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

BÚSQUEDA TEXTO-VISUAL DE PRENDAS DE VESTIR

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELÉCTRICO

NÉSTOR IGNACIO ÁLVAREZ MUÑOZ

PROFESOR GUÍA:
JUAN BARRIOS NÚÑEZ

MIEMBROS DE LA COMISIÓN:
FRANCISCO RIVERA SERRANO
RICARDO FONSECA ROMERO

Este trabajo ha sido parcialmente financiado por Impresee

SANTIAGO DE CHILE
2022

Resumen

El presente documento expone el trabajo realizado en el desarrollo de un buscador texto-visual, que combine texto e imágenes de prendas de vestir utilizando machine learning para mejorar la búsqueda tradicional dentro del e-commerce. La búsqueda visual consiste en comparar una foto de consulta de un usuario con las fotos de los productos de una tienda y mostrar los productos más parecidos visualmente a la foto de consulta. Para realizar esto se utilizan descriptores, los cuales consisten en una representación vectorial de la información más relevante de la imagen.

En las tiendas online de moda usualmente las fotos de los productos aparecen modeladas por una persona que también viste otras prendas. Cuando aparece más de un producto en una misma foto la búsqueda visual se puede confundir con cuál es el producto principal, lo cual ocasiona que el desempeño de la búsqueda por similitud disminuya considerablemente.

Para solucionar esto se exploró la utilización de técnicas de machine learning para crear un embedding o espacio descriptores que combine tanto información de texto como de imagen. Creado un espacio que combine la descripción del producto junto con las imágenes de los productos en la imagen es posible identificar el producto principal en la imagen y resolver el problema de los múltiples productos por imagen. Además de resolver la búsqueda visual este espacio combinado permite que la recomendación de productos similares considere tanto el texto como la imagen de un producto.

Respecto a los resultados se concluye que el 90 % de las veces es posible reconocer cuál es el producto principal que se esta vendiendo en una foto con múltiples productos. Además la recomendación de productos similares mejora entre un 15 % a 50 % (dependiendo de la tienda evaluada) cuando se identifica en una primera etapa el producto que se está vendiendo y luego se usa una medida de similitud en el espacio texto-visual.

Para el Señor Balto

Agradecimientos

juan.cl por su paciencia infinita

Tabla de Contenido

1. Introducción **1**

1.1. Contexto 1

1.2. Problema 3

1.3. Solución propuesta 4

1.4. Objetivos 4

1.4.1. Objetivo general 4

1.4.2. Objetivos Específicos 4

2. Marco teórico y estado del arte **6**

2.1. Information Retrieval 6

2.2. Red neuronal 7

2.3. Red neuronal convolucional 8

2.3.1. MobileNet 9

2.3.2. Transfer learning 10

2.4. Object Detection 10

2.4.1. Yolov3 11

2.5. Red neuronal recurrente 11

2.5.1. Redes LSTM 12

2.6. Word Embedding 12

2.7. Redes siamesas 13

2.8. Triplet loss 13

2.9. Triplet mining 14
2.10. Métricas de evaluación IR 15
2.10.1. Recall at k 15
2.10.2. Precision at k 16
2.11. Estado del arte 16

3. Descripción solución **17**
3.1. Descripción general 17
3.2. Implementación 19
3.2.1. Dataset 19
3.2.2. Modelo 22
3.2.3. Entrenamiento 25
3.3. Diseño de experimentos y evaluación 27
3.4. Experimentos Tienda 1 27
3.4.1. Identificar el producto que se vende en una imagen de la Tienda 1 . . 28
3.4.2. Mejora de las búsquedas por similitud en la Tienda 1 29
3.5. Experimentos Tienda 2 29
3.5.1. Obtención automática de bounding boxes 30
3.5.2. Identificar el producto que se vende en una imagen de la Tienda 2 . . 30
3.5.3. Encontrar elementos similares en la Tienda 2 30

4. Resultados y análisis **32**
4.1. Experimentos Tienda 1 32
4.1.1. Identificación del producto en venta Tienda 1 32
4.1.2. Mejora similitudes Tienda 1 35
4.2. Experimentos Tienda 2 35
4.2.1. Obtención automática de bounding boxes Tienda 2 35
4.2.2. Identificación del producto en venta Tienda 2 37

4.2.3. Mejora similitudes Tienda 2 38

Conclusión **40**

Bibliografía **42**

Índice de Tablas

4.1. Desempeño del modelo en la identificación del producto en venta según la cantidad de productos y elementos. Baseline representa como se desempeñaría una elección aleatoria. 32

4.2. Resultados diferentes similitudes Tienda 1 35

4.3. Tabla resultados similitudes Tienda 2 38

Índice de Ilustraciones

1.1. Ejemplo de búsqueda de elementos externos a la tienda. En este caso, el cliente sube una imagen de una prenda de vestir a una plataforma de e-commerce y se le presenta el elemento más similar dentro de su catálogo. 2

1.2. Ejemplo de búsqueda de elementos similares dentro de la tienda. 2

1.3. Ejemplo en el cual falla la búsqueda visual de Impresee por no considerar la información textual en la búsqueda visual: si un cliente busca un pantalón, no es aceptable que los primeros resultados pertenezcan a un sweater y a una polera. 3

2.1. Information Retrieval aplicado a la búsqueda por imágenes 6

2.2. Red neuronal multicapa. Se aprecian las capas de entrada y salida del modelo, y las capas ocultas. 7

2.3. Ejemplo de filtro clásico utilizado en el procesamiento de imágenes. 8

2.4. Ejemplo de filtros aprendidos por una red neuronal convolucional. 9

2.5. Comparación de la precisión y cantidad de parámetros de MobileNet en comparación a otro tipo de arquitecturas en la tarea de clasificación. 10

2.6. Uso de bounding boxes para la detección de objetos de interés. 11

2.7. Redes neuronales recurrentes. 12

2.8. Redes LSTM. 12

2.9. Triplet Loss. 14

2.10. Triplet mining: clasificación del tipo de tripletas según la distancia del anchor al positivo y la distancia del anchor al negativo. 15

3.1. Error encontrado al utilizar un detector. El texto descriptor de los artículos no se corresponde con las imágenes correspondientes a las prendas. 18

3.2. Al aportar más información, la predicción de la prenda se vuelve más precisa. 19

3.3. Polera tejida terracota. 19

3.4. En la imagen el producto de interés se titula “Polera terracota”, por lo cual el rectángulo rojo (que señala la prenda que se está vendiendo) se encuentra encerrando a la polera. Los demás rectángulos encierran el resto de prendas de vestir presentes (pantalones, zapatos). 20

3.5. En este ejemplo, una imagen se subdivide en 4 productos, cada uno acompañado de un texto más un escalar binario que señala si el recorte corresponde al texto del artículo (0) o no lo hace (1). 21

3.6. Reescalado de imágenes a 96 x 96 píxeles. En la imagen de la izquierda se mantuvo el aspect ratio mediante padding. En la imagen de la derecha éste no se mantuvo, causando que los pantalones pasaran a tener más similitud con la clase “short”. 21

3.7. Diagrama de alto nivel del modelo propuesto para el espacio texto-visual. . . 22

3.8. Loss de tipo “binary cross entropy”. Dependiendo de si el texto corresponde o no a la imagen, se tienen las categorías 0 y 1, respectivamente. 26

3.9. Mecanismo para seleccionar el bounding box correspondiente al título del producto. 28

4.1. Anotaciones simulando detector perfecto. 33

4.2. Ejemplo correcto selección bounding box 2 productos. 33

4.3. Error de tipo 1 Tienda 1. 34

4.4. Error de tipo 2 Tienda 1. 34

4.5. Similitudes tienda 1: En la primera fila se presentan los resultados de la similitud visual, en la segunda fila se muestra los resultados de la similitud texto-visual, en la tercera fila se presentan los resultados de la similitud visual escogiendo una detección representante por imagen y en la ultima final se presentan los resultados de la similitud texto-visual escogiendo una detección representante por imagen 36

4.6. Ejemplo 1 detección YOLOv3 Tienda 2 37

4.7. Selección bounding box correcto caso 1 Tienda test 38

4.8. Búsqueda por similitud tienda 2: En la fila de arriba se observa las similitudes obtenidas al buscar el producto mediante descriptores visuales cuya imagen se encuentra en la tercera fila y su texto es çhaleco sole chaleco sin mangas medidas más abajo en descripción". En la fila de al medio se observan las similitudes obtenidas por descriptores texto-visuales 39

Capítulo 1

Introducción

1.1. Contexto

El e-commerce consiste en la compra y venta de bienes y servicios a través de Internet, donde para poder ofrecer estos bienes y servicios las tiendas usualmente implementan un sitio web en el cual los consumidores puedan comprar dichos productos.

El e-commerce ha presentado una creciente tendencia de uso durante los últimos años, con un notorio aumento durante la pandemia por SARS-CoV-2 asociado a la asepsia que implica en contraste con la compra presencial. Es por lo anterior que muchos rubros de distinta naturaleza (entre ellos, casas de retail) han decidido optar o se han visto obligados a utilizar el e-commerce como su principal forma de venta.

Cada plataforma de e-commerce consta de:

- Productos: Artículos propiamente tales que están en venta.
- Catálogo: Conjunto de productos que se encuentran a la venta en la tienda.
- Ficha de producto: Información que describe el producto, debido a que en el e-commerce no es posible mostrar la prenda físicamente se quiere que el producto quede lo mejor descrito mediante esta información.

En la ficha de productos es usual encontrar tanto texto e imágenes. En lo que respecta al texto, usualmente hay una breve descripción de las mejores cualidades del producto, mientras que en lo que respecta a las imágenes corresponden a diferentes vistas dentro de distintos contextos.

Entre las tareas que debe ejecutar de manera correcta una plataforma de e-commerce se encuentran:

1. Búsqueda de elementos externos a la tienda: Dado un potencial cliente que se encuentra interesado en un artículo (por ejemplo una polera) el cual vio en alguna red social,

película, etc. En caso de que el potencial cliente quisiera adquirir la misma prenda o una muy similar existen dos grandes enfoques. En el enfoque tradicional la persona mediante texto buscaría la polera describiéndola textualmente, sin embargo, describir verbalmente una prenda de vestir puede ser muy difícil. No obstante existen alternativas tales como la búsqueda visual, mediante la cual el cliente puede subir una foto de la prenda en cuestión a la plataforma de e-commerce, encontrar la prenda que era de su interés o una muy similar, y realizar la compra, situación que se ilustra en la Figura 1.1.

Figura 1.1: Ejemplo de búsqueda de elementos externos a la tienda. En este caso, el cliente sube una imagen de una prenda de vestir a una plataforma de e-commerce y se le presenta el elemento más similar dentro de su catálogo.

