UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

ESTUDIO Y ANÁLISIS DE BÚSQUEDA VISUAL-SEMÁNTICA Y SU APLICACIÓN

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

FABIÁN ALEJANDRO GONZÁLEZ ESPINOSA

PROFESOR GUÍA:
JUAN BARRIOS NUÑEZ

MIEMBROS DE LA COMISIÓN:
FELIPE BRAVO MARQUEZ
PABLO GONZÁLEZ JURE

SANTIAGO DE CHILE
2019

Resumen

El presente documento expone el trabajo realizado durante el estudio y análisis de distintas técnicas de aprendizaje de máquina, dentro de las áreas del procesamiento de lenguaje natural y reconocimiento visual, que permitan desarrollar un servicio de búsqueda visual-semántico para la recuperación de productos de tiendas en línea.

El problema a resolver, nace de la necesidad de realizar un estudio y análisis sobre búsqueda visual-semántica, con el objetivo de evaluar si la incorporación de texto mejora o no la precisión de los resultados arrojados por un buscador visual. Además, se requiere comparar distintas técnicas y algoritmos de búsqueda que utilicen texto e imagen, con el propósito de encontrar y definir, la mejor forma de combinación de ambas características.

La solución propuesta para resolver el problema planteado, consta de realizar un estudio de las técnicas y modelos más actuales para el procesamiento de lenguaje natural y reconocimiento visual, junto con métodos de evaluación de sistemas de recuperación de información. Posteriormente, se implementan distintos algoritmos de búsquedas por texto e imágenes, que son evaluados frente a un conjunto de diversas consultas en las que se muestre el comportamiento de los algoritmos diseñados.

Respecto de los resultados, se concluye que la incorporación de texto a la consulta mejora en aproximadamente un 16 % la precisión de los resultados obtenidos en comparación con una búsqueda visual. Además, se concluye que los descriptores utilizados para la caracterización de textos e imágenes, son útiles para la realización de búsquedas por similitud. Junto con lo anterior, los resultados indican que la utilización de descriptores de textos obtenido mediante modelos de vectorización Bag of Word con TF-IDF, en conjunto con vectores descriptores de imágenes obtenidos con redes convolucionales, conforman la mejor combinación encontrada para la realización de búsquedas de productos dentro de catálogos de tiendas.

En memoria de mis abuelos Irma y Guido

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

Contexto

Impresee[1] es una empresa dedicada a la investigación aplicada de la inteligencia artificial, que se ha especializado en la implementación de servicios de búsqueda avanzada de productos dentro de catálogos de tiendas en línea. Los tipos de búsqueda ofrecidos por la empresa son: búsqueda por texto y búsqueda visual. Este último es el que caracteriza a la empresa, ya que permite encontrar dentro de un catálogo, los productos más similares a una imagen o un dibujo dado por el usuario. Este buscador visual es capaz de extraer las características de un objeto dentro de una imagen, ya sea forma, color, detalles, entre otros; y encontrar aquellos productos del catálogo que se asemejan más al objeto buscado. En la imagen de la Figura 1.1, se observan ejemplos de resultados entregados por el buscador visual de Impresee, para los dos tipos de servicios ofrecidos por la empresa, es decir, por foto/imagen y por dibujo.

Figura 1.1: Ejemplos de resultados utilizando los servicios de búsqueda visual de Impresee

Actualmente, la empresa desea expandir sus servicios para ser utilizados en redes sociales como Twitter, Facebook o Instagram, ya que dentro de estas aplicaciones es natural para el usuario compartir imágenes. El objetivo de esta expansión es que el usuario sea capaz de encontrar el producto que desea, sin la necesidad de acceder a la página de la tienda específica en la que se encuentra el producto que busca, mejorando así la experiencia del usuario en la búsqueda de su producto. Por otra parte, la integración con redes sociales hizo que la empresa

[1]Impresee: `https://impresee.com/`

quiera incorporar una búsqueda visual-semántica a sus servicios, con el propósito de mejorar la precisión de sus resultados.

Ahora bien, una búsqueda semántica[2] es aquella que comprende la intención y el contexto de la búsqueda, con el objetivo de mejorar la precisión de los resultados. Este tipo de búsqueda considera la variación de las palabras, los sinónimos, el contexto de la búsqueda, entre otros, y es posible encontrarlo en grandes motores como lo son Google o Bing. Por lo tanto, entenderemos por búsqueda visual-semántica a la búsqueda visual de un producto incorporando texto en lenguaje natural a la consulta.

Motivación

Dentro de la situación planteada previamente, nace la necesidad de estudiar técnicas de aprendizaje de máquina (Machine Learning) en el área del procesamiento de lenguaje natural y reconocimiento visual, que permitan implementar un servicio de búsqueda visual-semántica de productos de catálogos de tiendas en línea.

