una capa podría convertir una imagen de tres canales, a una imagen de 20 canales empleando 20 filtros distintos de por ejemplo 5x5x3, donde cada una tiene distinto tipo de información.
- Stride: Es el mismo parámetro que aquel visto en la sección anterior para un kernel, y único para todos los filtros aplicados en una misma capa.

Las técnicas de submuestreo, o *pooling*, disminuyen la dimensionalidad de las salidas de las capas convolucionales en adición a la reducción de dimensionalidad que ya ocurre al aplicar los *kernels*. Estas pueden resumir la información de distintas maneras, las más usadas son *max pooling* y *average pooling*, las cuales toman el máximo y el promedio de valores de una zona de una imagen respectivamente. El *pooling*, al igual que la convolución sobre una imagen, se aplica barriendo zonas de ella. Los parámetros de este son el tamaño de la zona y el *stride*.

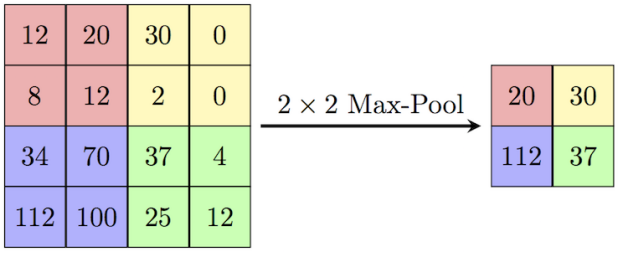


Figura 2.4: Ejemplo de *maxpooling* con tamaño de la región = 2 y *stride* = 2.

Luego de cada capa convolucional, se aplica una función de activación no lineal a la salida, ya que sin la existencia de esta, toda la red se podría reducir a un modelo lineal mucho más simple, perdiendo el propósito de la existencia de cada capa.

2.4. Arquitecturas CNN

Existen distintas arquitecturas CNN que se diferencian entre sí por la manera en que conectan sus capas o emplean sus filtros. Varias arquitecturas CNN famosas [10, 19, 6, 20]

fueron desarrolladas para competir en el concurso de clasificación de imágenes `ILSVRC` de ImageNet [16], y dado su buen desempeño fueron generalizadas para utilizarse en otros tipos de problemas, como por ejemplo segmentación [3, 14]. En las subsecciones que siguen se detallan las redes ResNet50 e Inception v3 para clasificación, y DeepLab v3 para segmentación semántica de imágenes.

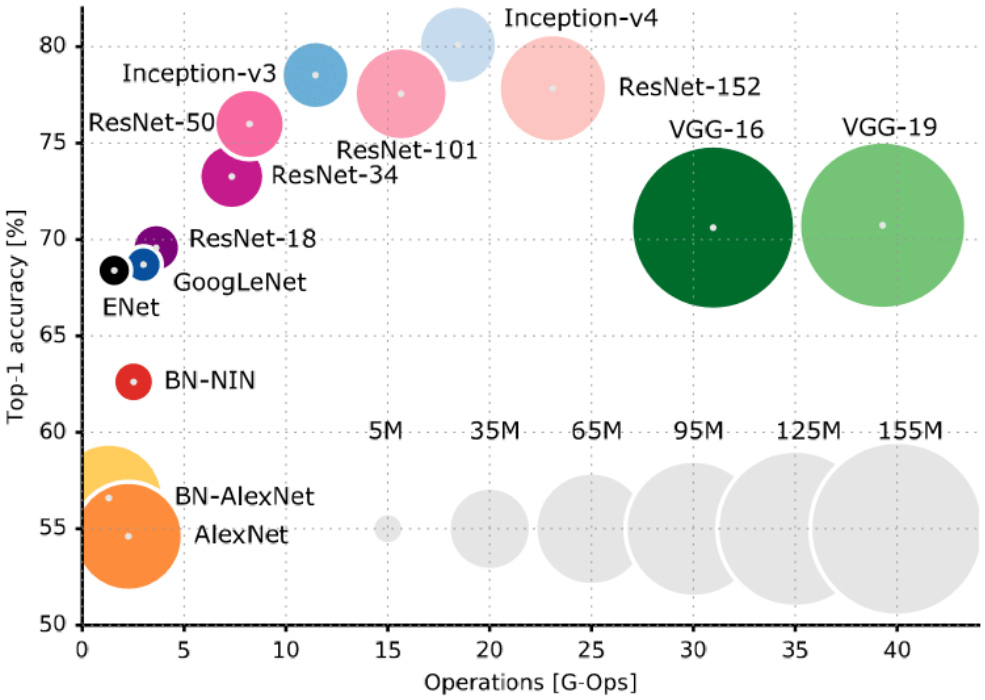


Figura 2.5: Cuadro comparativo de varias redes convolucionales populares. En el eje y, el *accuracy* de clasificación en el dataset ImageNet, mientras en el eje x, la cantidad de operaciones requeridas para procesar una imagen [9].

2.4.1. ResNet50

Las ResNet (*Residual Networks*) fueron presentadas en el `ILSVRC 2015`, logrando alcanzar un mejor desempeño en este *benchmark* que el criterio humano, alcanzando un 3,6 % de error de clasificación en ImageNet [6]. Estas arquitecturas se caracterizan por la conexión de sus capas, las llamadas *skip connections*, que constan en una conexión entre la salida de un bloque de capas y la entrada del mismo, como se muestra en la Figura 2.6. Este tipo de conexión le permite a la red disminuir el problema del *banishing gradient*, mejorando su rendimiento cuando se incorporan más capas, al contrario de como suele ocurrir con otras arquitecturas.

De esta manera, la salida de los bloques son la función de transferencia más la función identidad de la entrada. El efecto de este diseño en el entrenamiento, es que logra incorporar una componente dependiente únicamente de la función de costo evaluada en la entrada, lo que mejora la actualización de parámetros [6]. Existen varios modelos de ResNet, los que se diferencian por el número de capas que incorporan en su arquitectura. Como su nombre indica, la ResNet50 está compuesta por 50 capas convolucionales.

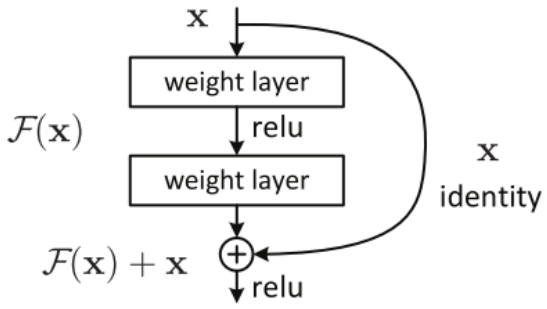


Figura 2.6: Conexión entre unas pocas capas denominada *bloque residual* [6].

2.4.2. Inception v3

Las GoogLeNet o más popularente conocidas como redes Inception, al igual que las otras arquitecturas mencionadas, fueron introducidas en el concurso `ILSVRC` para competir en la tarea de clasificación, ganando la versión del año 2014. Estas fueron diseñadas con la consideración de que muchas conexiones de una red CNN clásica son innecesarias, por lo que esta red se enfocó en disminuir la cantidad de parámetros necesarios. Para esto, reemplaza *kernels* de determinado tamaño por un mayor número de *kernels* más pequeños, con un método llamado de factorización de convoluciones [20], que permite obtener, con una menor cantidad de cálculos, mapas de características del mismo tamaño del que se obtendría con el uso de *kernels* sin factorizar. La red organiza su capas en módulos *Inception*, que consisten en conjuntos de *kernels* que se aplican secuencialmente y en paralelo sobre la entrada para ser concatenados a la salida, como se muestra en la Figura 2.7.

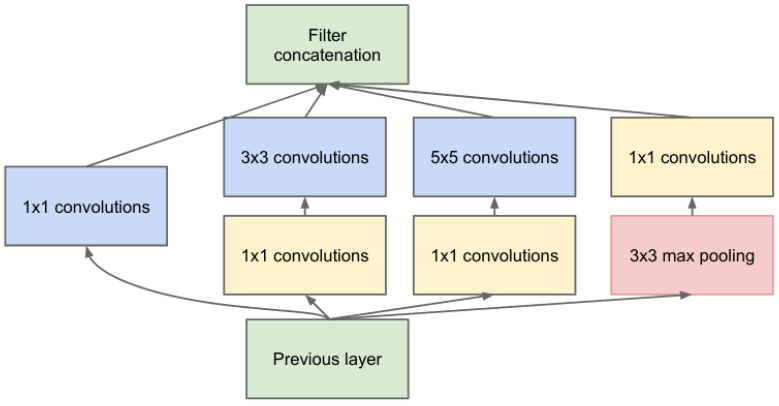


Figura 2.7: Módulo Inception.

Otra técnica importante incorporada en estas redes, es el *batch normalization*, que consiste en normalizar los valores de las entradas según los valores de conjuntos pequeños de ellas (*batches*). Esta técnica mejora la actualización de parámetros, puesto que le da estabilidad numérica al cálculo del gradiente.

2.4.3. DeepLab v3

DeepLab v3 es una arquitectura de segmentación semántica, lo que consiste en clasificar cada pixel de una imagen para definir las zonas que ocupan objetos de diferentes clases. Su característica principal es que hace uso de *atrous convolution*, el cual es un tipo de convolución con *kernels* dilatados que operan ignorando algunos pixeles como se muestra en la Figura 2.8. Esta convolución se usa en reemplazo a las operaciones de *pooling* conocidas, con lo que logra conservar mejor la información espacial entre pixeles a distintas escalas a diferencia de otros métodos que distorsionan la resolución de la imágen para lograr lo mismo [3].

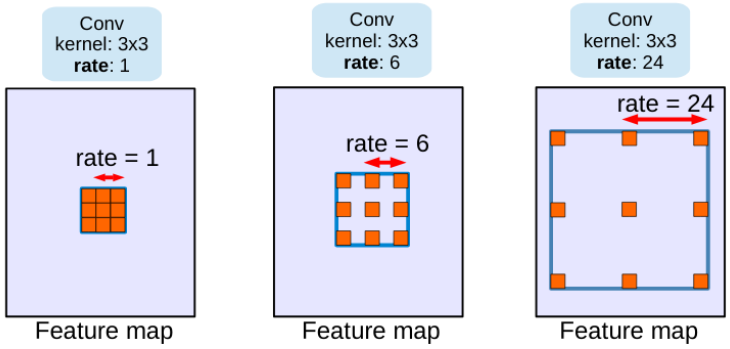


Figura 2.8: *Atrous convolution* con distintas razones de separación entre los pesos del *kernel* [3].

2.5. K-Means

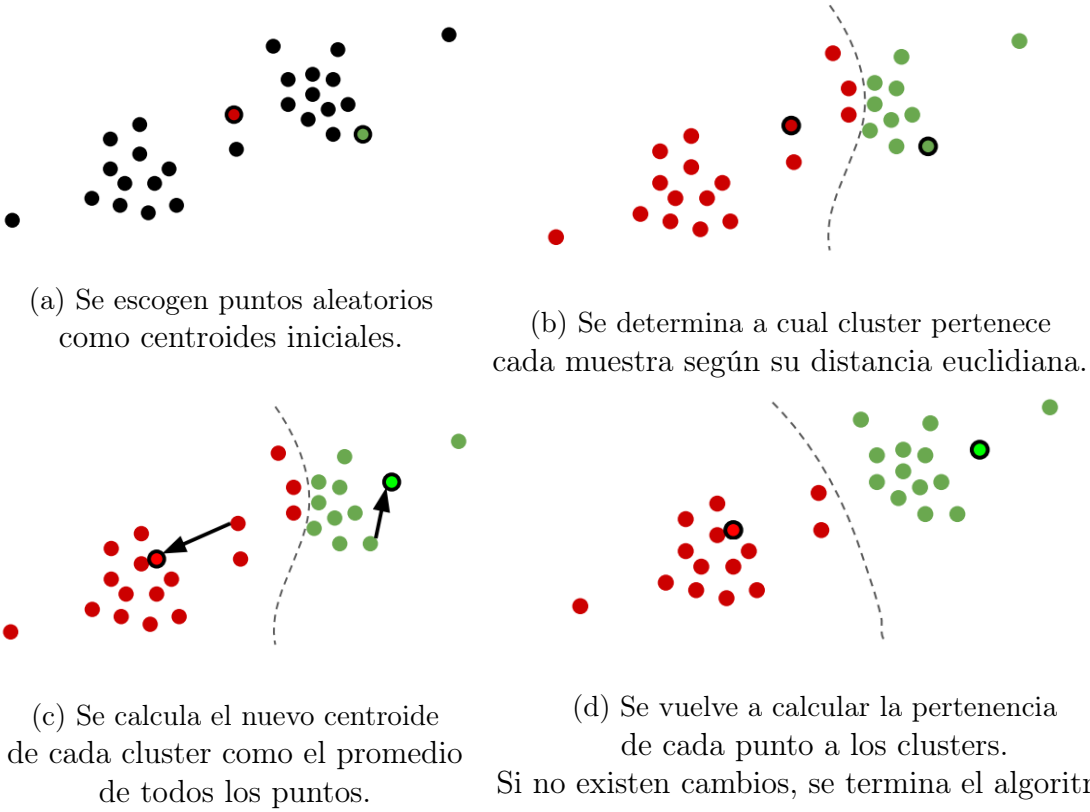
K-means es un algoritmo de *clustering* que se basa en agrupar puntos en el espacio en K conjuntos disjuntos entre ellos. Cada conjunto tiene un punto denominado centroide que se usa para determinar la pertenencia de otros puntos según la distancia a ellos. En la Figura 2.9 se detalla el algoritmo de convergencia para encontrar los clusters para dos *clusters* ($K = 2$).

Matemáticamente, K-means minimiza la función:

$$J(U, W) = \sum_{\mathrm{i}=1}^{N} \sum_{j=1}^{k} u_{\mathrm{i}j} ||x_{\mathrm{i}} - w_j||^2$$

Donde x_{i} es el punto i, w_j el j-ésimo centroide y $u_{\mathrm{i}j}$ una indicatriz que vale 1 si el punto j pertenece al cluster k y 0 si no. Así, plantea minimizar la suma de las distancias entre todos los puntos con los respectivos centroides de sus clases, para ajustar los conjuntos.