2. Búsqueda de elementos similares dentro de la tienda: Dado que un potencial consumidor esta observando un articulo dentro de una tienda de ecommerce es conveniente mostrarle productos similares dentro del catalogo los cuales podrían ser también de interés para este, lo cual conlleva a aumentar la probabilidad de que se concrete una venta. (Figura 1.2).

Figura 1.2: Ejemplo de búsqueda de elementos similares dentro de la tienda.

Dada la gran cantidad de sitios de e-commerce existentes es necesario que los servicios ofrecidos por las tiendas sean de uso intuitivo y agradable para el consumidor, pues de lo contrario es muy fácil que cambie de tienda, conllevando a que la tienda pierda una potencial venta.

Este trabajo se desarrolló para Impresee, la cual es una empresa dedicada a mejorar el e-commerce utilizando inteligencia artificial. Actualmente ofrece servicios de búsqueda por texto, imágenes, dibujos en blanco y negro y a color, sistemas de recomendación y analíticas de e-commerce.

1.2. Problema

Las tareas de búsqueda y recomendación basada en similitud implementadas por Impresee actúan netamente en base a descriptores de texto o a descriptores visuales. Para la obtención de estos atributos visuales, el algoritmo de Impresee no utiliza toda la imagen correspondiente al producto si no que aplica un detector que diferencia sus regiones de interés, lo que permite obtener características que representen a la prenda.

La empresa ha enfrentado una serie de desafíos particularmente en dos tareas: la búsqueda visual de elementos externos a la tienda y la búsqueda de elementos similares dentro de la tienda.

Con respecto a la búsqueda visual, ocurre frecuentemente que al buscar una prenda de vestir (en este caso el pantalón de la Figura 1.3) se muestran productos visualmente parecidos dentro de la base de batos que no corresponden a la prenda buscada, a veces incluso pertenecientes a categorías completamente distintas; este error ocurre por no considerar la información textual dentro de la búsqueda visual. En el e-commerce dedicado a la venta de vestuario es común que el producto principal (que se está vendiendo y coincide con el texto de búsqueda) aparezca acompañado de otros productos que le otorgan contexto; sin embargo, estos productos muchas veces no se encuentran disponibles en la tienda, ya sea por falta de existencias o porque la tienda no los ofrece dentro de su catálogo. Lo anterior merma en gran manera la calidad de la búsqueda visual de prendas de vestir ofrecida por Impresee.

Figura 1.3: Ejemplo en el cual falla la búsqueda visual de Impresee por no considerar la información textual en la búsqueda visual: si un cliente busca un pantalón, no es aceptable que los primeros resultados pertenezcan a un sweater y a una polera.

Con respecto a la búsqueda de productos similares dentro de la tienda, actualmente Impresee calcula esta semejanza basada en texto, sin embargo, esto resulta poco flexible al depender netamente del texto que haya escrito la persona encargada de darle un nombre

al producto, sin considerar la forma, color o textura de la prenda (pues estos son atributos visuales que son difíciles de captar mediante la utilización de texto).

1.3. Solución propuesta

Gracias a los avances en el campo del deep learning, actualmente es posible relacionar información de dominios distintos como son el texto y las imágenes. Una de las formas más frecuentemente utilizadas es mediante el uso de un espacio común donde se mantienen relaciones métricas, es decir, las representaciones de un mismo objeto se encuentren más cerca entre sí que las representaciones de otros objetos.

Los productos de un catálogo cuentan con una inherente naturaleza texto-visual, es decir, no es posible comprender a cabalidad qué se está vendiendo en un catálogo de e-commerce a no ser que se cuente con el texto e imagen del producto. Para solucionar los dos problemas anteriores se creó un espacio vectorial de descriptores que se denominó "texto-visual", el que contiene descriptores que representan el texto, la imagen y la combinación de ambos.

Con este espacio texto-visual, que contiene relaciones métricas de interés, es posible solucionar el problema de la búsqueda visual de productos similares tanto dentro del catálogo como fuera de éste (por ejemplo subir una imagen de un producto a un sitio de ecommerce y encontrar los elementos más similares). Para cada imagen será posible saber cuál de los productos es el que efectivamente se está ofreciendo dentro de todos los artículos que aparecen en una misma imagen y así no tener la necesidad de calcular descriptores para aquellos productos que no se están vendiendo. Con respecto a la búsqueda de productos similares dentro de la tienda, será posible utilizar información texto-visual para este objetivo, lo que se espera capte de mejor manera la noción de similitud entre productos, puesto que en muchos casos puede ocurrir que por un mal desempeño de los descriptores, el objeto más similar visualmente corresponda a un objeto que o bien no se esta vendiendo o bien posee un texto radicalmente distinto.

1.4. Objetivos

1.4.1. Objetivo general

El objetivo general de este trabajo es analizar e implementar un buscador de prendas de vestir que aproveche tanto la información textual como visual de los productos presentes en un catálogo para poder ofrecer una mejor experiencia de compra a los clientes de Impresee.

1.4.2. Objetivos Específicos

1. Diseñar y entrenar un modelo de tipo red neuronal que sea capaz de crear un embedding en el que puedan coexistir objetos visuales, textuales y texto-visuales.

2. Evaluar el desempeño de este modelo en la tarea de reconocer cuál producto es el que efectivamente se está vendiendo en una imagen, dada la información del título del producto.

3. Evaluar el comportamiento de este modelo en la recomendación de productos similares utilizando un enfoque texto-visual, en comparación a la recomendación netamente visual de productos.

4. Evaluar cómo aumenta el desempeño de la búsqueda por similitud al emplear una primera etapa de identificación del producto vendido en una imagen.

Capítulo 2

Marco teórico y estado del arte

2.1. Information Retrieval

Information Retrieval consiste en: dado un objeto de consulta y una colección de objetos, encontrar aquellos objetos relevantes para el objeto de consulta dentro de la colección de objetos, donde el concepto de relevante depende de la aplicación, los elementos considerados relevantes son aquellos "más similares" bajo alguna medida de similitud.

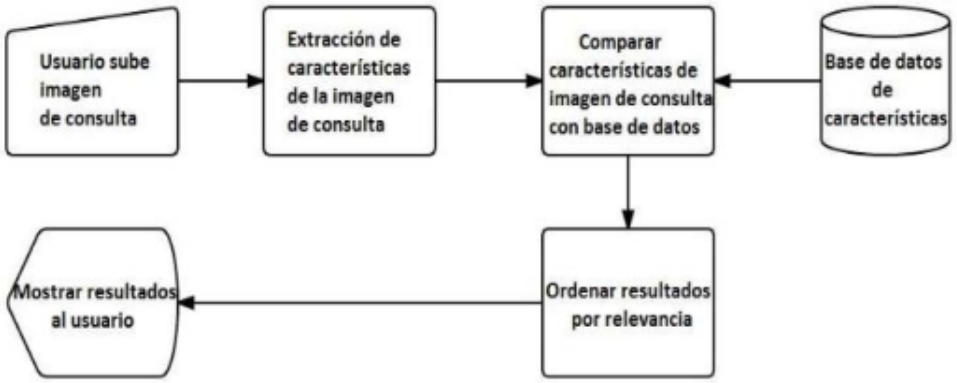

Figura 2.1: Information Retrieval aplicado a la búsqueda por imágenes

Para tratar de forma computacional el concepto de similitud, es necesario que entre los objetos que forman tanto la consulta como la base de datos se pueda aplicar una función de similitud; normalmente, a los objetos se les representa como vector, y la semejanza entre ellos se calcula con alguna medida de similitud de vectores, usualmente distancia euclidiana o similitud coseno.

En la Figura 2.1 es posible observar el esquema general del Information Retrieval: dado un objeto de consulta, se calcula un descriptor a este objeto, el cual es comparado con los otros objetos presentes en la base de datos. Finalmente, el sistema muestra los objetos ordenados de mayor a menor similitud.

En el caso del e-commerce la recomendación de productos o búsqueda de productos sigue este mismo esquema, con la diferencia de que sólo son relevantes los cinco, diez o veinte primeros resultados de la búsqueda, puesto que los usuarios no suelen explorar más allá de estas posiciones.

2.2. Red neuronal

Una red neuronal de una capa hace referencia a un modelo matemático que mapea un vector x de dimensión m a un vector y de dimensión n de la siguiente forma:

$$y = f(Wx + b) \tag{2.1}$$

donde W representa a una matriz de (m x n) dimensiones llamada "matriz de pesos", b representa un vector de n dimensiones llamado bias y la función f corresponde a una función usualmente no lineal llamada función de activación. Una red neuronal de varias capas no es más que la conexión sucesiva de varias redes neuronales de una sola capa unidas en serie, es decir, donde la salida de una es la entrada de la otra.

A la primera capa de la red usualmente se le define como "capa de entrada del modelo", a la última capa de la red se le llama "capa de salida del modelo", y a toda capa que no sea de salida o de entrada se le denomina "capa oculta" (Figura 2.2).

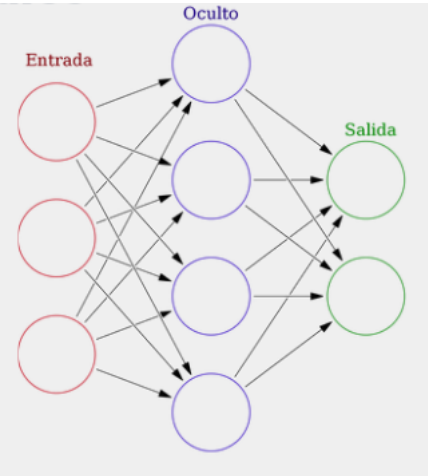

Figura 2.2: Red neuronal multicapa. Se aprecian las capas de entrada y salida del modelo, y las capas ocultas.

Los pesos de la red junto con el bias son parámetros del modelo que se deben aprender a través de los datos utilizando algún algoritmo. Así, dado un conjunto de datos de entrenamiento x_{i} de vectores de dimensión m, y sus respectivos outputs y_{i} de vectores de dimensión n, se calcula una cierta función de error dependiendo de la salida del modelo, la que se busca minimizar por medio de algún algoritmo basado en el gradiente descendente, lo que permite actualizar el valor de los pesos y los bias.

A diferencia de otros algoritmos clásicos de inteligencia artificial, las redes neuronales presentan una capacidad notablemente mayor de aprender relaciones complejas, además de ser capaces de conectarse de forma serial (una capa tras otra), lo que permite crear representaciones cada vez más convenientes y sofisticadas de los datos a medida que la red va más y más profundo [1]. No obstante, presentan el inconveniente de que a causa de su alta capacidad, sus resultados generalizando lo aprendido, dependen fuertemente de la arquitectura utilizada y de la variabilidad y cantidad de datos, corriendo el riesgo de que la red memorice completamente los datos a los que fue expuesta y que, consecuentemente, el modelo no generalice.