De lo anterior, se desprende el problema de relacionar texto en lenguaje natural con una imagen de un producto en un catálogo. Un problema similar ha sido caso de interés de muchos investigadores, quienes se han dedicado a estudiar cómo describir en palabras, el contenido visual de una imagen o video. Trabajos recientes [3], [7] plantean un enfoque diferente para la descripción textual de imágenes; esta solución propone traducir un texto a una imagen en vez de traducir una imagen a texto, este cambio de visión produce que se obtengan mejores resultados para el problema de la descripción autónoma de imágenes. Por lo tanto, el desafío del presente trabajo consta de investigar, implementar y analizar algunas de las distintas técnicas y modelos utilizados en la actualidad, para definir la mejor forma de relacionar texto e imagen para la realización de búsquedas de productos en catálogos de tiendas en línea.

En la imagen de la Figura 1.2a[3] se muestra un posible caso de uso de un buscador visual-semántico; dentro de la imagen se observa que el usuario publica un texto indicando que busca un par de zapatos, junto con una foto que muestra las características del producto que desea. Es posible notar que dentro del texto entregado solo es requerida la información del tipo de producto que busca (zapatos), el resto solo genera ruido en la búsqueda. Del mismo modo, en la imagen de la Figura 1.2b[4] se observa un caso de uso más favorable para el buscador ya que dentro del texto incluido en la publicación es posible extraer que el usuario no sólo busca un sombrero, si no que busca un sombrero de vaquero color rojo y que además sea mullido.

[2]Semantic search: `https://en.wikipedia.org/wiki/Semantic_search`

[3]Imagen obtenida desde Twiter: `https://twitter.com/ayewreckaa/status/978863088165416961`

[4]Imagen obtenida desde Twiter: `https://twitter.com/nameIessmel/status/1054171410514501632`

(a) Ejemplo de búsqueda de zapatos. (b) Ejemplo de búsqueda de un sombrero.

Figura 1.2: Ejemplos de búsquedas en redes sociales

Objetivos

Objetivo General

El objetivo general de este trabajo de título es estudiar, analizar e implementar técnicas actuales de machine learning en el área del procesamiento de lenguaje natural y reconocimiento visual, para desarrollar un servicio de búsqueda visual-semántico que encuentre dentro de los productos de un catálogo de una tienda, aquellos que concuerden con la semántica del texto entregado y tengan un parecido considerable con el objeto proporcionado en una imagen.

Objetivos Específicos

1. Definir el mejor descriptor de imágenes, tras la implementación de un buscador visual que utilice descriptores obtenidos desde 2 redes convolucionales distintas.
2. Diseñar, implementar y evaluar un algoritmo de búsqueda visual-semántico que utilice información textual y visual para encontrar un producto específico.
3. Entrenar una red neuronal multicapa (MLP) que realice una transformación de un descriptor de texto a un descriptor visual.
4. Analizar y comparar los resultados obtenidos para los buscadores visual y visual-semántico.

Capítulo 2

Marco Teórico

Dado el problema a resolver, se plantea el siguiente marco teórico, enmarcado principalmente en el área de la inteligencia computacional y la evaluación de sistemas de recuperación de información. Específicamente, en el presente capítulo se describen técnicas y modelos de machine learning de las áreas de procesamiento de lenguaje natural y reconocimiento visual. Además, se plantean métodos de comparación de vectores y métodos de evaluación de sistemas de recuperación de información, que son utilizados durante la implementación de los buscadores visual y visual-semántico.

2.1. Red Perceptrón Multicapa (MLP)

Una red de perceptrón multicapa (MLP), más conocido como red neuronal, es un esquema de computación distribuida inspirada en la estructura del sistema nervioso de los seres humanos. La arquitectura de esta red neuronal esta compuesta por la conexión de múltiples procesadores elementales, llamados perceptrones o neuronas; los cuales conforman un sistema adaptativo que posee un algoritmo para ajustar los pesos de cada conexión (Backpropagration) con el propósito de alcanzar los requerimientos de desempeño de un problema dado, mediante el uso de muestras representativas (conjunto de entrenamiento). Ahora bien, un sistema adaptativo es aquel que se va modificando acorde a los requerimientos del problema, según los resultados obtenido para un determinado conjunto de datos. Finalmente, una red neuronal es aquella que se compone por dos o más capas, donde cada una de ellas esta formada por la unión de uno o más procesadores elementales (neuronas). Es por ello que los componentes que conforman la arquitectura de una red de perceptrón multicapa son:

Perceptrón

Procesador elemental que tiene la capacidad de calcular una suma ponderada de sus entradas, para luego aplicar una función de activación y transmitir esta señal a otras neuronas.

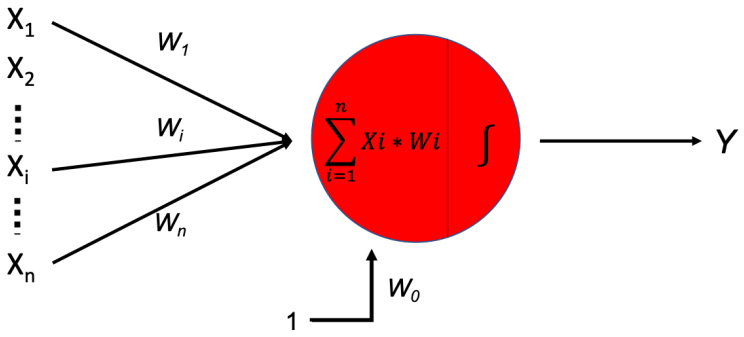


Figura 2.1: Esquema general de un perceptrón.