Las ventajas de este método es que es fácil de implementar y de relativamente bajo costo computacional. Las desventajas son que, es sensible a *outliers*, que tiene una componente aleatoria cuando se eligen los centroides iniciales, y que tiende a producir clusters del mismo tamaño, por lo que puede funcionar mal con datos desbalanceados.



(a) Se escogen puntos aleatorios como centroides iniciales.

(b) Se determina a cual cluster pertenece cada muestra según su distancia euclidiana.

(c) Se calcula el nuevo centroide de cada cluster como el promedio de todos los puntos.

(d) Se vuelve a calcular la pertenencia de cada punto a los clusters. Si no existen cambios, se termina el algoritmo.

Figura 2.9: Pasos del algoritmo de *clustering* K-Means.

2.6. Espacios de color

Los espacios de color son representaciones de los colores en un espacio, típicamente de tres dimensiones, llamados canales. Estos representan los colores con distintas componentes visuales. La elección de estos canales en los espacios de colores les brindan distintas propiedades. En este trabajo se detallan tres espacios de colores: RGB, HSV y CIELAB (o L*a*b*).

2.6.1. RGB

El espacio RGB es un espacio de colores aditivo, lo que significa que compone los colores según la suma de una base de colores, tal como lo hace la luz. Este descompone los colores en los canales rojo (R), verde (G) y azul (B). Es un espacio de colores muy común, dado que se usa para representar colores con fuentes lumínicas. El uso más familiar de RGB es en las pantallas LED o LCD. En la Figura 2.10 se puede observar como funciona la composición de colores de manera aditiva con luces de los tres colores que componen RGB.

2.6.2. CIELAB

El espacio CIELAB (o L*a*b*) es un espacio de colores diseñado para considerar los colores de manera similar a la que los humanos los percibimos. Para esto, descompone los colores en tres componentes: L* para la luminosidad; a*, como eje de tonos verde-rojo; y b* como eje de tonos azul-amarillo. Su propiedad más relevante para este trabajo, es que

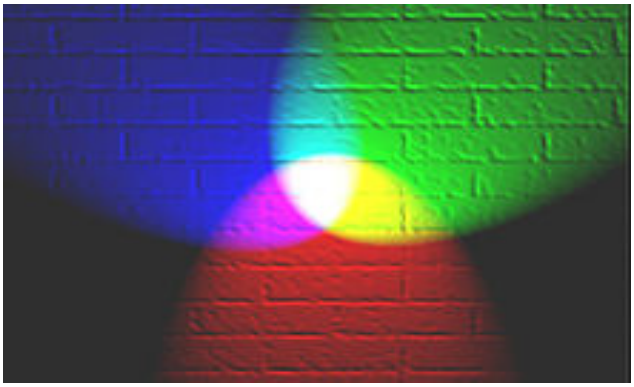


Figura 2.10: Composición aditiva de colores con luces de colores rojo, verde y azul, principio del espacio RGB.

conserva matemáticamente la distancia perceptiva entre los colores. Esto quiere decir que los colores que se perciben más diferentes por el humano tienen mayor distancia en este espacio, y viceversa.

2.6.3. HSV

El espacio HSV descompone los colores en los canales matiz (*hue*), saturación (*saturation*) y brillo (*value*). El matiz corresponde al tono del color (rojo o verde por ejemplo), la saturación a qué tan intenso es este tono, y el brillo a la intensidad de la luz con la que vemos el color. Su característica más importante es que logra separar en una variable la tonalidad del color, lo que le da la ventaja de ser más invariante a los cambios de iluminación en escenas. Gracias a esto también, este espacio se hace útil para definir los colores de manera similar a como lo hacen los humanos, nombrándolos por su tono. En la Figura 2.11 se observa la distribución de los colores en coordenadas cilíndricas del espacio HSV.

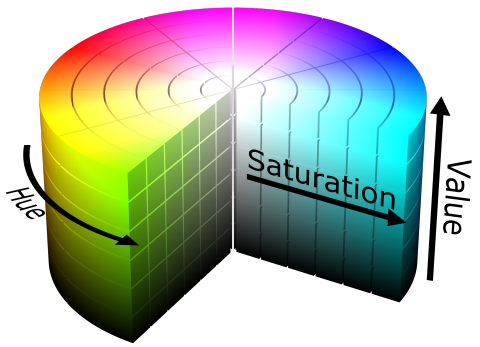


Figura 2.11: Representación cilíndrica del espacio HSV.

2.7. Métricas de evaluación

En la presente sección, se detallan las métricas usadas en el contexto de clasificación, segmentación y *retrieval* de imágenes, que son las tareas relevantes en este trabajo.

Varias de estas métricas se definen en función de otras cuatro variables, que representan el número de verdaderos positivos (TP), falsos positivos (FP), verdaderos negativos (TN) y falsos negativos (FN), los que toman distintos sentidos en cada contexto. Estas variables tienen el mismo sentido en clasificación y segmentación, puesto que podemos ver la segmentación como clasificación de pixeles, pero tienen un sentido distinto en *retrieval*. A continuación se muestra su definición en los contextos de clasificación y en *retrieval*.

- TP: Se predice correctamente la presencia de una clase, o se recupera un artículo relevante.
- FP: Se predice erróneamente la presencia de una clase, o se recupera un artículo irrelevante.
- TN: Se predice correctamente la ausencia de una clase, o no se recupera un artículo irrelevante.
- FN: Se predice erróneamente la ausencia de una clase, o no se recupera un artículo relevante.

2.7.1. *Precision*

En clasificación, corresponde a la razón de ejemplos efectivamente positivos sobre todos aquellos predichos como positivos por un modelo, indicando de una manera qué tan "limpias" son las predicciones. En *retrieval* indica la tasa de elementos relevantes dentro de todos los recuperados.

$$Precision = \frac{TP}{TP + FP}$$

2.7.2. Average Precision

En *retrieval*, el *average precision* (AP) corresponde al promedio de los valores del *precision* en las posiciones donde se recupera un elemento relevante. Al calcularse en razón de los resultados ordenados, el AP evalúa en conjunto la relevancia y el orden de los resultados mostrados por un sistema de recuperación.

Por ejemplo, considerando los sistemas de recuperación A y B, en los que se realiza una *query* común para la que se esperan tres resultados, y la relevancia de cada resultado mostrado a continuación (donde un 1 representa que se recupera un elemento relevante y un 0 uno no relevante):

$$A : [1, 1, 0]$$
$$B : [1, 0, 1]$$

Se tiene que el *average precision* obtenido para cada uno es:

$$AP_a = \frac{1}{3} \cdot \left(\frac{1}{1} + \frac{2}{2} + 0 \right) = 0{,}67$$

$$AP_b = \frac{1}{3} \cdot \left(\frac{1}{1} + 0 + \frac{2}{3} \right) = 0{,}55$$

Donde a pesar de que ambos sistemas muestran la misma cantidad de resultados relevantes, el AP resulta mayor si estos se muestran antes.

En casos donde el número de artículos relevantes disponible es elevado y es de interés la recuperación solo de algunos, se puede limitar la cantidad a un número de resultados relevantes k. En este caso, la métrica toma el nombre de *average precision at k* (AP@k).

Otra manera de obtener el AP, es calculando el área bajo la curva *precision-recall*, tanto en un contexto de clasificación como de *retrieval*, sin embargo, en este trabajo se utiliza el AP según la primera definición dada.

Mean Average Precision (mAP)

El mAP es usado para evaluar los resultados de un sistema de *retrieval*. Este se calcula como el promedio de los AP de todas las consultas, indicando el promedio de elementos relevantes de todos los recuperados. Para un número N de *queries*, el mAP se formula de la siguiente manera:

$$\text{mAP} = \frac{\sum_{\text{i}=1}^{N} AP_{\text{i}}}{N}$$

Cuando se considera una cantidad limitada de resultados relevantes k, el mAP se denomina mAP@k y se calcula promediando los AP@k de cada consulta hecha sobre el sistema.

2.7.3. *Recall*

También conocido como la tasa de verdaderos positivos, corresponde a la razón de ejemplos predichos como positivos, sobre todo el universo de positivos existentes. No suele usarse en *retrieval*, puesto que generalmente el universo de positivos es demasiado grande, lo que vuelve difícil interpretar el desempeño con esta métrica.

$$Recall = \frac{TP}{TP + FN}$$

2.7.4. *Accuracy*

En clasificación, el *accuracy* mide la cantidad de ejemplos positivos y negativos que fueron clasificados como tal, sobre todos los ejemplos existentes. Este se define como:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2.7.5. Curva *precision-recall*

La curva *precision-recall* muestra la variación del *precision* en función del *recall*. Los cambios en estas métricas se dan al considerar distintos umbrales de confianza en la clasificación. Con estas dos variables, logra mostrar qué tan "limpias" son las predicciones de un modelo, a medida que se predicen más ejemplos como positivos, lo que ocurre al disminuir el umbral de confianza necesario por un modelo para encontrar la presencia de una clase.

2.7.6. Curva ROC

La curva ROC (*Receiver Operating Characteristic*) es similar a la curva *precision-recall*, caracterizando la tasa de falsos positivos contra la de verdaderos positivos. Al igual que la curva *precision-recall* puede servir tanto para determinar el desempeño de un modelo de clasificación binaria, como también para determinar el punto de operación deseado. Una diferencia importante que tiene con la curva *precision-recall* es que no considera el universo de ejemplos negativos, por lo que cuando se tiene una gran cantidad de ellos, la evaluación de los modelos con esta curva puede resultar engañosa.

2.7.7. *Intersection over union*

La *intersection over union*, o IoU, es una métrica usada para la evaluación de modelos de segmentación de imágenes. Ella mide la razón entre la intersección y la unión de la zona donde se predice que existe un objeto de una clase determinada, con la zona real de donde este se encuentra (*ground truth*). De esta manera, nos indica que tan ajustada a la zona real es nuestra predicción, siendo mejor cuando más alto es el valor de IoU. La Figura 2.12 ilustra esta métrica entre dos conjuntos.

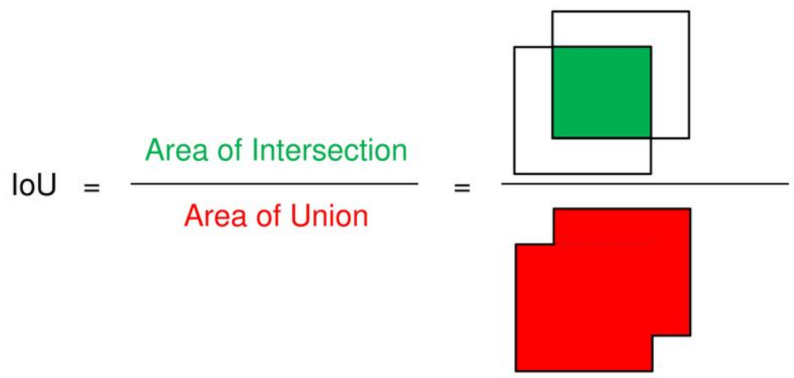


Figura 2.12: Ilustración de la métrica IoU.

2.8. *Term frequency-inverse document frequency*

Term frequency-inverse document frequency, o tf-idf, es una medida de relevancia usada en búsquedas por texto para encontrar los documentos más relevantes a un término o palabra. Para esto, utiliza otras dos medidas en función de la frecuencia de un término en un documento (*term frequency*), y un ajuste inverso a la frecuencia del término en todos los documentos (*inverse document frequency*) de un universo.

Según el *term frequency*, se considera más relevante un documento que contiene más veces un término que otro. El *inverse document frequency* por su parte, se encarga de ajustar la relevancia según el inverso de la frecuencia de un término entre todos los documentos, para restarle importancia a conectores y otras palabras, que no aportan mucho al contexto del documento al repetirse mucho. Matemáticamente, *Term frequency* (tf) se formula como la frecuencia de un término en un documento $f_{t,\mathrm{d}}$. Para considerar el largo variable de los documentos, este valor se normaliza por el largo de estos. *Inverse document frequency* (idf) por su parte, cuenta el número de documentos donde aparece un término y les da un peso inversamente proporcional a este, escalado según la función logaritmo.

Su formulación se muestra a continuación, donde t es un término, d un documento, $|\mathrm{d}|$ su largo, D el universo de documentos y $|D|$ el total de documentos del universo.

$$\mathrm{tf}(t,\mathrm{d}) = \frac{f_{t,\mathrm{d}}(t)}{|\mathrm{d}|}$$

$$\mathrm{idf}(t,D) = log\left(\frac{|D|}{|\mathrm{d}\in D : t\in \mathrm{d}|}\right)$$

$$\text{tf-idf}(t,\mathrm{d},D) = \mathrm{tf}(t,\mathrm{d})\cdot \mathrm{idf}(t,D)$$

Para realizar búsquedas con *tags*, se puede considerar cada *tag* como un término y al conjunto de *tags* asociados a un archivo como los documentos.

2.9. Similitud coseno

La similitud coseno es una medida de similitud entre vectores, calculada como el coseno del ángulo entre ellos, o equivalentemente, como el producto punto dividido por el producto de sus magnitudes.

Esta métrica considera similares los vectores según su orientación sin importar su magnitud. Su rango de valores va entre $[-1,1]$. Indica completa similitud cuando vale 1, completa disimilitud cuando vale -1, y no encuentra relación cuando vale 0. Considerando los vectores $\vec{A}$ y $\vec{B}$, la similitud coseno se calcula como:

$$\text{similitud coseno}(\vec{A},\vec{B}) = \frac{\vec{A}\cdot\vec{B}}{|\vec{A}||\vec{B}|}$$

En búsquedas por texto, como con tf-idf, puede usarse para medir la similitud semántica entre documentos, considerando vectores donde a cada componente se le asigna el tf-idf de un término.

Capítulo 3

Estado del arte

En el presente capítulo se presenta la investigación de trabajos orientados al estudio y desempeño de distintos métodos relevantes al etiquetado de imágenes de vestuario. En las secciones a continuación se entrega una descripción de los trabajos según la metodología empleada.

3.1. Clasificación

Los trabajos basados en métodos de clasificación para realizar etiquetado de imágenes, definen una serie de clases que determinan las características a encontrar en ellas, según lo cual se les asigna una etiqueta. Estos se pueden separar a su vez en dos grupos: clasificación con descriptores de atributos visuales, y clasificación con redes neuronales convolucionales (CNN).