2.3. Red neuronal convolucional

Las redes neuronales convolucionales son un tipo de red neuronal especializado en el procesamiento de información que posea una naturaleza espacial, logrando excelentes resultados particularmente en el análisis de imágenes. Las redes neuronales tradicionales presentan el problema de conectar toda neurona de entrada con toda neurona de salida, lo cual puede ser muy poco conveniente en caso de tratar con imágenes o cualquier tipo de dato que tenga algún sentido espacial. Previo a la creación de las redes neuronales convolucionales, para tratar con imágenes, primero se escogía una imagen y se transformaba en un vector unidimensional que luego era entregado como entrada a la red, lo que ocasionaba la pérdida de toda información espacial, además de ser necesaria una cantidad de cómputo excesiva para imágenes de tamaños relativamente medianos, siendo aún así el desempeño bastante ineficiente. Las redes neuronales convolucionales solucionan esto mediante el concepto de filtro, utilizado ampliamente en el procesamiento de imágenes. Un filtro usualmente corresponde a una matriz con ciertas características que se aplica a una imagen para resaltar sus detalles relevantes (Figura 2.3).

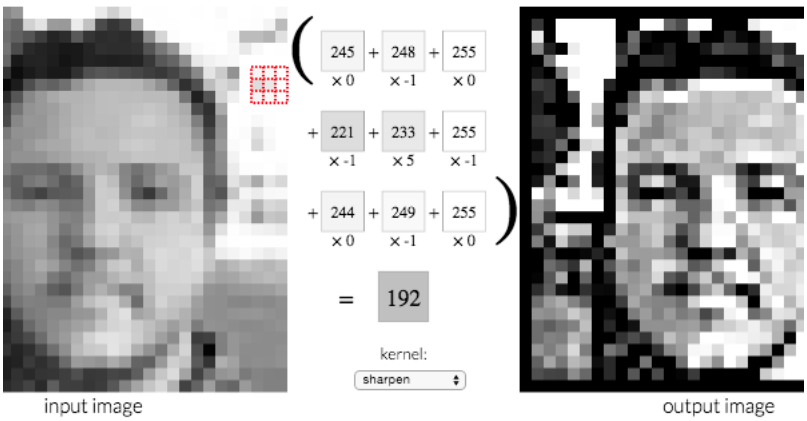

Figura 2.3: Ejemplo de filtro clásico utilizado en el procesamiento de imágenes.

Estos filtros usualmente eran diseñados por expertos en visión computacional y eran acordes a la tarea que se quería resolver. Bajo el paradigma de las redes neuronales, estos filtros se volvieron capaces de ser aprendidos, además de sumarle no linealidades tales como funciones de activación (Figura 2.4).

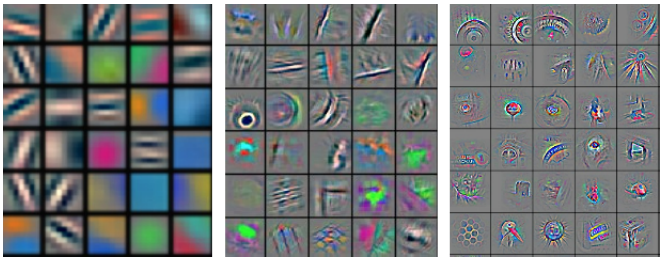

Figura 2.4: Ejemplo de filtros aprendidos por una red neuronal convolucional.

Lo anterior permite crear redes que procesen imágenes mediante la aplicación de filtros sucesivos aprendidos directamente de los datos; de esta forma, las primeras capas de la red corresponden a filtros que aprenden representaciones básicas (identificación de líneas verticales y horizontales, relación entre colores, entre otras), mientras que las últimas capas corresponden a filtros que logran identificar elementos más conceptuales, tales como la presencia o ausencia de cierta forma geométrica.

Luego del procesamiento convolucional es usual insertar una capa fully connected para relacionar estos nuevos features de más alto nivel, de manera de proceder a realizar tareas más complejas como clasificación o regresión.

Las redes neuronales convolucionales lograron superar con creces a cualquier otro algoritmo de inteligencia artificial clásico existente hasta el momento, constituyendo en la actualidad el algoritmo utilizado por defecto en lo que respecta al procesamiento de imágenes.

2.3.1. MobileNet

Si bien las redes convolucionales son uno de los mejores algoritmos en lo que respecta a visión computacional, continúan siendo costosas en términos computacionales. Los recientes avances en redes neuronales han permitido entrenar redes convolucionales cada vez más profundas, las que si bien han aumentado su rendimiento notablemente en distintas tareas, han causado al mismo tiempo que la capacidad computacional necesaria tanto para entrenar como para realizar inferencia con estas redes sea cada vez mayor. En el año 2017 se crean las redes de tipo MobileNet[1], las cuales a costa de perder rendimiento en comparación a arquitecturas más profundas como ResNet[2] o DenseNet[3], logran entrenarse y efectuar inferencia ocupando muchos menos parámetros y a una velocidad notoriamente mayor que sus contrapartes [3](Figura 2.5).

[1]https://keras.io/api/applications/mobilenet/
[2]https://keras.io/api/applications/resnet/
[3]https://keras.io/api/applications/densenet/

Model	Top-1 accuracy	Num. Params.
VGG-16	71.0	14,714,688
MobileNet	71.1	3,191,072
Inception V2	73.9	10,173,112
ResNet-101	76.4	42,605,504
Inception V3	78.0	21,802,784
Inception Resnet V2	80.4	54,336,736

Figura 2.5: Comparación de la precisión y cantidad de parámetros de MobileNet en comparación a otro tipo de arquitecturas en la tarea de clasificación.

2.3.2. Transfer learning

Las redes neuronales convolucionales necesitan de una gran cantidad de imágenes para poder generalizar lo aprendido, sin embargo, en muchos casos que tal cantidad de datos no se encuentra disponible. Una alternativa consiste en tomar la parte convolucional de una red entrenada en un dataset de grandes dimensiones (que se espera entregue features adecuados para una gran variedad de tareas) y conectar encima una capa fully connected para adaptar los datos al problema en particular que se esté estudiando. A esta sección de la red que se encarga de extraer las características usualmente se le conoce como backbone. Dependiendo de la cantidad de datos que se tenga a disposición, es posible descongelar las últimas capas del backbone para poder adaptar aún mejor las características extraídas al problema particular en estudio. En caso de contar con una muy poca cantidad de datos, la técnica común es congelar el backbone, pues datos con tan poca variabilidad (dataset pequeño) sólo llevarán al overfitting y a una disminución del desempeño de validación de la red [1].

2.4. Object Detection

La detección de objetos puede definirse como: dada una imagen, encontrar las regiones en las cuales se encuentran instancias de objetos de interés, donde qué es y qué no es un objeto de interés depende de la aplicación. En la Figura 2.6 se define que entre los objetos de interés se encuentran los perros, los autos y las bicicletas; así, la red señala dónde se encuentran instancias de estos objetos. Actualmente, la mayoría de las aplicaciones de detección de objetos se lleva a cabo mediante el uso de redes neuronales. Es común que en problemas de Information Retrieval el objeto o región de interés sólo se encuentre en una sección de la imagen (por ejemplo, rostros en una imagen de un paseo peatonal). Detectar la región en la cual se encuentra dicho objeto permite calcular sus descriptores, de forma de evitar el ruido propio de la imagen o de otros objetos.

Las detecciones realizadas por la red normalmente son de tipo rectangular, denominándose también bounding boxes o simplemente detecciones. Estos bounding boxes son las coordenadas del rectángulo que encierra a los objetos de interés, los cuales son los outputs de la red en cuestión. Para cada uno de los bounding boxes la red entrega un valor de confianza, el cual representa la probabilidad de que dentro de la detección efectivamente se encuentre un

objeto de interés. (Figura 2.6).

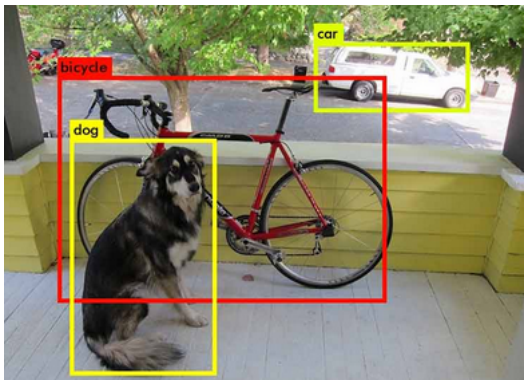

Figura 2.6: Uso de bounding boxes para la detección de objetos de interés.

En general, en los problemas de detección se descartan todos aquellos rectángulos que tengan una confianza menor a cierto umbral para evitar tener una gran cantidad de falsas detecciones o detecciones redundantes. El valor de este umbral depende del problema en específico y de la aplicación, siendo en general un hiperparámetro del modelo.

Por último, es usual que aun habiendo ajustado el umbral de confianza gran parte de los detectores entregue muchos bounding boxes redundantes, para evitar esto existe el hiperparámetro de non max supression, el cual penaliza aquellos bounding boxes que son muy similares a otros.

2.4.1. Yolov3

Yolov3 es un framework de detección de objetos basado en redes neuronales del año 2016 que, si bien a la fecha no es posible considerarlo dentro de los modelos estado del arte, se hizo muy popular gracias a los buenos resultados que presenta en gran variedad de aplicaciones, además de existir a la fecha gran cantidad de modelos ya entrenados para detectar diversas clases de objetos [6].

2.5. Red neuronal recurrente

Así como las redes neuronales clásicas fallaban en captar las relaciones espaciales de los datos, para lograr capturar su naturaleza secuencial se crearon las redes neuronales recurrentes. Este tipo de redes permite procesar la información de manera secuencial, procesando la entrada de los datos paulatinamente (Figura 2.7), demostrando ser uno de los mejores algoritmos para el procesamiento de texto siendo sólo superados por los algoritmos de tipo transformers; sin embargo, estos últimos requieren de una cantidad de datos mucho mayor, situación que es contraria a la de este trabajo.

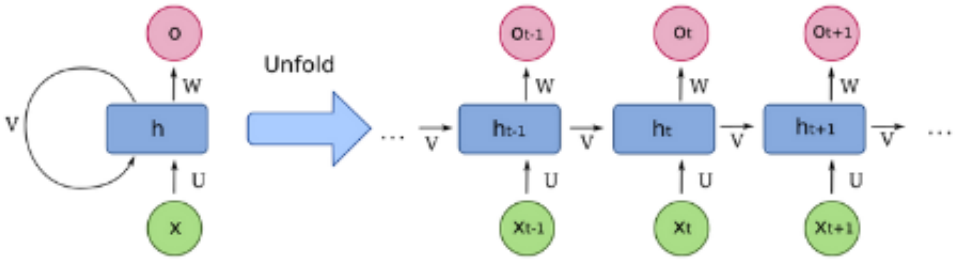

Figura 2.7: Redes neuronales recurrentes.