3.1.1. Clasificación con descriptores

Los trabajos basados en el uso de descriptores de imágenes se caracterizan por extraer atributos o características adecuados a un ambiente, y en general obtienen buenos resultados al costo de ser poco generalizables.

Tomasik et al. [21] utilizan *k nearest neighbors* sobre un espacio de *bag of visual words* de descriptores SIFT de imágenes de Amazon[1], para clasificarlas entre 12 clases referentes a detalles de bajo nivel de los artículos. Aunque se obtiene cerca de un 80 % de *accuracy* de clasificación de algunas clases, se observa que el ambiente de las fotografías usadas es muy controlado, lo cual es una característica que beneficia mucho a estos métodos, que requieren ser meticulosamente ajustados y resultan poco flexibles en otros ambientes.

Ben Salem et al. [2] utilizan descriptores de atributos visuales como el contraste, correlación de píxeles, la energía y homogeneidad de imágenes de tres clases de telas para clasificarlas con *support vector machines* (SVM). Con esta metodología obtienen un *accuracy* cercano al 100 % con algunos de estos descriptores. Se reconoce en este trabajo que no es posible ge-

[1] `https://www.amazon.com/`

neralizar este método para reconocer visualmente todos los tipos de telas, debido a que no todas presentan texturas lo suficientemente distinguibles entre ellas.

3.1.2. Clasificación con CNN

Los trabajos orientados a la clasificación de imágenes con redes neuronales convolucionales son más diversos que los métodos previamente presentados, dada la mejor capacidad de generalización de estos algoritmos en distintos ambientes. En general, muestran que los modelos que comparten información de todas las clases tienen una ventaja sobre aquellos que no. El uso de redes neuronales convolucionales para la limpieza y asignación de etiquetas a imágenes muestra tener la ventaja de poder ser aplicable a grande escala, a diferencia de otras metodologías que crecen en complejidad con el número de datos como *label propagation* [25].

Abdulnabi et al. [1] comparan el desempeño de varios modelos de clasificación binaria independiente contra modelos de clasificación binaria pero que comparten parte de su estructura con otros. Muestran que los modelos que comparten su estructura presentan cerca de un 5 % más de *accuracy* en la tarea de clasificación de atributos de vestuario de distinta naturaleza como colores, texturas y formas, entre otros. Sin embargo, tienen la desventaja de ser más complejos en su implementación, la que puede crecer inmensamente cuando se tiene una alta cantidad de clases. Los modelos independientes tienen la ventaja de ser más fáciles en su implementación, y de permitir la extracción de atributos especializados de una clase para su uso en otras tareas, como búsqueda por similitud de vectores.

Inoue et al. [8] comparan el desempeño de modelos CNN independientes con otros comparten información de todas las clases para discriminar entre ruido de fondo y características de bajo nivel de prendas que pueden estar correlacionadas, como mangas de trajes y relojes por ejemplo. Muestran que la mejora en desempeño al utilizar modelos que comparten características visuales puede variar entre un 2 % a un 40 % en el *accuracy* obtenido en la clasificación de algunas clases. Además, muestran la utilidad del modelo implementado realizando limpieza en una bases de datos, quitando etiquetas mal asignadas producto a esta confusión entre atributos de alto y bajo nivel.

Tong Xiao et al. [23] utilizan una red neuronal convolucional para realizar etiquetado y filtrado de etiquetas ruidosas de imágenes de vestuario, entrenando modelos con una porción pequeña de datos bien etiquetados y otros con una porción masiva de datos con etiquetas ruidosas. Con esta red clasifican las etiquetas de imágenes tomadas de Internet como ruidosas o no, con el fin de eliminar aquellas que se determina que están mal asignadas. Demuestran que el uso de una base de datos pequeña pero limpia para entrenar los modelos en una primera instancia mejora los resultados sobre otras técnicas como *transfer learning* [15], mostrando que las imágenes bien etiquetadas tienen un gran valor para llevar más allá tareas relacionadas.

3.2. Extracción de atributos

Los vectores de atributos corresponden a representaciones numéricas de características visuales de las imágenes. Estas son de utilidad para realizar búsquedas por similitud. Los

trabajos mostrados a continuación se enfocan en la extracción de atributos de redes neuronales convolucionales (*deep features*), con las que según algún criterio de similitud entre imágenes con ellas, poder etiquetar o quitar etiquetas ruidosas a imágenes de artículos de vestuario.

Zakizadeh et al. [24] utilizan una CNN para realizar limpieza sobre la base de datos *DeepFashion* [13], la cual es una base de datos de moda ampliamente usada en tareas de clasificación debido a su gran número de clases referentes a detalles de bajo nivel de prendas de vestuario. Con esta red, extraen *deep features* de las imágenes para limpiar etiquetas ruidosas, comparando estos vectores de atributos con otros de imágenes de las que se tiene certeza de que están bien etiquetadas. Muestran en su trabajo que el entrenamiento de CNN con los datos limpios mejoran el *accuracy* en la clasificación de atributos de bajo nivel alrededor de un 30 %.

Saha et al. [17] implementan una serie de redes neuronales convolucionales que aprenden a representar distintas características de prendas de vestuario como color y patrones de las telas, junto con las correlaciones que existen entre estas distintas características. De estas redes obtienen vectores de atributos bien definidos en cada componente, es decir, donde las k_1 primeras componentes representan de color, las siguientes k_2 texturas, etc; lo que les permite realizar búsquedas más detalladas que con *deep features* extraídas de alguna capa *fully-connected* de una CNN. Considerando las correlaciones, son capaces de mejorar los resultados de búsqueda en comparación a modelos que no consideran las correlaciones. Su aplicabilidad en la industria de la moda, es que permite mejorar los resultados de búsquedas por texto, permitiendo darle más importancia a determinadas características por sobre otras de manera simple.

3.3. Recuperación de imágenes

Los trabajos enfocados en la búsqueda y recuperación de imágenes se centran en métodos que permitan definir un criterio de relevancia adecuado para entregar los resultados de búsquedas (*queries*). Las dos maneras principales en que los trabajos llevan a cabo esta tarea es mediante la recuperación de imágenes basada en contenido visual (*content based image retrieval*, CBIR), y por texto o *tags* (*tag based image retrieval*, TBIR).

Huant et. al. [7] realizan recuperación de imágenes de prendas de vestuario de catálogos profesionales en base a su similitud visual con imágenes de la misma índole, pero tomadas por personas comunes en ambientes ruidosos, denominado *cross domain image retrieval*. Para esto, emplean una red neuronal convolucional que aprende a crear representaciones comunes para imágenes de ambos ambientes, lo que permite obtener vectores de características de imágenes que permitan la búsqueda y recuperación de artículos a partir de imágenes de cualquiera de ellos. Demuestran que con el uso de una red especializada para la creación de representaciones de características visuales en distintos ambientes, pueden obtener cerca del doble de *top-20 accuracy* en la recuperación de artículos, en comparación a cuando se utilizan vectores tomados de redes no especializadas para la creación de estos vectores, mostrando que existen problemas de generalización de características al emplear la última metodología.

Wu et al. [22] proponen un *framework* para realizar completado de etiquetas de imágenes tomadas de Internet para mejorar los resultados de búsqueda y recuperación. En este, consi-

deran la correlación entre etiquetas en razón de su ocurrencia conjunta en las imágenes y la similitud visual entre imágenes, para definir un criterio que determine cuando se deba asignar una etiqueta a una imagen mal anotada, en razón de qué tan similar resulta a otras visual y semánticamente. Luego de corregir etiquetas de cuatro *datasets*, llevan a cabo la recuperación de imágenes con los *tags* disponibles en cada uno de ellos. Con este método, obtienen mejores resultados que otros métodos de etiquetado de imágenes, al costo de resultar más costoso computacionalmente.

Capítulo 4

Metodología

En este capítulo se detallan las tareas llevadas a cabo para la implementación del sistema de *auto tagging* propuesto como objetivo de este trabajo.

La metodología seguida para la implementación y la evaluación de la solución propuesta se separa en cuatro tareas principales, detalladas en las secciones que siguen. A continuación se da un detalle general de cada una:

- Recolección de datos (Sección 4.1): Se detallan las bases de datos recolectadas. Esta tarea es necesaria, puesto que para la implementación de CNN es un requisito contar con una alta cantidad de imágenes.
- Definición de clases (Sección 4.2): Se muestra la definición de las clases que este trabajo plantea que podrían aplicarse como filtros en un catálogo virtual de *ecommerce*. Se definen 9 clases de color y 40 clases de características visuales de prendas, según las clases disponibles en las bases de datos utilizadas y los procesos de filtrado llevados a cabo (ver Sección 5.1). Adicionalmente, se definen otras 4 clases para una tarea de segmentación auxiliar implementada en el sistema de clasificación de color. Para cada conjunto se definen imágenes asociadas, usadas en el entrenamiento y la evaluación de los algoritmos implementados, lo que se detalla en la sección siguiente.
- Sistema de etiquetado (Sección 4.3): Se muestra la composición de los sistemas de clasificación. Se presentan las arquitecturas CNN, datos usados para el entrenamiento y evaluación de cada una, junto con las métricas usadas para esto último.
- Retrieval (Sección 4.4): Como última tarea, se lleva a cabo un proceso de búsqueda y recuperación de imágenes (*retrieval*) de catálogos de tiendas reales, utilizando las cada una de las clases definidas como términos de búsqueda. Esto, con el objetivo de darle validez al sistema en un contexto real, mostrando los resultados de filtrado de un catálogo real de artículos de vestuario.

4.1. Recolección de datos

En esta sección se detalla y muestra la composición de las bases de datos recolectadas para su uso en la implementación y evaluación de los distintos métodos.

Se utilizan dos bases de datos públicas de moda, *DeepFashion* [13] y *DeepFashion* 2 [5]; y una tercera base de datos privada, compuesta por imágenes de tres tiendas reales de vestuario. La composición detallada de cada una se muestra a continuación.

4.1.1. DeepFashion

DeepFashion es una base de datos pública de imágenes de vestuario. Contiene sobre 800.000 imágenes tomadas en ambientes profesionales e informales. Separa sus datos en cuatro *benchmarks* enfocados en distintas tareas: *Attribute Prediction, Consumer-to-shop Clothes Retrieval, In-shop Clothes Retrieval* y *Landmark Detection*. En este trabajo se utiliza el *Attribute Prediction benchmark*, que está enfocado en la predicción de atributos de las prendas (correspondientes a detalles de bajo nivel) y está conformado por 289.222 imágenes distribuidas en 50 categorías de prendas y 1.000 clases de atributos, agrupados en 5 tipos de clases: de textura (*texture*), tipo de tela (*fabric*), forma (*shape*), particularidades (*part*) y estilo (*style*), como se muestra en la Figura 4.1.

Figura 4.1: Muestra de la composición de imágenes del *benchmark Attributte Prediction* de DeepFashion.

4.1.2. DeepFashion 2

Este *dataset* contiene 491.000 imágenes en las que existen 801.000 instancias de prendas en total. Tiene anotaciones con información detallada de cada imagen. Dentro de la información relevante para este trabajo, se encuentran:

- *Source*: La fuente de la imagen, que puede ser de tienda (*shop*) o de una persona común (*user*).
- *Category name*: Categoría de la prenda, que pueden ser una o varias de las 13 siguientes clases: *short sleeve top, long sleeve top, short sleeve outwear, long sleeve outwear, vest, sling, shorts, trousers, skirt, short sleeve dress, long sleeve dress, vest dress* o *sling dress.*

- *Segmentation*: Coordenadas de los puntos que comprenden el o los polígonos que delimitan la zona que ocupa cada prenda en la imagen.

En la Figura 4.2 se puede observar una muestra de las imágenes de esta base de datos, con parte de la información de las anotaciones visualizada.

Figura 4.2: Muestra de la composición de la base de datos *Deep Fashion* 2 junto con la visualización de la información de las anotaciones de las imágenes.

4.1.3. *Dataset* privado

El *dataset* privado se compone de 2.345 imágenes de tres catálogos de tiendas de vestuario distintas: 882 de Bronzesnake[1], 833 de Ring of Fire[2] y 630 de Umbrale[3]. Es creado en este trabajo para evaluar los sistemas implementados en un ambiente real de *ecommerce* de vestuario.

Las imágenes de este *dataset* son etiquetadas con los nombres de las clases definidas en este trabajo. Además, se toman algunas de estas imágenes para definir subconjuntos de datos usados para la evaluación de otras tareas no relacionadas al *retrieval*, mostrado más en detalle en la Sección 4.2.

Una muestra de las imágenes que componen este *dataset* se puede apreciar en la Figura 4.3.

[1] `https://bronzesnake.com/`
[2] `https://ringoffireclothing.com/`
[3] `https://www.umbrale.cl/`

(a) Bronzesnake (b) Ring of Fire (c) Umbrale

Figura 4.3: Muestra de las imágenes de las tres tiendas que componen el *dataset* privado.

4.2. Definición de clases

En esta sección se muestran las clases que representan características de prendas de vestuario definidas, que se propone podrían implementarse como filtros en catálogos de vestuario, además de un conjunto de clases auxiliares usadas en la tarea de segmentación llevada a cabo. Esta definición comprende solo el criterio y origen de las imágenes que conforman las clases. El número de imágenes de cada clase, utilizado para el entrenamiento o evaluación de los métodos implementados se muestran más adelante en la Sección 4.3.

Las clases definidas se pueden separar en cuatro grupos: atributos, categorías, colores, y segmentación. Las características y composición de imágenes de cada uno se muestra en las subsecciones que siguen, según cada grupo.

4.2.1. Atributos

Los atributos son las clases más relevantes en este trabajo, puesto que representan características de bajo nivel en prendas de vestuario, como forma, diseño o tela, que son los filtros más escasos en los catálogos virtuales y por ende los más interesantes de aplicar. La definición de atributos se hereda del *benchmark Attribute Prediction* de *DeepFashion*, cuyas clases e imágenes definen clases de atributos en este trabajo.