2.5.1. Redes LSTM

Las redes LSTM corresponden a un tipo de red neuronal recurrente que pretendía resolver uno de los mayores problemas de esta clase de redes: éstas eran poco capaces de relacionar información temporalmente lejana, como por ejemplo, el inicio y el final de una oración. En la actualidad, se han convertido en el algoritmo utilizado por defecto en lo que respecta al uso de redes neuronales recurrentes [8] (Figura 2.8).

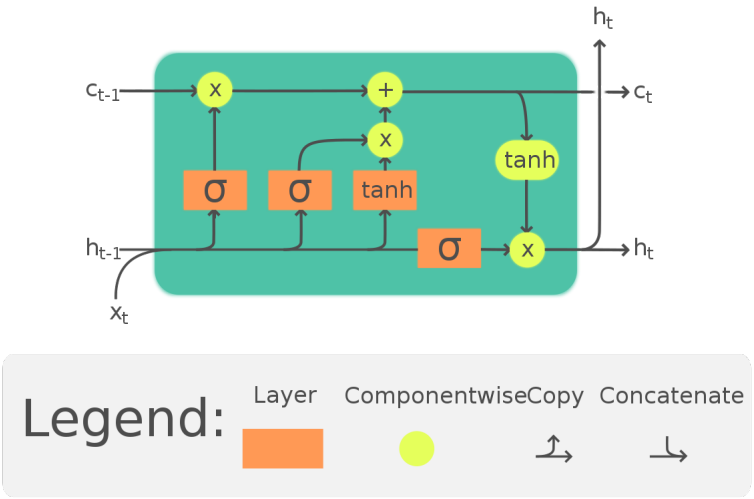

Figura 2.8: Redes LSTM.

2.6. Word Embedding

Los word embeddings corresponden a vectorizaciones de texto aprendidas por una red neuronal a través de la realización de una determinada tarea. En este contexto es posible

citar trabajos como GloVe[4], Word2vec[5] o FastText[6], todos los cuales giran en torno a la idea de que dada una palabra en un corpus, predecir las palabras que rodean a ésta. Lo anterior ha demostrado tener excelentes resultados en tareas clásicas del procesamiento de lenguaje natural, como la clasificación de texto.

Sin embargo, también es posible aprender un word embedding para una tarea en específico, entrenando la red para clasificar o realizar regresión sobre un conjunto de datos, de forma de obtener una vectorización especialmente útil para ese caso de estudio, a diferencia de los word embeddings obtenidos mediante Word2vec, GloVe o FastText.

Un problema usual de todos los word embeddings obtenidos de grandes corpus de texto es que no logran captar de manera adecuada relaciones de otros aspectos de la realidad como podrían ser las imágenes, puesto que conceptos visualmente distintos tienden a aparecer bajo los mismos contextos en los corpus de texto, con lo que los embeddings de polera, camisa, o blusa tienden a ser demasiado similares, lo cual podría ser altamente perjudicial para un modelo que busque diferenciar aquellos conceptos.

2.7. Redes siamesas

Las redes siamesas corresponden a un tipo de arquitectura caracterizada por compartir los pesos de la red para diferentes entradas, permitiendo mapear los diversos ejemplos entregados a la red a un espacio común. Han sido ampliamente utilizadas en relacionar información de distintos dominios (ej: imágenes y dibujos) y identificación de personas.

2.8. Triplet loss

La triplet loss corresponde a un tipo de función de pérdida que se aplica usualmente a las salidas de una red siamesa, buscando asignar objetos similares cerca y objetos distintos lejos.

El enfoque más clásico para crear un espacio de similitudes consiste en entrenar una red para clasificación, esperando que los features obtenidos en la penúltima capa reflejen las relaciones de similitud esperadas. Sin embargo, en este enfoque no se tiene un control directo sobre qué es similar a qué. A modo de ejemplo, si se entrena un clasificador de perros y se utilizan estos features para establecer similitud, las imágenes de perros más similares serán podrían ser las de perros del mismo color, perros en la misma posición, o imágenes del mismo perro. La triplet loss ataca este problema directamente estableciendo relaciones de similitud y no esperando que éstas se generen espontáneamente de alguna tarea auxiliar como clasificación.

El funcionamiento de la triplet loss es el siguiente: dado un objeto base o anchor (x_{anchor}),

[4]https://nlp.stanford.edu/projects/glove/
[5]https://www.tensorflow.org/tutorials/text/word2vec
[6]https://fasttext.cc/

un objeto similar al objeto base o positivo ($x_{positivo}$), y un objeto diferente al objeto base o negativo ($x_{negative}$), la triplet loss de este trío de objetos se calcula como:

$$Loss = max(\text{dist}(x_{anchor}, x_{positivo}) - \text{dist}(x_{anchor}, x_{negativo}) + margen, 0) \tag{2.2}$$

Donde *margen* hace referencia a un hiperparámetro del modelo. Al minimizar la pérdida de esta función se logra que los pares anchor-positivo sean más cercanos entre sí que los pares anchor-negativo; sin embargo, para evitar que los ejemplos negativos sean llevados extremadamente lejos, se incluye el margen, el cual ocasiona que si la distancia anchor-negativo es mayor en una cantidad margen a la distancia anchor-positivo, la pérdida pase a ser 0. Lo anterior se puede ver claramente en la Figura 2.9:

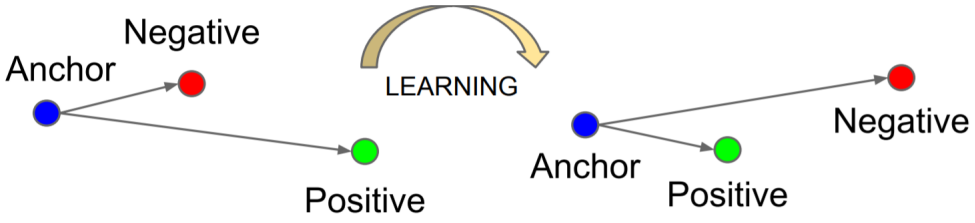

Figura 2.9: Triplet Loss.

Volviendo al ejemplo de los perros, se podría entrenar una triplet net para reconocer perros donde efectivamente las imágenes más similares a la de consulta consistan en el mismo perro visto desde distintos ángulos (envés de, por ejemplo, perros en las mismas poses), donde en este caso la relación de similitud a la que la red le daría más atención sería la de identidad. La triplet loss ha demostrado tener gran éxito en estructurar espacios cuando la cantidad de datos es poca.

2.9. Triplet mining

Al entrenar un modelo utilizando como función de pérdida la triplet loss para que ejemplos similares queden cerca y ejemplos diferentes queden lejos, es muy común encontrarse con que la pérdida converge rápidamente a 0 y que los resultados en tareas como clasificación o retrieval sean bastante pobres. Esto ocurre porque la cantidad de tripletas útiles (con loss mayor a 0) decrece rápidamente una vez se inicia el entrenamiento, por lo que al escoger ejemplos aleatorios la red dejará prontamente de aprender. Para afrontar lo anterior existen diferentes estrategias para calcular tripletas útiles para el aprendizaje del modelo, todas las cuales se engloban dentro de lo que se conoce como "triplet mining" (Figura 2.10).

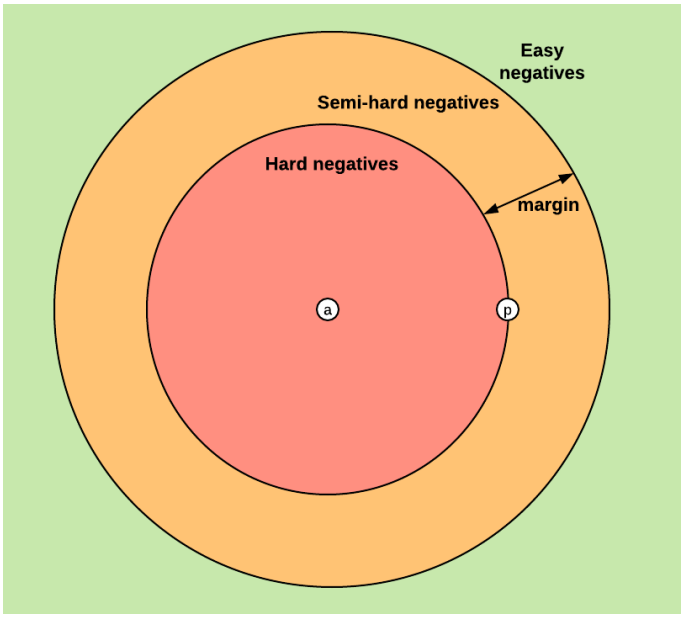

Figura 2.10: Triplet mining: clasificación del tipo de tripletas según la distancia del anchor al positivo y la distancia del anchor al negativo.

El triplet mining puede ser separado en online triplet mining y offline triplet mining. El primero hace referencia a tomar un batch de ejemplos y crear las tripletas más adecuadas al mismo tiempo que se hace la inferencia; sin embargo, esto sólo tiene éxito si dentro del batch existen ejemplos cuya pérdida no es 0. Dada la poca cantidad de datos con los que se realizó este trabajo, se optó por el offline triplet mining, que consiste en calcular la pérdida antes de entregar los ejemplos a la red de forma de asegurar que reciba solamente ejemplos de los que pueda extraer información (pérdida distinta de 0).

2.10. Métricas de evaluación IR

En el Information Retrieval es necesario poder cuantificar el desempeño de un sistema recomendador, para lo cual existen muchas métricas que toman distintos niveles de relevancia dependiendo de la aplicación que se le busque dar al sistema. Dado que este trabajo está centrado en el e-commerce de prendas de vestir, se busca que las primeras respuestas del sistema sean adecuadas, sin importar lo que suceda más haya de las primeras 20 o 10 respuestas.

2.10.1. Recall at k

Dada la consulta de un objeto en un sistema recomendador. El recall at k corresponde a la cantidad de consultas correctas dentro de las k primeras respuestas del sistema recomendador, dividido por la cantidad de respuestas correctas posibles que presentaba dicho objeto. Valores usuales de k son 5 y 10, puesto que en las plataformas de ecommerce los clientes solamente miran las primeras 5 o 10 respuestas.

2.10.2. Precision at k

Dada la consulta de un objeto en un sistema recomendador. El precision at k corresponde a la fracción de consultas correctas dentro de las k primeras respuestas del sistema recomendador. Valores típicos de k al igual que en el recall at k son 5, 10 o 20.