En el subconjunto de datos de *DeepFashion* usado, se definen 5 tipos de atributos distintos, que son conjuntos de clases referentes a distintas características de las prendas. A continuación, se entrega una descripción de cada uno de estos tipos:

- *Texture*: Texturas o diseños que tiene una prenda, como flores, cuadros o bordados, entre otros.
- *Fabric*: Corresponde a la tela de una prenda, como mezclilla, lana o cuero, entre otros.
- *Shape*: Son distintas formas de las prendas, como suelta o ajustada al cuerpo, el largo de corte, entre otros.
- *Part*: Corresponden a distintas particularidades de muy bajo nivel de las prendas, como si tienen cierre, botones o bolsillos.
- *Style*: Es el estilo de una prenda o tenida, como formal, deportivo o tierno, entre otros.

Un estudio de la literatura hecho en este trabajo con el *Attribute Prediction benchmark* de *DeepFashion* [24], evidencia que existe desbalance de datos y redundancia entre clases, lo que se puede mejorar con un proceso de filtrado. Por esto, el conjunto de clases con sus respectivas imágenes asociadas, se definen a través de un filtrado de esta base de datos. Se exploran dos procesos de filtrado detallados a continuación.

Filtrado por similitud semántica

Un primer proceso de filtrado sobre los atributos se basa en calcular la similitud semántica entre los nombres de las clases para fusionarlas. Para esto, se utiliza un modelo Word2Vec entrenado por Google con más de 3 billones de palabras en inglés [4]. Con este, se obtienen vectores para el nombre de cada clase y se calcula la similitud coseno entre ellos, mezclando las clases con una similitud superior al 50 %. Esta mezcla se realiza considerando las clases por cada tipo de atributo definido por separado, es decir, solo entre clases tipo *texture*, solo entre clases tipo *fabric*, etc. El nombre de la clase resultante de la fusión de dos clases es el nombre de aquella con mayor número de imágenes. Finalmente, se aborda el desbalance eliminando todas aquellas clases con un número de imágenes inferior a 3.000. Se elige este número porque se observa que permite tener una muestra relativamente versátil de clases de cada tipo.

Filtrado manual

En vista de los resultados obtenidos por el filtrado semántico, en los que se observan muchas relaciones entre clases, que según criterio del autor, no corresponden, se lleva a cabo un segundo proceso de filtrado de las clases de *Deep Fashion* para definir los atributos utilizados. Este intenta disminuir el desbalance de clases existente, y en definir clases representativas de características de prendas de vestuario que se observan de manera relativamente común en nuestra cultura, y que puedan ser por ejemplo un criterio de búsqueda que usaría un cliente para buscar prendas en tiendas del *ecommerce* nacional.

El proceso de filtrado consiste en primero, eliminar todas aquellas clases con un número de imágenes menor a 3.000, luego eliminar las clases redundantes, conservando aquella con el mayor número de imágenes, y luego llevando a cabo un proceso de filtrado manual sobre las imágenes de cada clase. En vista de lo poco reproducible de este método, se propone como trabajo futuro la implementación de una metodología más objetiva para definir clases de atributos, o el uso de otra base de datos para la implementación de un sistema como el propuesto.

4.2.2. Categorías

Las categorías hacen referencia a clases de prendas como son pantalones, vestidos y chaquetas, entre otros. A pesar de que *DeepFashion* contiene este tipo de clases, como "Category", en el *Attribute Prediction Benchmark*, como se ve en la Figura 4.1, ya que las clases de *DeepFashion* en general tienen definiciones de clases demasiado específicas, con pocas imágenes, y en algunos casos de mala calidad, se prefiere el uso de las categorías de *DeepFashion* 2 para definir las de este trabajo. Las clases originales de prendas de *DeepFashion* 2 son las

[4] `https://code.google.com/archive/p/word2vec/`

trece siguientes: *short sleeve top, long sleeve top, short sleeve outwear, long sleeve outwear, vest, sling, shorts, trousers, skirt, short sleeve dress, long sleeve dress, vest dress* y *sling dress*, algunas de las cuales se diferencian por detalles de bajo nivel, como se ve en su representación de la Figura 4.4, lo que motiva filtrar estas clases también.

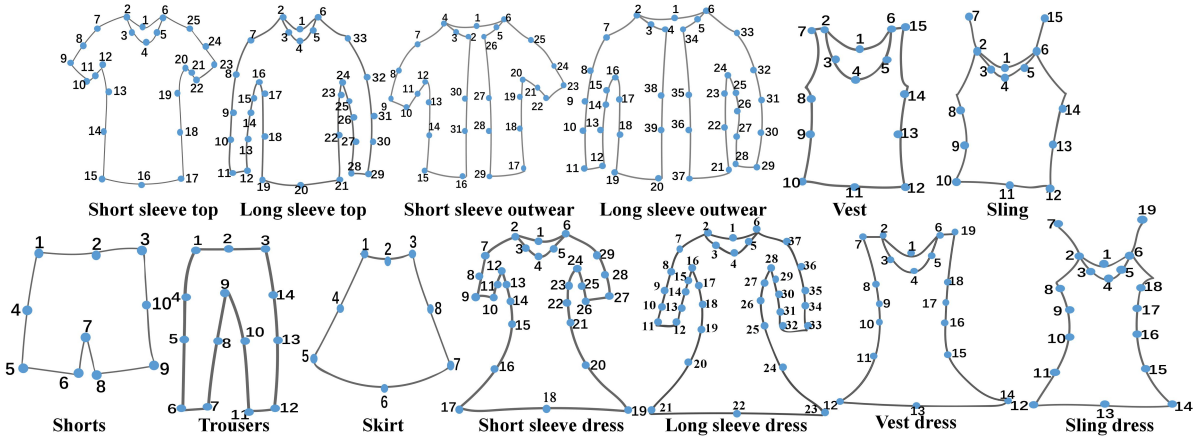


Figura 4.4: Clases de prendas de *DeepFashion* 2.

Filtrado

Como se menciona, el proceso de filtrado de *DeepFashion* 2 se realiza motivado por el hecho de que se observan pequeñas diferencias entre sus clases, que las vuelven redundantes, y que algunos casos, ya están consideradas en alguna clase de atributos (como el largo de mangas por ejemplo).

Este proceso es más simple que el de los atributos, y considera solo mezclar clases de un mismo tipo de prenda, es decir, todos los tipos de vestido (*dress*), de chaquetas (*outwear*), y de prendas de la parte superior del cuerpo (*top*) en una clase para cada uno. La agrupación realizada se resume en la Tabla 4.1.

Clase	**Compuesta por**
Top	Short sleeve top, long sleeve top, vest, sling
Outwear	Short sleeve outwear, long sleeve outwear
Dress	Short sleeve dress, long sleeve dress, vest dress, sling dress
Shorts	Shorts
Trousers	Trousers
Skirt	Skirt

Tabla 4.1: Composición de las clases filtradas con clases originales de categorías de *Deep Fashion* 2.

4.2.3. Colores

Además de características visuales particulares de prendas de vestuario, se proponen 9 clases de color para la implementación de filtros en catálogos. A pesar de que estas características suelen estar incluidas como filtros en la mayoría de catálogos actualmente, se proponen estas clases para hacer un estudio particular al contexto de vestuario.

Se busca que las clases de color representen colores simples y comunes. En vista de esto, las clases definidas son: blanco, gris, negro, rojo, amarillo, naranjo, morado, azul y verde.

Ya que la definición de estas clases se realiza solo por criterio del autor y no a partir de clases definidas en una base de datos, no se cuentan con imágenes de cada clase disponibles. Para evaluar el sistema clasificador de color se define un conjunto de 100 imágenes por cada clase, tomadas del *dataset* privado. Ya que este sistema no requiere ejemplos de color para la clasificación de esta característica, no se requiere un conjunto de entrenamiento.

4.2.4. Segmentación

Las clases de segmentación se proponen para representar clases de prendas que ocupan distintas zonas del cuerpo. Esto, con el objetivo de poder distinguir entre objetos de estas clases en una imagen mediante segmentación semántica, y extraer las zonas que ocupan las prendas para encontrar el color de cada una, planteando que solo distinguir entre las zonas del cuerpo basta para encontrar el color de una prenda, sin importar otras características de ella.

Se plantean diferencias entre tres tipos de prendas, las que cubren el torso, las piernas y el cuerpo completo; según las cuales se definen las clases *top*, *bottom* y *body* para representar cada una, respectivamente. Adicionalmente, se define una clase denominada fondo o *background*, que representa la zona que no resulta interesante en la imagen.

En vista de que el modelo de segmentación implementado requiere de un número alto de imágenes de entrenamiento, se agrupan las clases de *DeepFashion* 2 para calzar con las definidas en este trabajo y ocupar sus imágenes. La agrupación realizada se muestra en la Tabla 4.2.

Clase segmentación	**Compuesta por (clase de *DeepFashion* 2)**
Top	Short sleeve top, long sleeve top, short sleeve outwear, long sleeve outwear, vest, sling
Bottom	Shorts, trousers, skirt
Body	Short sleeve dress, long sleeve dress, vest dress, sling dress

Tabla 4.2: Definición de clases de segmentación a partir de las clases de *DeepFashion* 2.

4.2.5. Resumen

En la presente sección se muestra la definición de clases y las imágenes usadas para entrenar y evaluar los modelos implementados.

En base a las clases y ciertas definiciones usadas en las bases de datos públicas recolectadas, se definen las clases tipo categoría y atributos, donde un tipo corresponde a un conjunto de clases referentes a determinada característica de una prenda como telas o texturas.

Además se definen 9 clases de color que se desean encontrar en las prendas. Para llevar a cabo la clasificación de color como se ve más adelante, es necesario llevar a cabo una tarea

de segmentación semántica. Por lo cual, también se definen 4 clases para esta tarea.

4.3. Sistema de etiquetado

En esta sección se muestran los detalles de la implementación del sistema de etiquetado propuesto como objetivo de este trabajo. Para construirlo, se emplea el uso de redes neuronales convolucionales para realizar clasificación dentro de las clases definidas mostradas, y según ello asignar *tags* a imágenes de catálogos de tiendas de vestuario.

La clasificación llevada a cabo se separa en dos partes: categorías y prendas, y colores. Para la clasificación de categorías y prendas se utilizan dos arquitecturas CNN de clasificación distintas, ResNet50 e Inception-v3, de las cuales se implementan modelos de clasificación binaria y modelos de clasificación multi-etiqueta, con el fin de estudiar el de mejor desempeño en la tarea de clasificación en clases referentes a vestuario. Para la clasificación de color, se construye un sistema basado en la clasificación de los píxeles que conforman las prendas en una imagen, para lo cual se emplea un modelo de segmentación DeepLab-v3 para encontrar las máscaras de las prendas, y el algoritmo *k-means* para encontrar el color.

La implementación de todas redes utilizadas se realiza con la librería `PyTorch` para el lenguaje `Python`. Para el entrenamiento e inferencia con los modelos, se utiliza una GPU `Nvidia GeForce GTX1050ti`.

En las subsecciones que siguen se detallan los métodos, modelos, datos y métricas usadas en la clasificación de imágenes de las clases de las dos partes mencionadas.

4.3.1. Categorías y atributos

La clasificación de categorías y atributos se lleva a cabo con dos arquitecturas CNN: ResNet50 e Inception-v3. De cada una se implementan modelos de clasificación binaria de clases y clasificación multi-etiqueta. Los modelos multi-etiqueta se implementan en este caso por cada tipo de atributo y categorías por separado, existiendo un modelo multi-etiqueta para cada tipo por separado, y un modelo binario para cada clase.

Todos los modelos son entrenados a partir de modelos pre-entrenados con `ImageNet`, disponibles en `PyTorch`. Sin embargo, se encuentra de manera experimental que el mejor desempeño se alcanza congelando algunas capas para los modelos multi-etiqueta, y entrenando la red completa en los modelos binarios, usando el modelo pre-entrenado solo como inicialización de pesos. Así, no se congela ninguna capa para los modelos binarios, las primeras 12 para ResNet50 multi-etiqueta y las primeras 10 para Inception-v3 multi-etiqueta. En el entrenamiento de todos los modelos se utiliza la función de *loss binary cross entropy* y un optimizador Adam con *learning rate* de 10^{-5}.

Los conjuntos de imágenes de entrenamiento, validación y test para los modelos binarios se componen tomando todas las imágenes disponibles de una clase como ejemplos positivos, y un 125 % de esta cantidad de ejemplos negativos, elegidos en igual proporción de entre las demás clases del mismo tipo de la clase que se tomaron los ejemplos positivos. En casos donde una clase no puede aportar con una cantidad de ejemplos negativos suficientes, se

compensa tomando ejemplos extras de las demás clases. Para los modelos multi-etiqueta, se eligen todas las imágenes de cada clase como ejemplos positivos, con un máximo de 6.000 instancias de cada una para evitar tener un gran desbalance respecto a las clases con pocos ejemplos. Se elige este número dado que *Zakizadeh et al.* [24] plantea que este es un número razonable de imágenes para lograr un entrenamiento exitoso de CNN para distinguir detalles finos de vestuario.

Una vez definidos los ejemplos positivos y negativos de cada clase para cada modelo, se separan en un 70 % para entrenamiento, 15 % para validación y 15 % para test. El número de ejemplos específicos de las clases se muestra en la Figura 5.4. La evaluación de estos modelos se realiza mediante las curvas *precision-recall* y ROC obtenidas sobre el conjunto de test. Para construir esta curva para los modelos multi-etiqueta, se considera el problema de clasificación de cada una de sus clases de manera independiente del resto.

4.3.2. Colores

Para clasificar el color de las prendas, la imagen se segmenta para extraer las zonas que ocupan las prendas en ella y se clasifica el color de cada una según el valor de los píxeles de cada zona. Como ha sido mencionado, la clasificación de color se realiza entre las 9 clases de colores: blanco, gris, negro, amarillo, naranjo, rojo, verde, azul y morado; y la segmentación entre 3 clases que representan las zonas del cuerpo que usan las prendas: del torso (*top*), de las piernas (*bottom*) y de cuerpo completo (*body*), además de una clase que representa los objetos a ignorar en la imagen, el fondo (*background*).

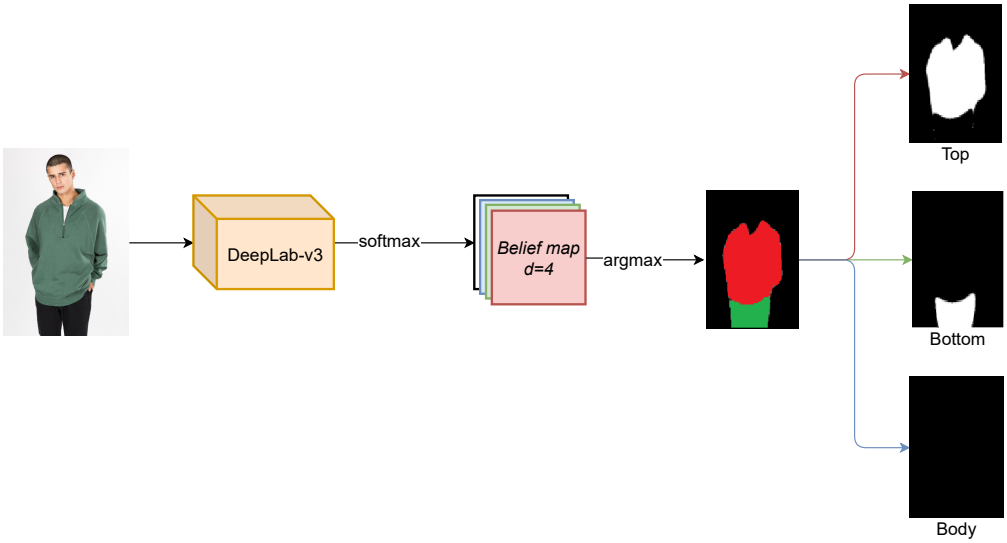


Figura 4.5: Obtención de las máscaras de las prendas a partir del modelo de segmentación.

La extracción de las máscaras de las prendas se realiza con el uso de un modelo CNN de segmentación semántica, DeepLab-v3. Como se muestra en la Figura 4.5, del modelo de segmentación se obtiene una imagen con los píxeles clasificados en las cuatro clases definidas, el que se separa en tres máscaras que indican las zonas de cada prenda por separado.

Una vez obtenidas las zonas de cada prenda, con cada una se lleva el proceso ilustrado en la Figura 4.6, en el que se extraen las tuplas (R,G,B) de la zona de la prenda en la

imagen, se transforman a espacio CIELAB y se agrupan en 3 *clusters* con *k-means*, donde cada centroide se asume como el representante de uno de los colores predominantes de la zona. Las tres tuplas (L*,a*,b*) obtenidas de *k-means* luego se transforman al espacio HSV, donde se determina la pertenencia a una clase de color según rangos de los valores H, S y V.

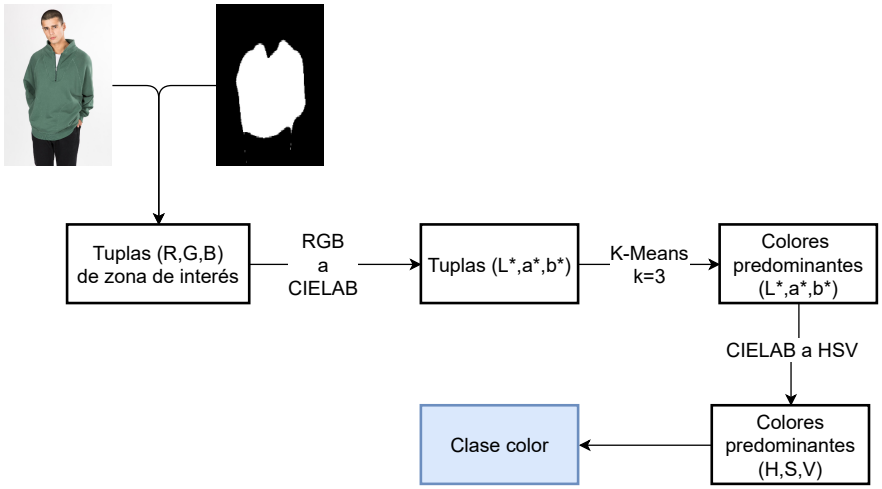


Figura 4.6: Proceso de clasificación de color de una prenda.

Se eligen 3 *clusters* para representar los colores predominantes de las prendas, puesto que, como se muestra en la Figura 4.7, al introducir más, se obtienen colores muy similares que no aportan para clasificar. Además, elegir pocos *clusters* ayuda a eliminar objetos pequeños que aparecen ocasionalmente a causa de errores en la tarea de segmentación, como accesorios o pelo, quitando un poco de ruido.

(a) Original (b) $k = 3$ (c) $k = 5$ (d) $k = 7$

Figura 4.7: Imagen original e imágenes con la prenda (vestido) cuantizada con distinto número de colores predominantes (k).

En particular, se utiliza una variante de esta arquitectura llamada DeepLab-v3-ResNet50, que utiliza bloques de una ResNet50 para extraer características que luego son alimentadas a su propia arquitectura. De la red, se ajustan las dimensiones de la última capa convolucional para entregar un mapa de características de 4 dimensiones, cual se evalúa con una función *softmax*. Esto entrega un mapa de creencias (*belief map*), del que se puede obtener la clasificación de cada píxel según la dimensión que tome el máximo valor en la posición correspondiente del *belief map*.

Para entrenar la red de segmentación, se utilizan imágenes de *DeepFashion* 2 que componen 8.032 instancias de prendas de la parte superior del cuerpo, 8.012 de prendas de parte inferior y 8.000 instancias de prendas de cuerpo completo, según la agrupación de clases definidas (ver Sección 4.2.3). Se utiliza una cantidad balanceada de datos para evitar el sobreajuste de la red, y limitada puesto al alto costo computacional que conlleva el entrenamiento de este modelo. La evaluación por su parte, se realiza con el conjunto de datos mencionado en la sección anterior, compuesto por 100 imágenes con máscaras hechas manualmente sobre imágenes de todos los *dataset*.

Para el entrenamiento, se utiliza la función de *loss cross entropy* y optimizador Adam con un *learning rate* de 10^{-5}. Un detalle importante es que la función de *loss* se pondera para medir el error por clase de: 0,5 para *background*, 1,0 para *top* y *bottom*, y 1,5 para *body*. Con esto se le da énfasis a la clasificación de prendas de cuerpo completo, las menos comunes según inspección de las bases de datos; y se le quita al fondo, la más presente.

Encontrada la clase de un color predominante, se impone un criterio para determinar que un color está presente en la imagen, que consta en imponer que los píxeles de un *cluster* de color, sean mayores en cantidad al 20 % de la cantidad de píxeles de la máscara de una prenda. Esto se hace para eliminar ruido introducido por objetos que no son prendas en las imágenes (bordes mal segmentados, accesorios y pelo de los y las modelos, entre otros), que generalmente representan pocos píxeles de la máscara.

La evaluación de la clasificación de color comprende evaluar tanto el desempeño del modelo de segmentación según el IoU, como también la clasificación de colores según el *accuracy* sobre el conjunto de test de color.

4.3.3. Resumen

Se implementan dos sistemas de clasificación. Uno encuentra características visuales particulares de prendas (telas, formas, diseños), denominadas clases de categorías y atributos, y el otro exclusivamente colores. Estos se utilizan para encontrar la presencia de características en las imágenes y según esto asignar *tags*.

El sistema de clasificación de categorías y atributos se basa en el uso de dos arquitecturas CNN, ResNet50 e Inception-v3, en dos modalidades de clasificación, multi-etiqueta y clasificación binaria. El sistema de color por su parte, se basa en el uso de una red de segmentación, DeepLab-v3, y el algoritmo k-means para realizar *clustering* sobre los píxeles de las prendas en las imágenes y clasificarlos según sus valores.

Las métricas usadas para evaluar los sistemas son las curvas *precision-recall* y ROC para los clasificadores de categorías y atributos, e IoU y *accuracy* para el sistema de color.

4.4. Retrieval

Para validar la capacidad de implementación del sistema propuesto en este trabajo en un catálogo de vestuario real, se lleva a cabo la tarea de filtrado de prendas de vestuario mediante la búsqueda y recuperación (*retrieval*) de imágenes, utilizando cada una de las

clases definidas como *tags* (exceptuando colores).

En primera instancia, se etiquetan las imágenes del *dataset* privado con cada uno de los modelos implementados, asignándole etiquetas de las clases de categorías y atributos definidas. El fin de asignar estas etiquetas es permitir la búsqueda de las imágenes midiendo la relevancia entre un término de búsqueda y el conjunto de etiquetas asociado a una imagen.

Para encontrar etiquetas en las imágenes, se determina el punto de operación de cada modelo según las curvas *precision-recall*, tomando el umbral de confianza más bajo que alcance sobre un 75 % de *precision* y un 50 % de *recall*. Se elige un valor alto de *precision*, puesto que en un contexto de *retrieval* de productos de *ecommerce* es importante mostrar productos relevantes, y además, porque todos los modelos alcanzan este valor en el conjunto de test.

Posteriormente, se conduce la búsqueda de artículos utilizando el nombre de cada clase de categorías y atributos definidos en este trabajo como *query*. Los resultados se ordenan según la relevancia entregada por la similitud coseno entre los vectores *tf-idf* de la *query* y los documentos, que en este caso corresponden al conjunto de *tags* de una imagen. El vector *tf-idf* de un documento (o de la *query*), corresponde a un vector de tantas dimensiones como términos (*tags*) existen en el universo de documentos (sets de *tags* de las imágenes), donde cada componente corresponde al valor de un término.

En vista de que una etiqueta puede asignarse a lo más una vez a cada imagen, se almacena la confianza de los modelos y se considera esta como la frecuencia de aparición del *tag* en la imagen (*term frequency*) en el *tf-idf*.

Ya que el sistema de clasificación de color no entrega una confianza en su predicción, no es posible ordenar los resultados de esta manera. Por esto, para evaluar este sistema, se clasifican las imágenes del conjunto de 100 imágenes por clase de color definido, y se mide el *accuracy* en la clasificación de estas.

El desempeño de la tarea de *retrieval* se determina según el *average precision* obtenido en las consultas de cada clase. Se limita el número de resultados a 15, debido a que es un número de resultados cercano a lo que entrega un catálogo de *ecommerce*, y garantiza que varias clases del *dataset* privado puedan ser analizadas.

4.4.1. Resumen

Se lleva a cabo un proceso de filtrado de imágenes de vestuario mediante la búsqueda de artículos con cada una de las clases definidas como *query*. Para esto, con los sistemas de clasificación implementados previamente, se asignan etiquetas a las imágenes del *dataset* privado. Estas etiquetas se utilizan para realizar búsquedas por texto con los *tags*, con el fin de evaluar el desempeño de los modelos en razón de los resultados que se obtienen con las etiquetas asignadas por ellos.

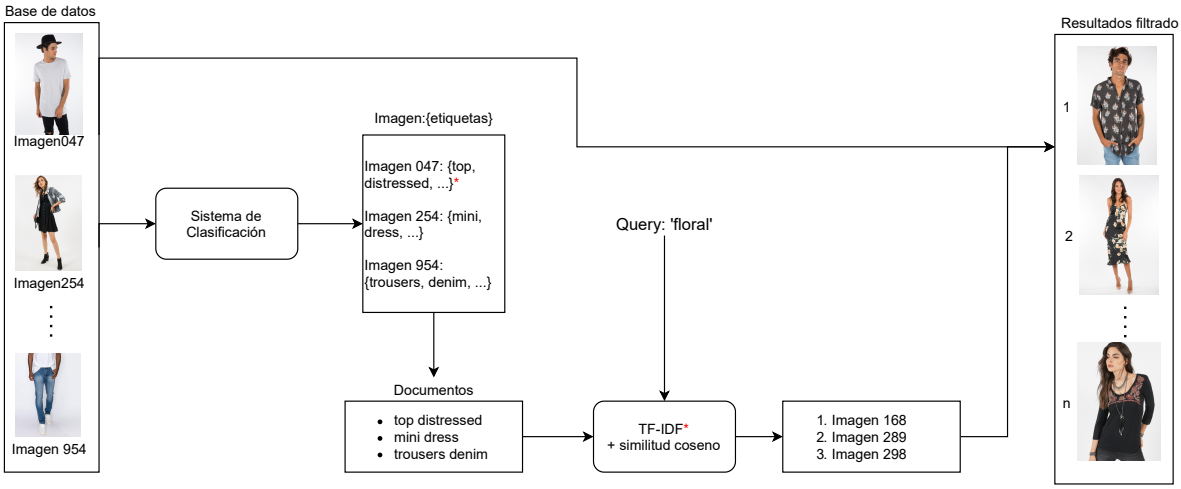


Figura 4.8: Metodología seguida para el *retrieval* de imágenes de vestuario realizando búsqueda basada en *tags* con tf-idf.

Capítulo 5

Resultados y análisis

En este capítulo se muestran los resultados obtenidos en cada etapa definida en la metodología de este trabajo. Junto con mostrar los resultados, se evalúa cada tarea según las métricas obtenidas y resultados visuales cuando sea pertinente.

5.1. Definición de clases

Se presentan las clases definidas y la cantidad de imágenes que compone cada una. Se muestran y separan en la definición de categorías y tipos de atributo. Además, se muestra la distribución de imágenes del *dataset* privado en las clases definidas luego de su etiquetado.

5.1.1. Atributos

Exploración

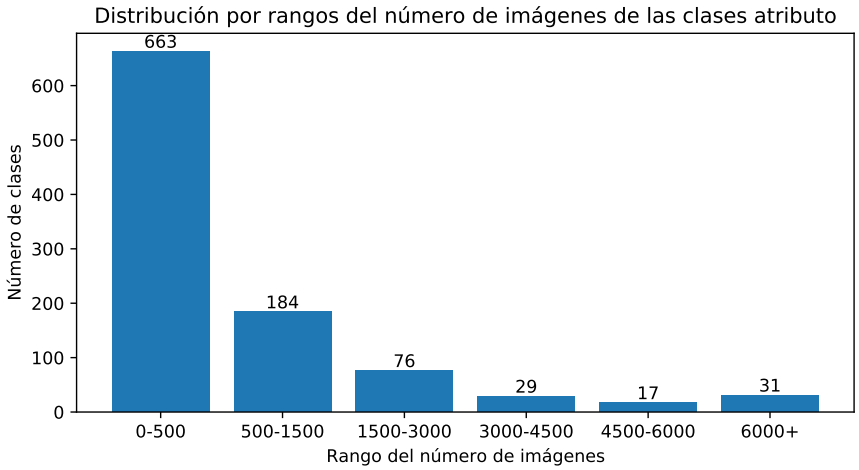


Figura 5.1: Distribución del número de clases según rangos de la cantidad de imágenes.