2.11. Estado del arte

Para el desarrollo de este trabajo se utilizaron ampliamente las redes siamesas como base. El paper "FaceNet: A Unified Embedding for Face Recognition and Clustering" [7] muestra con claridad cómo entrenar correctamente dichas redes (si bien para un fin bastante distinto como lo es el reconocimiento de rostros). Señala aspectos importantes tales como la relevancia de utilizar un alto batch size, embeddings de tamaños no muy profundos, normalización L2 para estos últimos, estrategias de triplet mining para el correcto funcionamiento de la triplet loss, entre otros.

También es posible citar como referencia el trabajo "Study on Fashion Image Retrieval Methods for Efficient Fashion Visual Search"[5], en el cual se utilizan redes siamesas para el Fashion Retrieval Visual. Además se demuestra que es posible utilizar redes siamesas para prendas de vestir, aunque sea sólo en el ámbito visual. Se rescatan otras ideas de interés tales como indicaciones para el entrenamiento de la triplet loss y el uso de una función auxiliar de clasificación en el mismo espacio para aumentar el desempeño de la red.

Finalmente se puede citar el trabajo "Flexible Fashion Product Retrieval Using Multimodality-Based Deep Learning"[4], donde se realiza un embedding textual y visual dando buenos resultados. No obstante, en este caso no se utiliza una capa de word embedding como tal y la función de pérdida es mucho más simple que la propuesta en este trabajo, puesto que la red sólo trata de tener como output un 1 cuando el texto corresponda con la clase y un 0 cuando no lo haga.

Capítulo 3

Descripción solución

3.1. Descripción general

El presente trabajo busca poder mejorar el desempeño de los buscadores de e-commerce mediante el uso de un espacio combinado texto-visual. Para realizar esta mejora se propone un esquema de 2 etapas:

1. Dado el nombre del producto y su imagen, encontrar cuál de todas las detecciones presentes en la imagen corresponde al producto que efectivamente se está vendiendo.
2. Encontrar los productos más similares dentro del catálogo de una tienda considerando no sólo texto o imagen sino ambos.

Encontrar cuál de las múltiples detecciones corresponde a la imagen que representa al producto y encontrar similitudes que consideren el aspecto visual y textual de un artículo, pueden ser vistos como aristas de un mismo problema. Esto se resuelve creando un modelo capaz de proyectar el texto de un producto, la imagen asociada al producto y la fusión del texto e imagen del producto en un mismo espacio en el que se cumplan ciertas relaciones métricas, tales como:

1. Dadas diversas detecciones asociadas a un producto, el embedding del texto del producto se encuentre más cercano al embedding del texto más la imagen de la detección que efectivamente se corresponde al título del producto, en comparación a todos los otros posibles embeddings texto-visuales presentes en la imagen del catálogo.
2. Dentro de este espacio aquellos productos visual y/o textualmente similares se encuentren cercanos, y que aquellos productos que tengan tanto una imagen como un texto semejantes se encuentren más cerca que otros productos que tengan sólo texto o imagen similar.

Alternativamente, para poder encontrar cuál detección corresponde al texto es posible optar por un enfoque más clásico al utilizar un detector de ropa y verificar si la clase predicha

por el detector corresponde a la clase de prenda que señala el texto; sin embargo, esto implica que la clase predicha por el detector calza justamente con el texto de la ropa. La situación se ilustra en la Figura 3.1.

Figura 3.1: Error encontrado al utilizar un detector. El texto descriptor de los artículos no se corresponde con las imágenes correspondientes a las prendas.

Más aún, si es que sólo se utiliza como información textual el tipo o clase de prenda, no es posible aprovechar otros aspectos descriptivos de ésta, tales como el color o textura, los cuales pueden ser fundamentales para identificar cuál detección corresponde efectivamente al texto y para encontrar mejores similitudes. En la Figura 3.2 se puede apreciar que utilizando toda la información textual teóricamente se puede escoger de mejor manera cuál es el rectángulo correcto; esto es especialmente útil en los casos donde el detector fallaría bajo el supuesto de sólo conocer la información de la clase de la prenda, no obstante, al aportar más información, es menos probable que la predicción falle. Por ejemplo, al sólo saber que se vende una polera es posible que el detector se equivoque, sin embargo, al saber que corresponde a una polera estampada es más difícil que esto ocurra, puesto que el pantalón bajo ella dista tanto de los conceptos de polera como de estampado.

Figura 3.2: Al aportar más información, la predicción de la prenda se vuelve más precisa.

3.2. Implementación

3.2.1. Dataset

En lo que respecta al dataset para entrenar el modelo, se utilizó el catálogo de uno de los clientes de Impresee que constaba de 320 productos, de los cuales 250 se utilizaron para entrenar el modelo y 70 se utilizaron con fines de evaluación.

Cada producto consta de un total de 1 a 4 imágenes, en las cuales se muestra a una persona modelando la prenda de vestir en distintas vistas y un texto asociado que la describe, como se muestra en la Figura 3.3:

Figura 3.3: Polera tejida terracota.

Aquellas imágenes que capturaban la prenda demasiado cerca (como por ejemplo, la

imagen del extremo derecho de la Figura 3.3) fueron descartadas del dataset, al considerar que no representaban correctamente al producto en cuestión por sí solas.

En el catálogo se definieron regiones de forma manual mediante bounding boxes que contenían las prendas de vestir observadas en la imagen. Dichos bounding boxes señalan además si la prenda es la que se está vendiendo. (Figura 3.4).

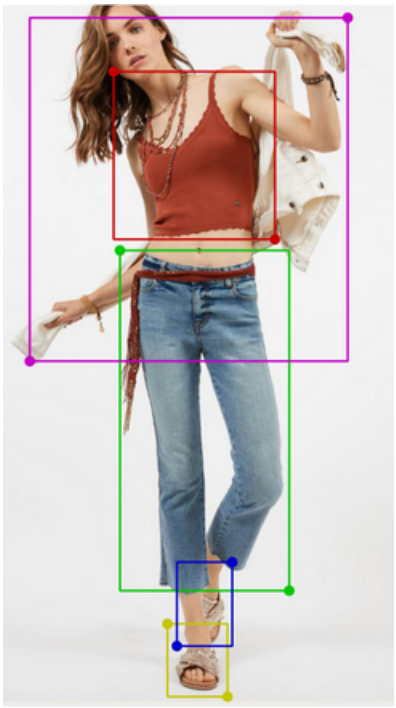

Figura 3.4: En la imagen el producto de interés se titula "Polera terracota", por lo cual el rectángulo rojo (que señala la prenda que se está vendiendo) se encuentra encerrando a la polera. Los demás rectángulos encierran el resto de prendas de vestir presentes (pantalones, zapatos).

Debido a que sólo interesan las regiones en las cuales se encuentran las prendas de vestir, se procedió a trabajar netamente con los recortes de éstas. Cada recorte venía asociado con el texto de su producto y un número binario que señala si la prenda en cuestión efectivamente corresponde a la prenda que se señala en el texto (Figura 3.5). Tomando en cuenta lo anterior, utilizando solamente los recortes, el dataset se consta de 2010 imágenes, de las cuales 857 corresponden a recortes que calzan con su texto y el resto corresponde a recortes de productos secundarios que no calzan con el texto del producto.

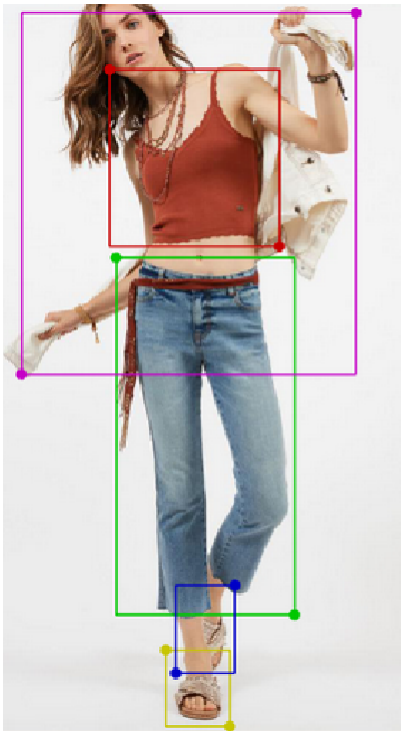

Imagen	Texto	Indicador
	Polera tejido terracota	1
	Polera tejido terracota	0
	Polera tejido terracota	1
	Polera tejido terracota	1

Figura 3.5: En este ejemplo, una imagen se subdivide en 4 productos, cada uno acompañado de un texto más un escalar binario que señala si el recorte corresponde al texto del artículo (0) o no lo hace (1).

Los recortes poseen distintos tamaños, por lo que se procedió a reescalarlos a un tamaño estándar de imagen de 96 x 96 píxeles teniendo cuidado de mantener el aspect ratio. Para lograr esto último se realizó padding alrededor de la imagen. Si no se mantiene el aspect ratio de la detección, la prenda cambia sus proporciones, lo que puede ocasionar que pase a tener más similitud con otra clase. Lo anterior se ejemplifica en la imagen derecha de la Figura 3.6, donde los pantalones pasan a tener más similitud con la clase “short”.

Figura 3.6: Reescalado de imágenes a 96 x 96 píxeles. En la imagen de la izquierda se mantuvo el aspect ratio mediante padding. En la imagen de la derecha éste no se mantuvo, causando que los pantalones pasaran a tener más similitud con la clase “short”.

Todas las imágenes del dataset presentan un fondo uniforme y de buena calidad.

Con respecto al texto, se contó con un vocabulario de 260 términos y largos de frase que

iban desde los 5 a los 12 términos. Además, cada frase que describe al producto comienza con el nombre de éste. Ejemplos de títulos de productos que se pueden encontrar en el dataset son:

- Polera Rayas en folia Misceláneo 1
- Chaleco Cardigan melange con flecos Misceláneo 2
- Polera Top Tejido Dolce Vita Blanco
- Pantalón Maternal Tela De Punto Negro

Los textos fueron limpiados de palabras que se considera no tienen relación con el aspecto de la prenda, tales como el nombre de los diseñadores, marcas, números, entre otros. Posteriormente, fueron codificados en una secuencia de vectores one-hot cuyo largo era del mismo tamaño del vocabulario (260).

3.2.2. Modelo

El modelo propuesto para la realización del espacio texto-visual puede apreciarse en la Figura 3.7:

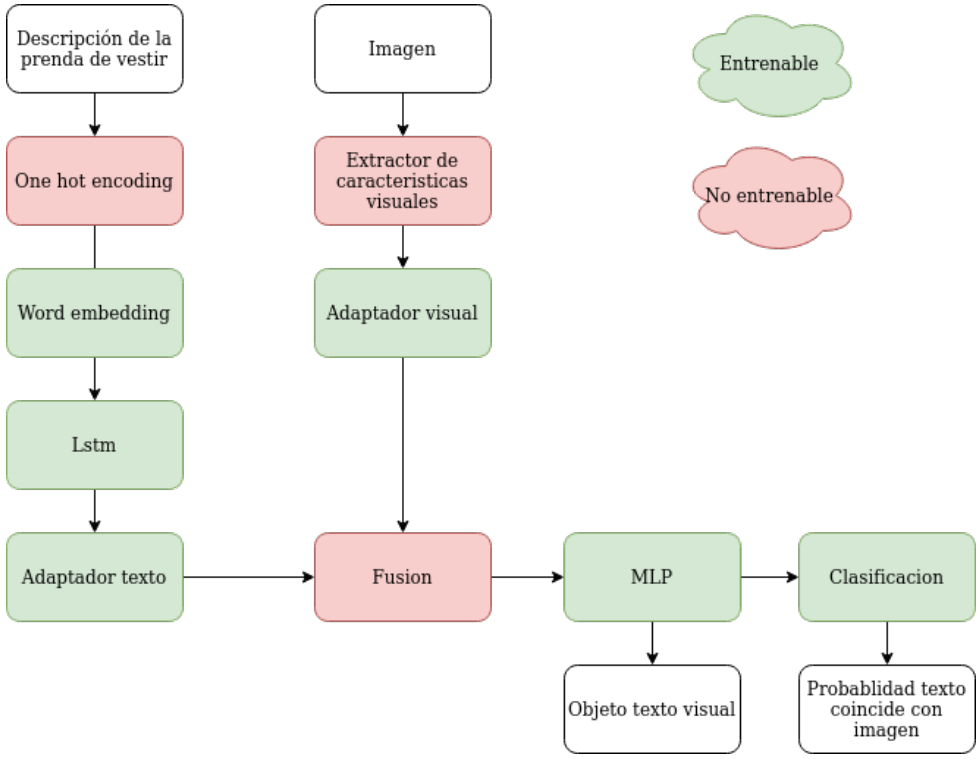

Figura 3.7: Diagrama de alto nivel del modelo propuesto para el espacio texto-visual.

Todo el modelo recibe tensores cuyas primeras dimensiones corresponden al batch size a ocupar, por lo cual son omitidas. A continuación se describirán cada uno de los bloques que conforman el modelo.

Word embedding

Debido a que las descripciones de las prendas de vestir, una vez quitadas las stopwords y palabras irrelevantes, poseían en general un largo menor a 8, se decidió que el modelo procesara solamente las 9 primeras palabras de la descripción. Dichas palabras, como fue mencionado anteriormente, se encontraban codificadas en one-hot encoding. En caso de que la frase tuviera menos de 9 palabras, se autocompletaba con vectores 0 del largo del diccionario.

Cada una de las palabras se pasa por esta capa fully connected de entrada 260 (largo del diccionario) y salida 10 (dimensión del word embedding) para pasar de una representación sparse como es one-hot, a una reprensión densa. La función de activación en este caso fue lineal, como en la mayoría de las capas dedicadas a realizar un word embedding. Que el word embedding tuviera una dimensión de 10 se debe a que el tamaño del vocabulario era bastante pequeño; en la mayoría de los experimentos, utilizar una dimensión mayor resultó en overfitting del modelo.

LSTM

Para poder llevar el significado de toda la frase a un vector, es necesario fusionar la información de cada uno de los vectores representantes de la descripción. Para realizar esto existen distintas alternativas, pero debido a la poca cantidad de datos y a la naturaleza secuencial de las descripciones (en general primero se menciona la prenda y luego sus características) se decidió que lo mejor era utilizar una red de tipo LSTM, que procese secuencialmente cada uno de los vectores representantes de cada una de las palabras y poseyera salida un solo vector de dimensión 8, dimensión la cual fue obtenida como un parámetro de diseño.

Sin embargo, las redes LSTM presentan el problema de que su rendimiento decae al realizar padding, puesto que al encontrarse con una secuencia de vectores cero al final de la frase, se disminuye su representatividad [2]. Para solucionar este problema existen las "bi-direccional LSTM", sin embargo, estas redes se consideran adecuadas cuando la cantidad de datos es suficiente, el cual no era el caso, por lo que se optó por utilizar una red LSTM de forma inversa, lo que hacía que la red no tuviera que encontrarse con secuencias de ceros una vez que ya hubiese procesado las palabras de la descripción.

Adaptador de texto

Corresponde a una capa fully connected de entrada 8 y salida 64 con función de activación tangente hiperbólica. Esta capa cumple la función de poder compatibilizar las dimensiones de la rama de texto y la rama de imágenes.

Extractor visual

Este bloque recibe una imagen de 96 x 96 píxeles y la entrega como entrada a una red MobileNet a la que se le extrajeron las capas de clasificación y se le agregó una operación

de Global Average Pooling. Así, al pasar la imagen de 96 x 96 píxeles se obtiene un vector de 1280. Este bloque no fue entrenado debido a que se consideró que no se contaba con la cantidad de imágenes suficientes para ello. Los pesos de la red MobileNet fueron inicializados desde pesos previamente entrenados en el dataset COCO[1] en la tarea de clasificación de 1000 clases, lo cual puede verse como una aplicación de transfer learning, pues se espera que los features aprendidos en una tarea tan general como el clasificar mil clases de objetos, si bien no sean los óptimos para todas las tareas, sí sean suficientemente representativos para la mayoría de las aplicaciones.

Adaptador visual

El adaptador visual consiste en una capa fully connected que tiene como entrada 1280 neuronas y como salida 64, con función de activación tangente hiperbólica, lo que permite comprimir la información y hacerla compatible con la otra rama de la red (rama de texto).

Fusión

El bloque fusión corresponde a un bloque que toma el vector saliente de la rama de texto y el vector saliente de la rama de imagen y los fusiona en uno solo, dando lugar a un objeto texto-visual. Para esto se utilizaron diferentes estrategias, tales como el promedio punto a punto, la suma punto a punto, la multiplicación punto a punto y la concatenación, encontrándose que el promedio era la mejor opción.

MLP

Este bloque corresponde a una red fully connected de dos capas que permite que se continúen mezclando ambos features más allá de un simple promedio. La entrada de esta capa es de dimensión 32 y la salida es de dimensión 16. Así, dado que se le entregan una imagen y un texto al modelo, es posible obtener un vector que representa el contenido de ambos. A este nuevo tipo de descriptor se le llamará "descriptor texto-visual".

Normalización

Al final de esta capa se aplica una normalización L2 para poder llevar todos los embeddings a la esfera unitaria, dándole un sentido mucho más concreto al margen de la triplet loss (siguiente sección) puesto que en la esfera unitaria la mayor distancia posible entre dos puntos es 2.

[1] https://cocodataset.org/home

Clasificador

Corresponde a una capa fully connected con sólo una neurona de salida y una función de activación sigmoide cuyo propósito consiste en realizar una clasificación entre elementos "plausibles y no plausibles" (si el vector texto-imagen procede de un texto que efectivamente calza con la imagen o no).

Es necesario notar que el modelo propuesto anteriormente es capaz de recibir netamente una imagen; en este caso la otra entrada a la red (es decir, el texto) consistiría en un vector de ceros. También es capaz de recibir texto (en cuyo caso la entrada a la imagen serían ceros) y texto e imagen al mismo tiempo, donde la entrada texto y la entrada imagen son distintas de 0.

3.2.3. Entrenamiento

El modelo propuesto cumple con poder enviar estos tres tipos de datos (imagen, texto y texto-imagen) a un sólo espacio; sin embargo, con los pesos del modelo sin entrenar (es decir, inicializados de manera aleatoria) los envía a un espacio "aleatorio" donde dos datos muy semánticamente distintos pueden quedar muy juntos o muy separados entre sí.

Para poder lograr que este espacio sea semánticamente valioso es necesario entrenar la red para que aprenda estas relaciones semánticas de interés. Para ello es necesario utilizar una función de pérdida, la que usualmente se encuentra relacionada a la tarea que se espera que la red desempeñe. Si bien es posible adoptar distintos enfoques para la construcción de este espacio, tal como pedir a la red que se repliquen las salidas dadas las entradas (autoencoder), en muchos de estos casos no se tiene un control exacto sobre qué relaciones métricas aprende la red. Para especificar estas relaciones existe el "Deep Metric Learning", en el que la función de perdida se corresponde a las relaciones métricas que se espera que el modelo aprenda directamente.

Loss function

Para crear un espacio combinado que considerara tanto texto como imagen se recurrió al uso de la triplet loss con fin de dotar de cierta métrica a este espacio. La idea detrás se basa en tres puntos que este espacio debe cumplir:

1. Que el embedding asociado al recorte de una prenda de vestir esté más cerca de los texto-imagen que corresponden a esta prenda. Para lo anterior basta con implementar una triplet loss en la que el anchor sea el recorte de la imagen, el positivo sea el texto-imagen correspondiente a la imagen y el negativo sea el texto-imagen que no corresponde a la imagen.
2. Que el embedding asociado al texto de una prenda de vestir esté más cerca de los texto-imagen que corresponden a esta prenda. Para lo anterior basta con implementar una triplet loss en la que el anchor sea el texto de la imagen, el positivo sea el texto-imagen

correspondiente a la imagen y el negativo sea el texto-imagen que no corresponde a la imagen.

3. Que el espacio sea capaz de diferenciar los pares texto-imagen que tienen y no tienen sentido. Por ejemplo, si al modelo se le entregan el recorte de un zapato y el texto de un zapato, ese embedding esté alejado del embedding del recorte de un zapato y el texto de una polera. Lo anterior implica que la red sepa qué tan "plausible" es la entrada a la red y así poder colocarlos en zonas distintas del espacio; para esto se propuso un loss del tipo "binary cross entropy" al final de la red, donde la categoría 1 se reserva para aquellos textos que no corresponden a la imagen y la categoría 0 se reserva para aquellos textos que efectivamente corresponden a la imagen.

Figura 3.8: Loss de tipo "binary cross entropy". Dependiendo de si el texto corresponde o no a la imagen, se tienen las categorías 0 y 1, respectivamente.

Por tanto, la loss total bajo la cual se entrena este modelo queda como la suma del triplet loss utilizando imágenes y texto-imágenes, el triplet loss utilizando texto y texto-imágenes y finalmente el loss de binary crossentropy que diferencia si el texto se corresponde o no con la imagen.

Triplet mining o selección de tripletas

Tripletas válidas: dados un recorte y un texto que corresponde a esa imagen, todas las tripletas válidas son de la forma:

- Positivo: un texto-imagen correspondiente al mismo producto.
- Negativo: un texto-imagen que no corresponde al mismo producto, o un texto-imagen del mismo producto que no es principal, es decir, que la imagen no es el producto que se está vendiendo.