El subconjunto de clases de atributos de *DeepFashion* utilizado si bien contiene una gran cantidad de imágenes, se encuentra que solo una pequeña porción de esta resulta útil, dado que existe un gran desbalance de datos entre clases. En la Figura 5.1 se observa que la mayor parte de las clases poseen menos de 500 imágenes, con lo cual no es posible entrenar las redes descritas anteriormente. Este problema se compensa en parte con la gran cantidad de clases disponibles, donde aún eliminando las clases con menos de 3.000 ejemplos, se obtiene una cantidad razonable de clases (77) lo suficientemente balanceadas para un correcto entrenamiento, a pesar que solo existan 31 clases con más de 6.000 imágenes, lo que representa un margen aceptable [24].

Otro problema que se evidencia es la redundancia entre clases, existiendo casos de mucha similitud visual como se observa en la Figura 5.2.

(a) *dot* (b) *dots* (c) *dotted* (d) *polka dot*

Figura 5.2: Imágenes de cuatro clases distintas que representan prendas con puntos.

Filtrado

Mediante inspección visual de las imágenes restantes luego del proceso de filtrado de atributos de *DeepFashion*, se determina que medir la similitud semántica entre ellos no resulta un criterio apropiado para fusionarlos. Esto se atribuye a que el modelo Word2Vec tiende a asignar alta similitud a atributos asociados al mismo contexto, como es el caso de la vestimenta. Considerando además el hecho de que medir la similitud entre clases de cada tipo definido acota aún más el contexto (como por ejemplo cuando se consideran solo telas), encontrando similitudes altas entre todas las clases comparadas, llegando a fusionar clases que introducen ruido, como se observa en la Figura 5.3. Se determina así, que la similitud semántica en este caso es un mal indicador de similitud visual.

(a) Imagen de la clase *chiffon*

(b) Imagen de la clase *crochet*

Figura 5.3: Disimilitud entre atributos de dos imágenes de clase tipo *fabric*, fusionadas en la clase *chiffon* según criterio de similitud semántica.

A pesar de que el filtrado manual tiene el problema de ser intrínsecamente subjetivo, con este se puede garantizar una mayor distinción visual entre clases que el método con similitud semántica no puede. Con este proceso de filtrado quedan las clases por tipo mostradas en la Tabla 5.1 y la distribución de imágenes de cada clase que se muestra en la Figura 5.4.

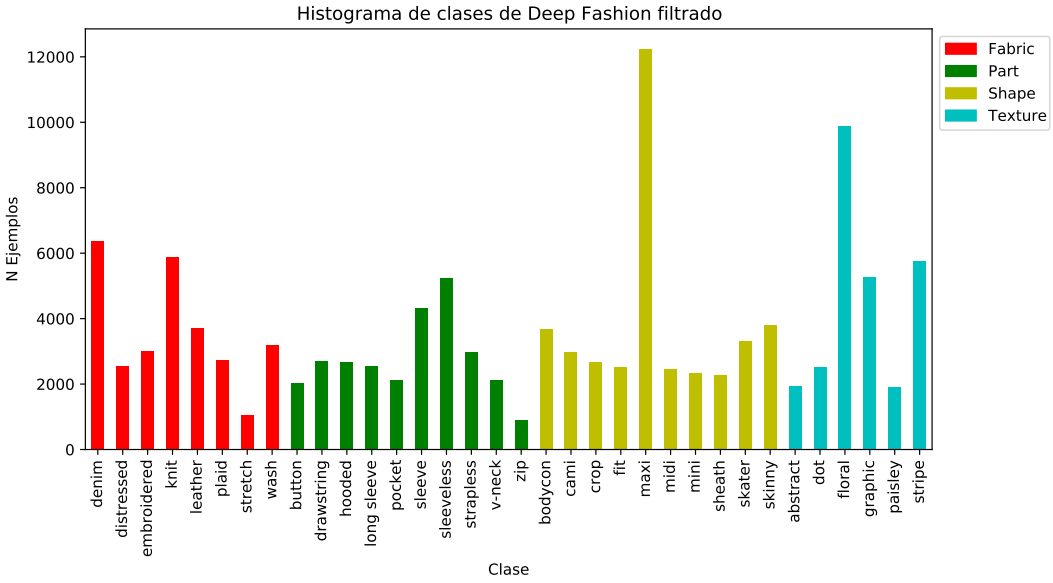


Figura 5.4: Distribución de instancias por clase de atributo de *DeepFashion*.

Tipo de clase	Clases	Número de clases
Texture	Abstract, dot, floral, graphic, paisley, stripe	6
Fabric	Denim, distressed, embroidered, knit, leather, plaid, stretch, wash	8
Shape	Bodycon, cami, crop, fit, maxi, midi, mini, sheath, skater, skinny	10
Part	Button, drawstring, hooded, long sleeve, zip sleeve, sleeveless, pocket, strapless, v-neck	10

Tabla 5.1: Clases finales de atributos por tipo.

5.1.2. Prendas

Exploración

La definición de las clases de categorías de prendas resulta más simple que la de atributos, dado que *DeepFashion 2* tiene una mejor calidad en la definición de estas. El problema que se observa con este *dataset* es que las diferencias entre las categorías se definen por detalles pequeños, como el largo de las mangas (*long sleeve dress y short sleeve dress*), y una en particular es muy general (*outwear*), representando cualquier tipo de prenda puesta sobre otra en la parte del torso. Además, se observa que la particularidad de las fotografías con fuente *shop*, más que ser fotografías usuales de catálogo, corresponden a fotografías tomadas en un ambiente profesional, como se muestra en la Figura 5.5, por lo que aún así presentan ruido por sus fondos y cambios de ángulos, escala e iluminación, lo que no se observa en imágenes de catálogo.

(a) (b) (c)

Figura 5.5: Imágenes anotadas con origen *shop* en *DeepFashion* 2.

Filtrado

La agrupación de clases de *DeepFashion* 2 realizada cubre las diferencias en detalles de bajo nivel de las prendas, y esta se muestra en la Tabla 5.2.

Clase	Compuesta por
Top	Short sleeve top, long sleeve top, vest, sling
Outwear	Short sleeve outwear, long sleeve outwear
Dress	Short sleeve dress, long sleeve dress, vest dress, sling dress
Shorts	Shorts
Trousers	Trousers
Skirt	Skirt

Tabla 5.2: Composición de clases filtradas con clases originales de prendas de *DeepFashion 2*.

5.1.3. Dataset privado

El *dataset* privado creado en este trabajo está compuesto por 882 imágenes de la tienda Bronzesnake, 833 de Ring of Fire y 630 de Umbrale, clasificadas en las clases de categorías y atributos definidas con los otros dos *dataset* La distribución de imágenes por clase se muestra en la Figura 5.6.

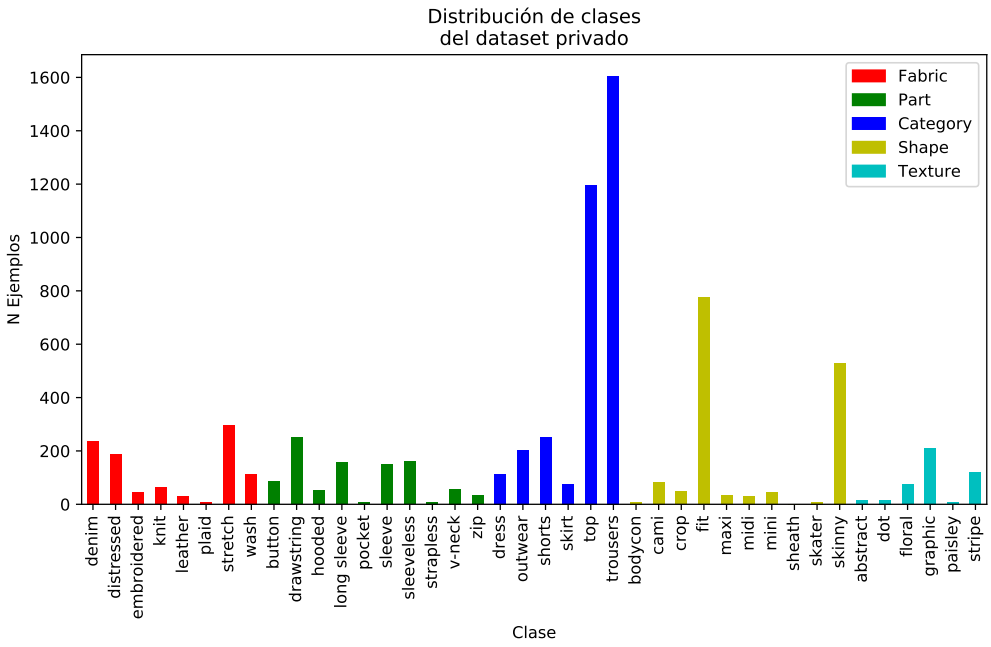


Figura 5.6: Distribución de instancias del *dataset* propio en cada clase definida.

Dado que este *dataset* es el utilizado para evaluar el sistema en un contexto real, es importante considerar el desbalance de clases que existe en él. Este desbalance favorece la recuperación de imágenes de las clases con más ejemplos, ya que mientras más imágenes ven los modelos de ellas, hay más posibilidad de que puedan etiquetarlas y ser efectivamente recuperadas.

5.2. Sistema de etiquetado

En la presente sección se muestran y analizan los resultados obtenidos por los métodos implementados para etiquetar imágenes que componen el sistema de etiquetado. Se presentan según sus partes en las subsecciones a continuación.

5.2.1. Categorías y atributos

Para evaluar los sistemas de clasificación de categorías y atributos se utilizan las curvas ROC y *precision-recall*. Dado que la implementación de los modelos multi-etiqueta se realiza con solo clases de un tipo, separado de clases de otro tipo, se contrasta el desempeño con los modelos binarios considerando estos mismo conjuntos. En la Tabla 5.3 se muestra el área bajo la curva de las curvas ROC y *precision-recall* obtenida por los modelos sobre su respectivo conjunto de test. RB, RM, IB e IM, se refieren a los modelos ResNet50 binario, ResNet50 multi-etiqueta, Inception-v3 binario e Inception-v3 multi-etiqueta; respectivamente.

Para simplificar el análisis sobre el desempeño de cada clase, se analizan por tipo a continuación.

Fabric

Respecto a las clases tipo *fabric* (referentes a la tela de las prendas), se observa en la Tabla 5.3, que se tiene un rendimiento similar para todas las clases con los cuatro modelos, destacando las clases *denim, distressed, wash* y *stretch* por su menor rendimiento sobre las demás.

En general, se atribuye el buen rendimiento en la clasificación de telas a que, primero, clasificando entre este universo de clases se tienen telas bien distinguibles entre sí, y segundo, a que varias de ellas además están definidas por una textura, que es la característica que las hace ser distinguibles visualmente, como tejidos (*knit*), bordados (*embroidered*) y tartán (*plaid*).

Considerando las clases *denim, distressed* y *wash*, observamos en la Figura 5.7 que existe una gran similitud entre ellas, lo que con las demás clases no ocurre. Que exista un desempeño más bajo en las clases más similares sugiere que los modelos aprenden correlaciones entre ellas con un efecto negativo, puesto que el estilo lavado (*wash*) y rasgado (*distressed*) de una tela suelen verse en telas de mezclilla (*denim*). Esta idea se refuerza al comparar el desempeño de los modelos binarios, los que si bien obtienen un menor desempeño en la clasificación de mezclilla (AUC-PR de 0.81 y 0.78 para ResNet50 e Inception-v3 multi-etiqueta respectivamente, contra 0.71 y 0.67 para los mismos modelos en modalidad de clasificación binaria), logran clasificar de mejor manera las clases *distressed* y *wash*.

Para *stretch*, que representa telas elásticas, se obtienen los peores resultados de clasificación, puesto que la característica que representa no es verificable mediante inspección visual, lo que refuerza la idea de que las telas con características que les den una textura son apropiadas para la clasificación mediante procesamiento de imágenes.

Tipo	**Clase**	**AUC ROC**				**AUC PR**			
		RB	**RM**	**IB**	**IM**	**RB**	**RM**	**IB**	**IM**
Fabric	Denim	0.79	**0.94**	0.75	0.93	0.71	**0.81**	0.67	0.78
	Distressed	0.79	**0.93**	0.79	0.92	0.67	0.58	0.7	0.55
	Embroidered	0.99	0.99	0.99	0.99	**0.99**	0.97	**0.99**	0.96
	Knit	0.98	**0.99**	0.98	0.98	**0.97**	0.96	**0.97**	0.96
	Leather	0.99	0.99	0.99	0.99	**0.99**	0.97	**0.99**	0.96
	Plaid	0.99	0.99	0.99	0.99	**0.99**	0.98	**0.99**	0.97
	Stretch	0.85	**0.94**	0.82	0.91	**0.78**	0.50	0.71	0.49
	Wash	0.85	**0.93**	0.84	**0.93**	**0.77**	0.61	**0.77**	0.58
Part	Button	0.94	**0.98**	0.94	**0.98**	**0.91**	0.86	**0.91**	0.85
	Drawstring	0.96	**0.99**	0.97	**0.99**	0.95	0.95	0.95	0.95
	Hooded	0.84	**0.98**	0.83	0.97	0.72	**0.92**	0.73	0.89
	Long Sleeve	0.94	**0.96**	0.92	**0.96**	**0.90**	0.68	0.87	0.73
	Pocket	0.94	**0.98**	0.92	0.97	**0.90**	0.89	0.87	0.85
	Sleeve	0.95	**0.97**	0.95	0.96	0.92	0.87	**0.93**	0.82
	Sleeveless	0.93	**0.98**	0.92	0.97	0.89	**0.94**	0.87	0.91
	Strapless	0.99	0.99	0.99	0.99	**0.99**	0.97	**0.99**	0.96
	V-Neck	**0.98**	0.97	**0.98**	0.97	0.97	0.78	**0.98**	0.74
	Zip	**0.99**	0.97	**0.99**	0.98	**0.99**	0.61	**0.99**	0.75
Shape	Bodycon	0.98	**0.99**	**0.99**	0.97	**0.98**	0.92	**0.99**	0.8
	Cami	0.97	**0.99**	0.98	0.95	0.96	0.92	**0.98**	0.78
	Crop	0.97	**0.99**	0.96	0.98	**0.96**	**0.96**	0.93	0.93
	Fit	0.84	**0.99**	0.81	0.97	0.78	**0.92**	0.74	0.79
	Maxi	0.99	0.99	0.99	0.99	**0.99**	**0.99**	**0.99**	0.97
	Midi	0.97	**0.99**	0.97	0.97	**0.97**	0.89	**0.97**	0.78
	Mini	0.92	**0.98**	0.90	0.97	0.86	**0.88**	0.83	0.77
	Sheath	**0.99**	**0.99**	**0.99**	0.98	**0.99**	0.93	**0.99**	0.85
	Skater	**0.99**	**0.99**	**0.99**	0.97	**0.99**	0.92	**0.99**	0.77
	Skinny	**0.99**	**0.99**	**0.99**	0.98	**0.99**	0.98	**0.99**	0.93
Texture	Abstract	0.87	**0.99**	0.88	0.98	0.82	**0.97**	0.86	0.91
	Dot	0.97	**0.99**	0.97	**0.99**	0.97	**0.99**	0.96	0.98
	Floral	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
	Graphic	0.98	**0.99**	0.98	**0.99**	0.98	**0.99**	0.98	**0.99**
	Paisley	0.99	0.99	0.99	0.99	**0.99**	0.97	**0.99**	0.93
	Stripe	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Category	Dress	0.89	**0.90**	0.89	0.88	0.75	0.78	0.74	0.75
	Outwear	0.89	**0.90**	**0.90**	0.87	0.83	**0.86**	0.84	0.82
	Shorts	0.92	**0.93**	**0.93**	0.91	0.85	**0.87**	0.86	0.82
	Skirt	0.84	**0.85**	**0.85**	0.84	0.61	**0.65**	0.63	0.62
	Top	0.85	**0.87**	0.86	0.85	0.77	**0.81**	0.79	0.78
	Trousers	0.95	0.95	**0.96**	0.94	0.89	**0.91**	**0.91**	0.90

Tabla 5.3: Área bajo las curvas ROC y precision-recall obtenido por los diferentes modelos para cada clase por tipo. En negrita los mejores resultados.

(a) Denim (b) Denim y wash (c) Denim y distressed

Figura 5.7: Imágenes pertenecientes a las clases *denim, wash* y *distressed* en *DeepFashion*.

Part

Para las clases tipo *part*, que corresponden al tipo referente a los detalles de más bajo nivel de las prendas, se observa un rendimiento similar en la clasificación de ellas, destacando dos casos particulares: las clases *zip* y *hooded*.

Para la clase *zip*, la con la menor cantidad de imágenes del tipo *part* (911), se observa que el mejor rendimiento se da con los modelos binarios (0.99 AUC-PR para ambos modelos binarios vs. 0.61 y 0.75 de ResNet50 e Inception-v3 multi-etiqueta respectivamente).

La clase *hooded* destaca por la diferencia entre el rendimiento de los modelos binarios y multi-etiqueta. Presentan con los últimos alrededor de un 20 % más de área bajo la curva *precision-recall* que los modelos binarios. Una explicación para esto, es que los modelos multi-etiqueta correlacionan características que aparecen juntas usualmente, como en este caso, polerones (*hoodies* en inglés) con la presencia de "gorro" (*hooded*). La diferencia con el caso anterior, es que la clase *hooded* tiene una cantidad mayor de ejemplos, lo que le permite a los modelos multi-etiqueta alcanzar un mayor aprendizaje y correlacionar de manera beneficiosa estas características. Con esto, se refuerza la importancia de tener datos balanceados para el entrenamiento de los modelos multi-etiqueta.

Shape

Para *shape* (formas de las prendas), según la Tabla 5.3, se observan rendimientos similares en la clasificación de sus clases. En general, los modelos de clasificación binaria obtienen el mejor desempeño en este tipo, lo que se le atribuye a que no existe una correlación de estas características visuales con otras, como ocurre con más frecuencias entre los tipos *fabric* y *part*, puesto que usualmente las prendas presentan solo una de las formas definidas, por lo que los modelos multi-etiqueta en este caso no son capaces de aprovechar una correlación existente. Los modelos binarios en este caso toman ventaja por el desbalance de datos existente, que como había sido mencionado, es más fácil solventar con estos modelos.

Texture

Las clases tipo *texture* son las de mejor desempeño de todos los tipos, puesto que su característica distintiva es de las más distinguibles visualmente. Además, al igual que con el tipo *shape*, estas clases suelen ser excluyentes entre sí en las prendas.

La clase que destaca en este caso es *abstract*, presentando alrededor de un 10 % menos de área bajo la curva con los modelos binarios que con su contraparte multi-etiqueta. Se atribuye este resultado a que la clase *abstract* es demasiado amplia en su definición, por lo que en este caso es beneficioso el uso de un modelo multi-etiqueta, ya que se refuerza la diferenciación entre esta clase y otras que podrían ser similares, como *floral* o *paisley*, teniendo una mayor cantidad de ejemplos negativos por clase que en el caso binario.

(a) Abstract (b) Floral (c) Paisley

Figura 5.8: Muestra de imágenes de baja resolución de clases tipo *texture* de *DeepFashion*.

Un aspecto relevante a considerar, observado en la Figura 5.8, es la baja resolución de las imágenes del *dataset* de donde se toman estas imágenes (*DeepFashion*), que quita detalle a las texturas y puede ser una causa del bajo rendimiento con detalles de muy bajo nivel, como ocurre con *paisley* (cachemir, diseño de lágrima).

Categorías

A pesar de que estas clases fueron definidas con el *dataset* más limpio (*DeepFashion* 2), y con datos balanceados, las clases tipo *garment* son las de peor rendimiento de clasificación con todos los modelos. Esto se explica por lo amplio del criterio empleado para definir las categorías y a lo distinto que pueden ser alguos estilos de las fotografías del *dataset*, los que como se aprecian en la Figura 5.9, tienen vistas muy distintas a lo tradicional de un catálogo de vestuario.

General

Observando los resultados de la Tabla 5.3 en conjunto con la distribución de datos de atributos de la Figura 5.1, destaca que para aquellas clases con una cantidad baja de ejemplos (*stretch, zip, abstract, paisley*), los modelos binarios suelan tener un mejor desempeño que su

(a) (b) (c)

Figura 5.9: Muestra de la variedad de vistas y escalas de prendas en imágenes de una misma clase (*top*) de clases de categorías de prendas.

contraparte multi-etiqueta. Esta diferencia se atribuye a la proporción de ejemplos positivos y negativos vistos por los modelos en su entrenamiento. Ya que para los modelos binarios es más fácil balancear la cantidad de datos, eligiendo menos ejemplos negativos en clases con pocas imágenes. Para los modelos multi-etiqueta por su parte, tener una clase con pocos ejemplos suele convertirse en un problema mayor, ya que deben estar presentes todas las otras clases, lo que suele llevar a una desproporción más grande aún de ejemplos positivos y negativos.

Entre modelos, se observa consistentemente que las diferencias en desempeño se deben a la distribución de los datos de entrenamiento, produciendo diferencias más notorias entre modelos binarios contra multi-etiqueta, más que entre una arquitectura y otra.

Comparando entre el área bajo las curvas ROC y *precision-recall*, observamos que según el área bajo la curva (AUC) ROC, muchas clases tienen un desempeño aparentemente idéntico, mientras que la *precision-recall* suele mostrar mayores diferencias. Esto sumado al hecho de que en el contexto de *retrieval* de artículos en el que se trabaja es más importante medir *precision* que la tasa de falsos positivos, hacen más valiosa a la curva *precision-recall* para la evaluación de estos métodos en nuestro contexto.

Basado en los resultados generales, se puede decir que en el caso donde se tiene un *dataset* balanceado, los modelos multi-etiqueta presentan mejor rendimiento. Además, entre las arquitecturas estudiadas, ResNet50 muestra ligeras pero consistentes mejoras respecto a Inception-v3 en la mayoría de las clases.

5.2.2. Colores

La evaluación del sistema clasificador de color comprende la evaluación de cada una de sus partes. El desempeño del modelo de segmentación se mide según el IoU, mostrado en la Tabla 5.4. La clasificación de color por su parte se evalúa mediante *accuracy*, mostrado en la Tabla 5.5.

Clase	**IoU** [%]
Fondo	93.39
Top	82.17
Bottom	77.71
Body	80.72
Promedio	83.50

Tabla 5.4: Desempeño por clase del modelo de segmentación.

Se observa en la Tabla 5.4 que la clase con mejor rendimiento en la segmentación es el fondo, lo que indica que el modelo tiene la capacidad de delimitar bien las siluetas de las prendas, y confunde más los límites entre ellas. Esta confusión marcando los límites de las prendas introduce ruido en la clasificación de color.

Color	**Accuracy** [%]
Blanco	86
Gris	76
Negro	100
Azul	90
Morado	93
Verde	66
Naranjo	45
Amarillo	56
Rojo	93
Promedio	78.33

Tabla 5.5: Accuracy de clasificación de color.

Respecto a la clasificación de color, el hecho de que se obtenga *accuracy* 100 % en la clasificación de algunas clases, indica que el criterio usado para verificar que un color está presente en la imagen es en parte capaz de eliminar el ruido de las imágenes. Sin embargo, los colores amarillo y naranjo se ven afectados por el ruido introducido por los casos donde aparece la piel de los o las modelos, ya que el color más cercano a los colores de piel son estos dos. En la Figura 5.10 se muestra un ejemplo de una imagen mal clasificada en la clase amarillo dado este fenómeno, donde se observa que la piel representa una cantidad considerable de píxeles en la imagen, cual es uno de los criterios utilizados para descartar ruido, que en este caso no logra cumplir su propósito.

(a) Imagen original. (b) Imagen cuantizada.

Figura 5.10: (a) Imagen mal clasificada en la clase amarillo debido al ruido introducido por la piel de la modelo. (b) Imagen cuantizada con los colores predominantes encontrados en la máscara indicada por el modelo de segmentación.

5.2.3. Tiempos de inferencia

La Tabla 5.6 muestra los tiempos de inferencia de una imagen para el modelo de segmentación DeepLab-v3, y las arquitecturas ResNet50 e Inception-v3 de clasificación.

Modelo	**Tiempo de inferencia [ms]**
DeepLab-v3-ResNet50	120
Inception-v3 multi-etiqueta	43
Inception-v3 binario	37
ResNet50 multi-etiqueta	32
ResNet50 binario	27

Tabla 5.6: Tiempo de inferencia para una imagen de los cinco modelos CNN utilizados.

El resultado más importante en cuanto el tiempo de inferencia de los modelos, es lo similares que son los modelos multi-etiqueta y los binarios. En vista de que los modelos multi-etiqueta pueden predecir varias etiquetas en una inferencia, su costo computacional por etiqueta puede ser alrededor de 10 veces menor que un modelo binario, dependiendo del número de clases implementado. Esto les da una ventaja pensando en su implementación en un contexto real.

Respecto al tiempo de inferencia del modelo de segmentación, se observa que este es alrededor de unas 4 veces mayor a los demás, lo que le da un gran costo computacional al sistema de clasificación de color. En vista de que la segmentación efectivamente permite extraer la zona de interés de las prendas en las imágenes, según los resultados de la clasificación de color, se propone la implementación de este modelo en otras tareas, como para clasificar otros atributos solo en las zonas de interés y evitar ruido, como se observa en el ejemplo mostrado en la Figura 5.11.

Figura 5.11: Imagen del *dataset* privado mal clasificada como *paisley* producto de los tatuajes del modelo.

5.3. Retrieval

Se presentan los resultados de búsqueda y recuperación de artículos del *dataset* privado con las etiquetas de categorías y atributos asignadas con los modelos entrenados. En la Tabla 5.7 se muestra el mAP obtenido en la recuperación de imágenes con etiquetas asignadas por cada modelo, y ordenaro por tipo de clase. Se consideran 15 resultados esperados para cada *query* (mAP@15), simulando la cantidad aproximada que muestra una página de resultados de un catálogo virtual, mostrados por tipo. La Tabla 5.8 muestra la Tabla **??** en detalle con cada clase. Se excluyen las clases *bodycon, cami, crop, dot, paisley, plaid, pocket, sheath, skater* y *strapless* del análisis de los resultados, puesto que no existe una cantidad suficiente de ejemplos de estas clases en el *dataset* privado (ver Figura 5.6) para poder recuperar el número de resultados establecidos (15). En ambas tablas, RB, RM, IB e IM se refieren a los modelos ResNet50 binario, ResNet50 multi-etiqueta, Inception-v3 binario e Inception-v3 multi-etiqueta.

Tipo	**RB**	**RM**	**IB**	**IM**
Fabric	0.26	**0.31**	0.22	0.18
Part	0.40	**0.46**	0.42	0.39
Shape	0.35	**0.43**	0.26	0.36
Texture	0.66	0.70	**0.72**	0.59
Category	0.47	**0.55**	0.45	0.43
General	0.41	**0.47**	0.39	0.37

Tabla 5.7: mAP@15 obtenido en la recuperación de artículos con etiquetas obtenidas con cada modelo por tipo de clase. En negrita los mejores resultados.