Para escoger qué tripletas debería ver el modelo se optó por la estrategia de "offline triplet mining", es decir, se computaron los embeddings de cada uno de los recortes, de cada uno

de los textos y de cada uno de los texto-imagen y se procedió a calcular aquellas tripletas tales que el triplet loss fuera mayor a 0. Para el entrenamiento de la triplet loss no se sugiere entrenar con las tripletas más difíciles, ya que usualmente éstas representan casos particulares o extremadamente complejos que podrían afectar el rendimiento del modelo. Así, se optó por ordenar las tripletas en función de su triplet loss y primeramente seleccionar al azar de entre las 100k a 1000k tripletas, avanzando luego al entrenamiento entre las 10k a 100k tripletas y finalmente entre las 5k a 20k tripletas, lo que probó dar los mejores resultados.

Hay tripletas que se evitaron en el entrenamiento debido a que existían varios productos con los mismos nombres, por lo cual se tuvo cuidado de no usar tripletas tales que positivo, negativo y anchor tuvieran el mismo texto.

Según diversos trabajos, la triplet loss es especialmente sensible al batch size que se ocupe, demostrándose esto al obtener los mejores resultados con batches del orden de las 1000 tripletas; además, debido al uso del offline triplet mining no se pueden utilizar técnicas que realicen modificaciones dentro de la misma red (tales como batch normalization), puesto que al realizar el offline triplet mining se garantiza que los embeddings tengan un loss mayor a 0, lo cual puede no ocurrir en caso de que se realice una modificación a estos embeddings (como se da al utilizar batch normalization o data augmentation online).

Otras consideraciones del modelo

El modelo fue entrenado mediante el optimizador Adam y learning rate decay = 0.0001[2]. Se trató de mantener la menor cantidad posible de neuronas para evitar overfitting.

3.3. Diseño de experimentos y evaluación

El modelo fue entrenado utilizando el vocabulario e imágenes de una tienda en particular, por lo que es interesante conocer si lo aprendido desde los datos de esta única tienda es extrapolable a otras tiendas, o si será necesario entrenar un modelo para cada una. En base a lo anterior, los experimentos se dividieron en Experimentos Tienda 1, que fueron realizados sobre el set de validación de la tienda con la que se entrenó, y Experimentos Tienda 2, que se llevaron a cabo sobre otra tienda con distinto vocabulario y formato de imágenes.

3.4. Experimentos Tienda 1

El dataset de evaluación de la Tienda 1 constó de un total de 70 productos, los cuales contenían 236 imágenes, existiendo para cada producto 2 a 4 vistas distintas. Cada una de estas vistas contenía de 1 a 4 detecciones, dando un total de 557 detecciones en el dataset, cada una con un texto asociado (nombre del producto en el cual éstas aparecían) y un escalar que indicaba si efectivamente la detección correspondía al texto del producto o no. Es necesario

[2]https://keras.io/api/optimizers/adam/

recordar que en este dataset los bounding boxes asociados a cada prenda de vestir fueron creados de forma manual, tratando de similar el comportamiento de un detector ideal. Debido a que este dataset viene de la misma fuente que el dataset de entrenamiento, se tiene que la gran mayoría de los términos presentes en la partición de validación ya estaban presentes en la partición de entrenamiento, por lo cual los casos en los que el modelo ve una palabra desconocida son bastante escasos; en esos casos se optó simplemente por eliminar la palabra de la frase.

3.4.1. Identificar el producto que se vende en una imagen de la Tienda 1

En este experimento se buscó confirmar si la red era capaz de identificar cuál de todos los bounding boxes era el que se estaba vendiendo, dado el texto de un producto y su imagen asociada. como se señaló anteriormente, es importarte recalcar que en este caso los bounding boxes fueron anotados manualmente, lo que evitó tener bounding boxes repetidos o colocados en lugares en los cuales no se encontraban prendas de vestir.

Para lograr esto se calculó el embedding texto-visual de todas las detecciones, el embedding del respectivo título del producto y la distancia de estos embeddings texto-visuales al embedding del título del producto. Luego, el recorte correspondiente al texto-imagen con la menor distancia al embedding del texto corresponde al recorte asociado al producto, lo cual se ilustra en la Figura 3.9.

Figura 3.9: Mecanismo para seleccionar el bounding box correspondiente al título del producto.

Cada recorte más su texto asociado fueron entregados a la red, obteniendo una cantidad n de embeddings ("Embedding texto visual 1" y "Embedding texto visual 2" en la Figura 3.9).

También se pasó el texto por la red (lo que equivale a pasar el texto y una imagen vacía), obteniendo un embedding de la descripción del producto ("Embedding texto producto" en la Figura 3.9. Luego, se calculó la distancia de cada uno de los embeddings al embedding de texto, y aquel con la menor distancia fue asignado como el recorte correspondiente a la imagen.

La métrica utilizada para evaluar el desempeño de la red correspondió al porcentaje de veces que la red efectivamente fue capaz de discernir cuál bounding box era el correcto.

3.4.2. Mejora de las búsquedas por similitud en la Tienda 1

En este experimento se compararon:

- La búsqueda por similitud mediante descriptores visuales.
- La búsqueda por similitud mediante descriptores texto-visuales.
- La búsqueda por similitud mediante descriptores visuales utilizando un único bounding box por imagen.
- La búsqueda por similitud mediante descriptores texto-visuales utilizando un único bounding box por imagen.

Donde el método para escoger el bounding box representante de cada imagen fue detallado anteriormente en la Figura 3.9.

En este caso se calcularon las métricas de Recall@5, Recall@10 y Recall@20, considerando como respuestas relevantes las imágenes correspondientes al mismo producto.

3.5. Experimentos Tienda 2

En este caso se utilizó el dataset correspondiente a otro cliente de Impresee, el cual tenía su propio vocabulario, constaba de 291 productos y presentaba solamente una imagen por producto. En una etapa de pre-procesamiento se eliminaron todas aquellas palabras que no existieran en la Tienda 1 y que formaran parte de las descripciones de los productos. Sin embargo, el vocabulario de esta tienda presentaba el problema de ser muy redundante en sus términos, junto a tener una mala ortografía. Por ejemplo, la palabra "polerón", aparecía como "poleròn", "poleron", y "polerones". Esto llevó a que la mayoría de las descripciones, una vez eliminadas las palabras no presentes en la Tienda 1, quedaran sin elementos. Para solucionar esto se recurrió a la librería difflib[3], la cual permite comparar strings. Dadas dos palabras, esta librería calcula el porcentaje de similitud entre ellas.

Luego, para cada palabra del dataset de la Tienda 2 se seguía el siguiente algoritmo:

[3]https://docs.python.org/3/library/difflib.html

- Si la palabra estaba también el la Tienda 1, no existía problema.
- Si la palabra no se encontraba en la Tienda 1, se examinaba si existía una palabra dentro del dataset 1 tal que la similitud entre ambas fuera mayor al 95 %. Si dicha palabra existía, ésta reemplazaba a la palabra de la Tienda 2. Si no existía, la palabra de la Tienda 2 era descartada del procesamiento.

Este dataset además estaba conformado por imágenes que, a diferencia de la Tienda 1, no presentaban un fondo uniforme.

Debido a que existían muchas categorías que no estaban presentes en el dataset de la Tienda 1 pero sí en la Tienda 2, se optó por no trabajar con estas categorías, ocupando sólo aquellas que estaban presentes tanto en la Tienda 2 como en la Tienda 1. También fueron descartadas las categorías pobremente representadas en el dataset 1.

3.5.1. Obtención automática de bounding boxes

Para obtener los bounding boxes de las detecciones se utilizó una red del tipo YOLOv3 ya entrenada en DeepFashion2[4] y se ajustaron los parámetros de threshold y nonmaxsupression de tal manera que la red efectivamente detectara toda prenda de vestir presente en una imagen, aunque esto significase la presencia de detecciones redundantes y/o falsos positivos.

3.5.2. Identificar el producto que se vende en una imagen de la Tienda 2

En este experimento al igual que en el caso de la Tienda 1, se mostrará si es que la red mediante el mismo esquema es capaz de dilucidar cuál de todos los bounding boxes presentes en la imagen es el producto en venta de acuerdo a su texto. Será posible además comprobar qué tan bien generaliza el modelo en la tarea de identificar el producto en venta, en lo que respecta a tiendas con distinto formato de imagen (poses de las modelos, fondo) y robustez del vocabulario (qué sucede cuando algunas palabras de la frase se omiten).

La métrica para evaluar es el porcentaje de veces en que la red elige uno de los bounding boxes adecuados, donde se habla de *bounding boxes correctos* y no de *bounding box correcto*, debido a que como estas detecciones fueron obtenidas por la red, varias de ellas son redundantes.

3.5.3. Encontrar elementos similares en la Tienda 2

Dado que tanto el diccionario como el tipo de imágenes cambiaba, se procedió a evaluar qué tan bien se desempeñaba el modelo en la tarea de encontrar elementos similares dentro de la tienda. Debido a que en este caso no se contaba con las otras vistas de la prenda, se

[4]https://github.com/switchablenorms/DeepFashion2

contaron cuántos elementos eran aceptados como plausibles. Es decir que si se buscaba la imagen de una polera aquellas respuestas correctas serian los retrievals relacionados a polera. Además solo se calcularon métricas al usar un representante por imagen, puesto que debido a la alta cantidad de bounding boxes redundantes se aumenta artificialmente el desempeño del buscador sin seleccionar un bounding box por imagen.

Capítulo 4

Resultados y análisis

4.1. Experimentos Tienda 1

A continuación se presentan los resultados obtenidos en el dataset correspondiente a la partición de validación de la tienda con la cual se entreno el modelo o Tienda 1.

4.1.1. Identificación del producto en venta Tienda 1

En este caso la red obtuvo un muy buen desempeño en la identificación del producto en venta, sin embargo, estos resultados variaron según la cantidad de elementos presentes en la Tabla 4.1.1:

caso	cantidad de elementos	porcentaje	baseline
2 productos	94	97 %	50 %
3 productos	38	94 %	33.3 %
4 productos	41	78 %	25 %
5 productos	7	71 %	16 %
todos productos	180	91.02 %	39.45 %

Tabla 4.1: Desempeño del modelo en la identificación del producto en venta según la cantidad de productos y elementos. Baseline representa como se desempeñaría una elección aleatoria.

En la Figura 4.1 se puede apreciar un caso de ejemplo en el que aparecen tres productos, donde se anotaron como detecciones el zapato, el pantalón y la polera, siendo esta última el producto a vender. Es posible observar que la distancia entre el embedding del texto y el embedding del texto más la polera es de 1.39, mientras que las distancias al embedding del texto más el zapato y al embedding texto más el pantalón son de 1.59 y 1.61, respectivamente. Luego la menor distancia corresponde a la polera, concluyendo correctamente que ese es el elemento que se está vendiendo.

Figura 4.1: Anotaciones simulando detector perfecto.

En la Figura 4.2 es posible apreciar otro ejemplo, en este caso de dos productos. El sistema también es capaz de reconocer cuál de ellos se está vendiendo.