De la Tabla 5.8, destacan los resultados de la clase *graphic* por su rendimiento perfecto. Observando las imágenes de entrenamiento y del *dataset* privado, se observa que el estilo de estas en ambos *dataset* es muy similar, y además representan una característica que resulta

Tipo	Clase	RB	RM	IB	IM
Fabric	Denim	0.56	0.58	**0.64**	0.31
	Distressed	0.08	**0.18**	0.09	0.15
	Embroidered	**0.25**	0.21	0.18	0.22
	Knit	0.34	**0.46**	0.24	0.09
	Leather	0.12	**0.20**	0.06	0.07
	Stretch	**0.30**	0.29	0.14	0.17
	Wash	0.19	**0.26**	0.21	0.25
Part	Button	0.38	**0.58**	0.43	0.22
	Drawstring	0.36	0.12	**0.43**	0.37
	Hooded	0.43	0.49	0.56	**0.63**
	Long sleeve	0.33	**0.39**	0.23	**0.39**
	Sleeve	0.51	**0.59**	0.52	0.49
	Sleeveless	0.66	**0.80**	0.67	0.57
	V-Neck	0.27	**0.46**	0.29	0.25
	Zip	**0.23**	0.22	0.19	0.17
Shape	Fit	0.10	**0.19**	0.08	0.16
	Maxi	0.38	**0.52**	0.37	0.40
	Midi	0.46	**0.55**	0.28	0.47
	Mini	0.23	0.38	0.11	**0.44**
	Skinny	**0.57**	0.54	0.45	0.31
Texture	Abstract	0.53	**0.66**	0.64	0.50
	Floral	0.50	0.55	**0.67**	0.38
	Graphic	1.00	1.00	1.00	1.00
	Stripe	**0.60**	**0.60**	0.57	0.46
Category	Dress	0.46	**0.60**	0.39	0.37
	Outwear	**0.12**	0.04	0.11	0.05
	Shorts	0.43	**0.60**	0.37	0.51
	Skirt	0.60	**0.68**	0.25	0.39
	Top	0.76	0.71	**0.85**	0.58
	Trousers	0.44	0.69	**0.70**	0.67

Tabla 5.8: *Average precision* obtenido en la búsqueda y recuperación de 15 imágenes (AP@15) del *dataset* privado con etiquetas generadas por cada modelo. En negrita los mejores resultados por clase.

fácilmente identificable, dado a la particularidad que presentan los estampados. En la Figura 5.12 se muestra una imágenes de *DeepFashion* y del *dataset* privado respectivamente.

Relativo a telas (*fabric*), destacan las clases *leather, wash* y *distressed* por su mal rendimiento en la recuperación. Al igual que como se evidencia en la clasificación, las clases conflictivas entre sí muestran mal desempeño cuando se estudian por separado, lo que ocurre con los modelos binarios. Estas características al encontrarse relacionadas con otras, resultan fácilmente confundibles si no se entrenan en conjunto. En la Figura 5.13 se muestran los primeros resultados de la búsqueda *leather* con etiquetas de cada modelo, donde se hace evidente que los modelos relacionan la presencia de cuero con la presencia de color negro.

(a) DeepFashion (b) *Dataset* privado

Figura 5.12: Imágenes de ejemplo de clase *graphic* de DeepFashion y el *dataset* privado.

(a) (b) (c) (d)

Figura 5.13: Primer resultado del *retrieval* de imágenes del *dataset* privado filtrando por atributo *leather* (cuero) con etiquetas asignadas por los modelos (a) ResNet50 binario, (b) ResNet50 multi-etiqueta, (c) Inception-v3 binario y (d) Inception-v3 multi-etiqueta.

Un resultado llamativo es el buen rendimiento en la recuperación de imágenes con los *tags* de categorías, obteniendo con ellas un mAP más alto que otros tipos, a pesar de obtener el AUC de ambas curvas más bajo en el conjunto de test. Una explicación para esto, es que como ha sido mostrado, el ambiente de las fotografías de *DeepFashion* 2 (con las que se componen las clases de categorías) es variado en ángulos e iluminación, lo que dificulta la clasificación en el conjunto de test, sin embargo, esto no sucede en el *dataset* privado, ya que las fotografías de catálogo se realizan en un ambiente controlado y sin ruido de fondo.

Sumado a lo anterior, observando en la distribución de clases del *dataset* privado mostrada en la Figura 5.6, se evidencia que las categorías definidas son las más presentes entre los catálogos que componen el *dataset* privado. Como había sido mencionado, esto facilita que los modelos puedan encontrar etiquetas que permitan la recuperación de artículos, ya que tienen más oportunidades de clasificar correctamente uno de estos ejemplos. Con este resultado, se refuerza la importancia del *precision* sobre otras métricas, ya que un *precision* alto en este caso permite obtener los resultados relevantes antes que los irrelevantes, mejorando el desempeño del sistema de recuperación.

De manera general, se tiene que ResNet50 multi-etiqueta es el mejor modelo para el etiquetado de imágenes. Esto se explica por el hecho de que este comparte información de todas las clases, lo que resulta beneficioso en el proceso de entrenamiento debido a que la red puede diferenciar las clases conflictivas al tener estos ejemplos de manera simultánea. Sumado a esto, los modelos multi-etiqueta son capaces de obtener los resultados de clasificación en una única inferencia, entregando los resultados para todos los atributos al mismo tiempo, al contrario de los modelos binarios que necesitan de tantas inferencias por imagen como atributos existan.

Se confirma con el uso del *dataset* privado, que el desempeño de los modelos entrenados con los *dataset DeepFashion* y *DeepFashion* 2, varía considerablemente en algunas clases al cambiar el contexto a imágenes de catálogo. Mientras las curvas *precision-recall* y ROC muestran clasificación casi perfecta en algunas clases, los resultados de *retrieval* no se acercan a esto. Un ejemplo de esto es la clase *leather*, la cual alcanza 0.99 de área bajo la curva *precision-recall* en la clasificación (ver Tabla 5.3), pero que sin embargo alcanza como máximo un *average precision* de 0.2 en la recuperación de artículos del *dataset* privado (ver Tabla 5.8), puesto que en este último existen prendas con materiales que resultan confundibles con el cuero, como chaquetas de tela y parkas de color negro.

Una causa atribuida a lo anterior, es que el universo de imágenes de los *dataset* públicos utilizados resulta menos complejo que el del *dataset* privado, puesto que en el último existe un universo más amplio de características visuales que en los conjuntos de test no son observados. Esta limitación del universo de características que se observan en los conjuntos de test beneficia el desempeño mostrado por los modelos en su clasificación, lo que resulta engañoso al implementarlos en otro ambiente.

Un aspecto importante a considerar de la recuperación de imágenes, es el criterio de relevancia de los resultados que entrega la similitud coseno. Debido a que los vectores *tf-idf* son normalizados en este cálculo, la frecuencia de las etiquetas (proporcional a la confianza de las etiquetas asignadas por los modelos según la metodología propuesta) resulta menos relevante respecto al conjunto de etiquetas que son asignadas a las imágenes, ya que esto le da la orientación al vector por sobre la magnitud. Esto resulta beneficioso cuando los modelos asignan etiquetas correctamente con baja confianza, pero perjudicial cuando hay muchas etiquetas presentes en una imagen, puesto que le quitan relevancia a la característica buscada, orientando el vector *tf-idf* en otro sentido, que disminuye la relevancia de una imagen respecto a la búsqueda. En vista de esto, se propone la mejora del método de recuperación como trabajo futuro.

Capítulo 6

Conclusiones

En el presente trabajo se plantea mejorar la calidad de los catálogos virtuales de tiendas de vestuario, implementando filtros basados en características visuales de bajo nivel que permitan buscar sus productos de manera más detallada de lo que estas herramientas permiten actualmente.

El sistema de etiquetado planteado se basa en el uso de métodos de procesamiento de imágenes que permitan la asignación automática de *tags* para realizar búsquedas por texto de artículos, utilizando estos *tags* para medir la relevancia de ellos. Se estudia el desempeño de dos arquitecturas CNN en la clasificación imágenes de prendas de vestuario, ResNet50 e Inception-v3 en dos modalidades de clasificación distintas: clasificación binaria y clasificación multi-etiqueta. Además se implementa y evalúa un sistema de clasificación de color de prendas basado en el uso de un modelo de segmentación y un algoritmo de *clustering* sobre los píxeles que componen cada prenda.

Se evalúa el desempeño del sistema desarrollado mediante la recuperación de imágenes de un *dataset* privado compuesto por fotografías de tres tiendas reales de *ecommerce* de vestuario, con el fin de validar la utilidad del sistema en un contexto real. Respecto a los resultados de este proceso, se pueden concluir los siguientes puntos:

- Las texturas y formas, junto con algunas particularidades de bajo nivel bien definidas como el largo de mangas, son las características más reconocibles en las prendas y por ende las más fáciles de implementar como filtros. Esto, dada su distinguibilidad visual, que permite que los modelos aprendan bien las características que las representan y logren introducir poco ruido al etiquetar. Dado a que se identifica que los catálogos virtuales carecen de estos filtros, estas características resultan las de mayor valor para su implementación en un catálogo real.
- Algunas características no resultan apropiadas para ser clasificadas mediante métodos de procesamiento de imágenes. Clases como telas, donde sus texturas son poco reconocibles en imágenes pueden introducir ruido, especialmente cuando se encuentran correlacionadas con otras características, que si no son estudiadas a fondo por los modelos, son fáciles de confundir con otras.
- Las clases denominadas categorías (*category*), si bien presentan buen desempeño en

el *retrieval* de imágenes de catálogo, se determina que no agregan mucho valor a las tiendas, puesto que estas últimas ya disponen de estas categorías para filtrar productos, y en algunos casos, más detalladas.

- Los modelos de clasificación multi-etiqueta presentan un mejor desempeño en general que sus contrapartes binarias. Sin embargo, los últimos pueden llegar a obtener un mejor desempeño que los multi-etiqueta en clases bien definidas y con pocos datos.
- La clasificación de color muestra buenos resultados, demostrando que el sistema propuesto tiene valor para un catálogo real. Sin embargo, la red de segmentación usada le confiere un gran costo computacional al sistema, volviéndolo infactible para aplicaciones que prioricen el tiempo de respuesta del sistema.

La recomendación del autor es utilizar el sistema de *auto tagging* diseñado para asignar etiquetas a imágenes de manera *offline*, utilizando clases referentes a texturas y formas, las que como ha sido mencionado, se identifican como las que pueden agregar mayor valor para ser agregadas a catálogos virtuales. Con estas clases, se puede esperar un mAP entre un 47 % y un 60 % en la recuperación de al menos una página de resultados. Esto confirma que es posible implementar filtros en catálogos virtuales de vestuario mediante *tags* generados automáticamente por el sistema propuesto.

De esta manera, se da por cumplido el objetivo general de este trabajo, que consiste en la implementación de un sistema de *auto tagging* de imágenes de vestuario que permite aumentar la cantidad de filtros disponibles actualmente en los portales de *ecommerce* de este rubro.

6.1. Trabajo futuro

En vista de que se observa que en algunas imágenes el ruido presente puede influenciar negativamente el desempeño de los modelos de clasificación, y de que se logra obtener un buen desempeño de segmentación de prendas, se propone el uso conjunto de métodos de segmentación con clasificación de características visuales para considerar solo las zonas de las imágenes que resultan relevantes y evitar falsos positivos en la asignación de etiquetas.

Como se verifica en otros resultados de *retrieval* de imágenes, existe confusión entre clases de características visuales que se refieren a distintos tipos de atributos que en este trabajo fueron estudiados como conjuntos de clases disjuntos (tipos de clases). En vista de esto, se propone la implementación de modelos multi-etiqueta que consideren una gran cantidad de atributos, ya que como se expone, estos son capaces de reconocer correlaciones entre clases, y que por ende les permiten diferenciar mejor entre ellas; con un costo computacional muy similar a modelos de clasificación binaria especializados para cada característica visual.

Se propone un método más sofisticado que el usado para la recuperación de imágenes. Como se expone, existen correlaciones entre características que se pueden aprovechar tanto desde el dominio visual como semántico, con lo que pueden mejorar los resultados de clasificación y recuperación.

Por último, se evidencia la necesidad de una base de datos con información sobre atributos detallados de las prendas con una mejor calidad que los *datasets* usados en este trabajo, los

que presentan muchas clases redundantes y con ejemplos insuficientes. En razón de esto, se propone la creación de una base de datos de mejor calidad y cantidad de imágenes orientada a un ambiente de catálogo virtual de vestuario.

Bibliografía

[1] Abrar H. Abdulnabi, Gang Wang, Jiwen Lu, and Kui Jia. Multi-task cnn model for attribute prediction. *IEEE Transactions on Multimedia*, 17(11):1949–1959, Nov 2015.

[2] Y. Ben Salem and S. Nasri. Texture classification of woven fabric based on a glcm method and using multiclass support vector machine. In *2009 6th International Multi-Conference on Systems, Signals and Devices*, pages 1–8, 2009.

[3] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017.

[4] Samuel Dodge and Lina Karam. A study and comparison of human and deep learning recognition performance under visual distortions, 2017.

[5] Yuying Ge, Ruimao Zhang, Lingyun Wu, Xiaogang Wang, Xiaoou Tang, and Ping Luo. A versatile benchmark for detection, pose estimation, segmentation and re-identification of clothing images. *CVPR*, 2019.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[7] Junshi Huang, Rogerio S. Feris, Qiang Chen, and Shuicheng Yan. Cross-domain image retrieval with a dual attribute-aware ranking network. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[8] Naoto Inoue, Edgar Simo-Serra, Toshihiko Yamasaki, and Hiroshi Ishikawa. Multi-label fashion image classification with minimal human supervision. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[9] Chris Kawatsu, Frank Koss, Andy Gillies, Aaron Zhao, Jacob Crossman, Benjamin Purman, Dave Stone, and Dawn Dahn. Gesture recognition for robotic control using deep learning. 08 2017.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[11] George Lever. Tendencias del Comercio Electrónico en Chile. https://www.ecommerceccs.cl/tendencias-del-comercio-electronico-en-chile-octubre-2019/, 2019.

[12] George Lever. Fashion Online: Análisis del impacto de la pandemia y desafíos de la moda online. https://www.ecommerceccs.cl/fashion-online-analisis-de-impacto-de-pandemia-y-desafios-de-la-moda-online/, 2020.

[13] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[14] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038, 2014.

[15] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, United States, June 2014.

[16] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[17] A. Saha, M. Nawhal, M. M. Khapra, and V. C. Raykar. Learning disentangled multimodal representations for the fashion domain. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 557–566, March 2018.

[18] Guillem Sanz. La "nueva normalidad" del e-commerce: de la subsistencia a un nuevo estilo de vida. Mayo 2020.

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

[20] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[21] Brian Tomasik, Phyo Thiha, and Douglas Turnbull. Tagging products using image classification. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 792–793, New York, NY, USA, 2009. ACM.

[22] L. Wu, R. Jin, and A. K. Jain. Tag completion for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):716–727, 2013.

[23] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from mas-

sive noisy labeled data for image classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[24] Roshanak Zakizadeh, Michele Sasdelli, Yu Qian, and Eduard Vazquez. Improving the annotation of deepfashion images for fine-grained attribute recognition, 07 2018.

[25] Xiaojin Zhugy and Zoubin GhahramanigyH. Learning from labeled and unlabeled data with label propagation. 2002.