Figura 4.2: Ejemplo correcto selección bounding box 2 productos.

En la Figura 4.3 es posible apreciar un ejemplo donde la red no puede diferenciar cuál es el producto en venta. Gran parte de los errores ocurren en categorías de ropa pobremente representadas en la partición de entrenamiento como “falda”.

Figura 4.3: Error de tipo 1 Tienda 1.

Existe además otro tipo de imágenes en las que el modelo falla, tales como algunos casos donde un bounding box se encuentra dentro de otro a causa del traslape existente entre ambos (Figura 4.4).

Figura 4.4: Error de tipo 2 Tienda 1.

En general el funcionamiento del modelo es bastante adecuado, logrando acertar en el 91.02 % de los casos. Considerando que la cantidad de nombres de productos es bastante

recall	visual	texto-visual	visual 1 representante	texto-visual 1 representante
R@k	53 %	58 %	56 %	63 %
R@5	57 %	64 %	57 %	69 %
R@10	65 %	76 %	65 %	77 %

Tabla 4.2: Resultados diferentes similitudes Tienda 1

reducida, por lo que puede existir mucha correlación entre términos a causa de las peculiaridades del dataset. Por ejemplo, si "falda" aparecía siempre junto a la palabra "oscura" en el dataset, era posible que las representaciones de ambos términos fueran muy cercanas, lo que no sucedería en caso de un dataset mayor.

Debido a que el modelo presenta un buen funcionamiento, se espera que al utilizar más productos (es decir, contar con más información tanto textual como visual) el modelo mejore aun más su desempeño.

4.1.2. Mejora similitudes Tienda 1

En lo que respecta a comparar la búsqueda o semejanza mediante similitud visual, similitud texto-visual y sus respectivas versiones escogiendo un elemento por imagen, se obtuvieron los siguientes resultados, los cuales se se resumen en la tabla 4.2:

En la figura 4.5 es posible observar cualitativamente los resultados en un caso de prueba. Es posible notar que al utilizar netamente los descriptores visuales sobre todos los posibles resultados se tienen malos resultados, puesto que muestran productos no relacionados a una polera, en el caso de utilizar descriptores texto visuales sobre todos los posibles bounding boxes se mejora la similitud puesto que al buscar la polera solo se muestran poleras similares, por último al escoger primeramente un bounding box representante de cada imagen se obtienen mejores resultados tanto en la búsqueda visual como en la búsqueda texto-visual.

4.2. Experimentos Tienda 2

A continuación se presentan los resultados obtenidos en el dataset correspondiente a la tienda que no se usó para entrenar; Tienda test o Tienda 2.

4.2.1. Obtención automática de bounding boxes Tienda 2

La métrica en este caso consiste en la cantidad de veces que la red efectivamente logra asignarle un bounding box a la prenda de interés. En este caso se tuvo que variando el parámetro de nms y treshold fue posible lograr que en el 98 porciento de los casos la red fuera capaz de al menos asignarle un bounding box al producto de interés.

La forma manual en la que se colocaron los bounding boxes es muy similar a la forma en

Figura 4.5: Similitudes tienda 1: En la primera fila se presentan los resultados de la similitud visual, en la segunda fila se muestra los resultados de la similitud texto-visual, en la tercera fila se presentan los resultados de la similitud visual escogiendo una detección representante por imagen y en la ultima final se presentan los resultados de la similitud texto-visual escogiendo una detección representante por imagen

la que los colocó la red YOLOv3 en la mayoría de los casos, lo cual puede apreciarse en la Figura 4.6.

Figura 4.6: Ejemplo 1 detección YOLOv3 Tienda 2

4.2.2. Identificación del producto en venta Tienda 2

Los resultados en este caso fueron altamente satisfactorios, lo que pudo deberse en parte a que no se trabajó con las clases pobremente representadas en el dataset (como por ejemplo, el caso de "falda", donde el modelo sólo vio unos pocos elementos). Al trabajar netamente con las clases de seleccionadas, se obtuvo que el modelo logra identificar la prenda en cuestión en un 93 % de las veces. Es importante recordar que en este caso no se presentan los resultados separados por cantidad de productos.

Al analizar cualitativamente los resultados es posible inferir que el hecho de utilizar un detector real aumenta el rendimiento del modelo, contrario a lo esperado. Así, es posible considerar que la redundancia de bounding boxes ayuda al modelo a encontrar el correcto.

El hecho de que la red haya funcionado en un dataset en el que no todos los términos se parecen da a entender que el modelo posee una gran capacidad de generalización. Además, el dataset de esta tienda posee la peculiaridad de que el fondo de las imágenes es radicalmente distinto al de la tienda de entrenamiento, por lo que es posible inferir que el uso de información textual otorga cierta robustez al ruido de fondo.

Figura 4.7: Selección bounding box correcto caso 1 Tienda test

4.2.3. Mejora similitudes Tienda 2

En este caso no se contaba con múltiples vistas de un producto, por lo cual se procedió a caracterizar aquellos elementos relevantes como aquellos que pertenecen a la misma clase del producto y efectivamente la imagen del recorte corresponde al texto del productos. Obteniéndose los siguientes resultados (Tabla 4.3).

precision	visual 1 representante	texto-visual 1 representante
p@5	59.31 %	81.38 %
p@10	50 %	74.81 %

Tabla 4.3: Tabla resultados similitudes Tienda 2

Los resultados cualitativos de las similitudes encontradas se pueden observar en la figura 4.8

Figura 4.8: Búsqueda por similitud tienda 2: En la fila de arriba se observa las similitudes obtenidas al buscar el producto mediante descriptores visuales cuya imagen se encuentra en la tercera fila y su texto es çhaleco sole chaleco sin mangas medidas más abajo en descripción". En la fila de al medio se observan las similitudes obtenidas por descriptores texto-visuales

Cuantitativamente se observa que los descriptores texto-visuales efectivamente representan una gran mejora en la búsqueda de similitudes en la tienda de test, cualitativamente se observa que al considerar la información textual el modelo tienda a darle prioridades a elementos los cuales son de la misma categoría que el objeto de búsqueda, por lo que la calidad de la búsqueda aumenta notablemente.

Conclusión

En este trabajo se desarrolló un modelo basado en redes neuronales, el cual es capaz de aprovechar la información textual y visual de un catálogo de e-commerce para identificar cuál producto se vende en un imagen. Esto permite encontrar mejores similitudes en comparación a sistemas más tradicionales como la similitud basada en texto o la similitud basada en imágenes. Considerar la información visual y textual conlleva una serie de ventajas, entre ellas:

1. Lograr identificar cuál producto se está vendiendo dentro de la tienda. Esto mejora notablemente el rendimiento de cualquier buscador, puesto que en la búsqueda no se consideran aquellas imágenes correspondientes a productos que no están en venta. En este espacio texto-visual es posible identificar cuál producto está en venta cerca del 90 % de las veces en la Tienda 1 y un 93 % de las veces en la Tienda 2.
2. Encontrar mejores similitudes dentro de los productos de la tienda. Esto permite recomendar a un usuario alternativas de producto que sean similares tanto desde un punto de vista textual como visual. Esto aumenta en un 15 % (de un 65 % a un 75 %) el recall at 10 en la Tienda 1 y en un 50 % (de un 50 % a un 74.81 %) el precision at 10 en la Tienda 2.

Al incluir dentro de la triplet loss el elemento combinado "texto-visual" (fusión de texto e imagen) fue posible aumentar radicalmente el desempeño del modelo, en comparación a utilizar solo texto y imágenes. Esto se debe a que la similitud de un objeto formado por texto e imagen, no tiene por qué ser la suma de las similitudes de texto e imagen por separado, problema que no está presente al considerarlos como objetos texto-visuales.

Debido a los excelentes resultados obtenidos validando en la segunda tienda, se infiere que el poder de generalización de este modelo es bastante grande para tiendas del mismo dominio (en este caso prendas de vestir). Sin embargo, al entrenar con un vocabulario mucho más extenso a través de una serie de tiendas se espera que se obtenga un modelo que mejore aún más los resultados y pueda ser aplicado a cualquier tienda del dominio.

En lo que respecta a llevar esta metodología a otros dominios más allá de las prendas de vestir, se tiene que cumplir el supuesto de que exista un detector capaz de identificar todas las regiones de interés presentes en la imagen (productos). Este detector no requiere que las categorías sean asignadas correctamente, puesto que, dado el texto de la prenda es posible identificar cuál detección corresponde al producto señalado en el texto y, más aún,

la redundancia de detecciones no se mostró que afectara en lo que respecta a seleccionar la detección correcta. Para entrenar basta con seleccionar manualmente las regiones en las cuales se encuentran las prendas de interés e identificar cuál de todas esas regiones corresponde a la prenda señalada en la descripción del producto, y con esta información entrenar un modelo.

Es necesario señalar que al utilizar el enfoque texto-visual se saca mucho más provecho de la información, por lo que la cantidad de datos necesarios para entrenar un modelo funcional es radicalmente menor a otros enfoques tales como traducción de texto a imagen.

Trabajo futuro

Este trabajo podría continuarse de distintas formas. En caso de que se quisiera desarrollar un modelo que sea generalizable a todo tipo de tienda (vestuario, decoración, etc.) y el detector no fuera capaz de identificar el producto entre sus detecciones, sería posible realizar un detector condicional a la información del texto, similar a lo realizado en este trabajo utilizando la triplet loss. Sin embargo, es esperable que este enfoque tome muchos más datos, del orden de decenas de miles.

También se puede construir un dataset de gran escala para evaluar cómo mejora el desempeño del modelo y explorar relaciones interesantes que puedan surgir de éste, las cuales no pueden ser apreciadas utilizando un vocabulario tan reducido como en este caso.

Dado lo anterior, con una cantidad suficiente de datos sería posible construir un word embedding visual, es decir, construir una vectorización de texto capaz de captar relaciones de similitud visual.

Bibliografía

[1] Chollet.F. *Deep Learing with Python*. Manning Publications, 2017.

[2] Mahidhar Dwarampudi and N. V. Subba Reddy. Effects of padding on lstms and cnns. *ArXiv*, abs/1903.07288, 2019.

[3] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv*, abs/1704.04861, 2017.

[4] Yeonsik Jo, Jehyeon Wi, Minseok Kim, and Jae Yeol Lee. Flexible fashion product retrieval using multimodality-based deep learning. *Applied Sciences*, 10:1569, 2020.

[5] Sanghyuk Park, Minchul Shin, Sungho Ham, Seungkwon Choe, and Yoohoon Kang. Study on fashion image retrieval methods for efficient fashion visual search. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 316–319, 2019.

[6] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *ArXiv*, abs/1804.02767, 2018.

[7] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015.

[8] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for language modeling. In *INTERSPEECH*, 2012.