La Figura 2.1 muestra el esquema general de una neurona. En este esquema se observan las entradas (denotadas por los X_i), los pesos sinápticos (representados por los W_i), el bias (W_0), la sumatoria ponderada de las entradas, la función de activación (denotada por una integral que representa una función sigmoidea) y la salida (Y).

Capas

La unión de uno o más perceptrones conforman lo que se conoce como capa, existiendo diferentes tipos según su ubicación en la arquitectura de la red. Estas capas son:

- **Capa de entrada**: Esta capa esta compuesta por todas las neuronas que introducen los valores de entrada (vectores de características) de un caso, a la red neuronal.
- **Capa oculta**: Esta es una capa conformada por neuronas que reciben las características entregadas por la capa de entrada o capas anteriores y cuyas salidas son la entrada de capas posteriores o la capa de salida.
- **Capa de salida**: Compuesta por perceptrones cuyas salidas corresponden a la salida de toda la MLP.

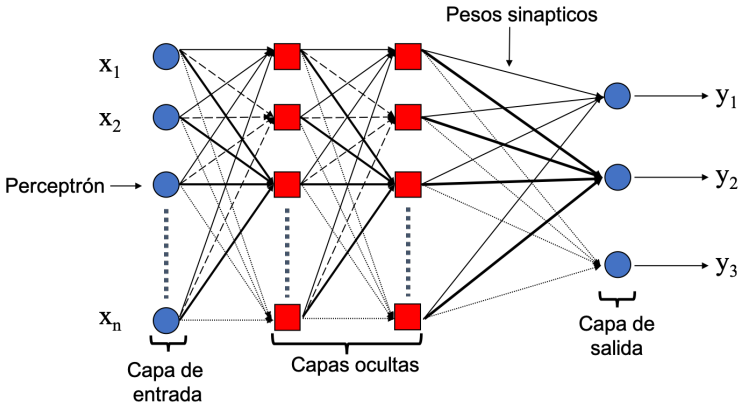


Figura 2.2: Esquema general de una MLP.

La Figura 2.2 muestra una MLP de cuatro capas (una capa de entrada, dos capas ocultas y una capa de salida). En ella se observan las entradas (X_i), los perceptrones representados por cuadrados rojos y circulos azules, las salidas (Y_i) y las distintas capas. Además, se observan las interconexiones entre neuronas.

Pesos

Los pesos son coeficientes que se adaptan dentro de la red y determinan la intensidad de la señal de entrada registrada por la neurona. Una neurona recibe múltiples entradas simultáneas. Cada entrada tiene su propio peso relativo que indica la importancia de la entrada dentro de la función de activación de la neurona. Estos pesos tienen el mismo propósito que las fuerzas sinápticas de las neuronas biológicas, es decir, transmitir información. En ambos casos, algunas entradas son más importantes que otras, de manera que tienen mayor efecto en el procesamiento de la respuesta neuronal.

Bias

El bias corresponde a un parámetro de entrada extra que cada neurona posee y es utilizado para ajustar la suma ponderada de las entradas agregando un término constante. Este valor se encuentra definido por el peso sináptico W_0, según la Figura 2.1.

Función de activación

La función de activación, como dice su nombre, es una función que cada neurona aplica al resultado de la sumatoria ponderada de las entradas para calcular la salida efectiva de la neurona. Las funciones de activación más utilizadas son las sigmoideas para la capa final y ReLU[1] para las capas intermedias.

2.2. Redes Neuronales Convolucionales

Las redes neuronales convolucionales son muy similares a las MLP, se componen de neuronas que tienen pesos y sesgos que deben ser aprendidos. Cada neurona recibe entradas, realiza una operación matemática y luego aplica una función de activación. Estas redes trabajan modelando de forma consecutiva pequeñas piezas de información, y luego combinándolas en las capas más profundas de la red.

Principalmente son utilizadas para el procesamiento de imágenes en problemas de reconocimiento visual, ya que se caracterizan por su buen desempeño en este tipo de tareas. Estas

[1]Kaggle (2018), *Rectified Linear Units (ReLU) in Deep Learning*, `https://www.kaggle.com/dansbecker/rectified-linear-units-relu-in-deep-learning`

redes se componen por 3 tipos de capas: capas de convolución, capas de pooling y una capa clasificadora que es completamente conectada. Cada una de estas capas cumple un propósito distinto. Al principio de la red se encuentran las capas de extracción de características (capas de convolución), sigue una capa de reducción de muestreo (capa de pooling) y al final de la red se encuentra una MLP para realizar la clasificación final sobre las características extraídas.

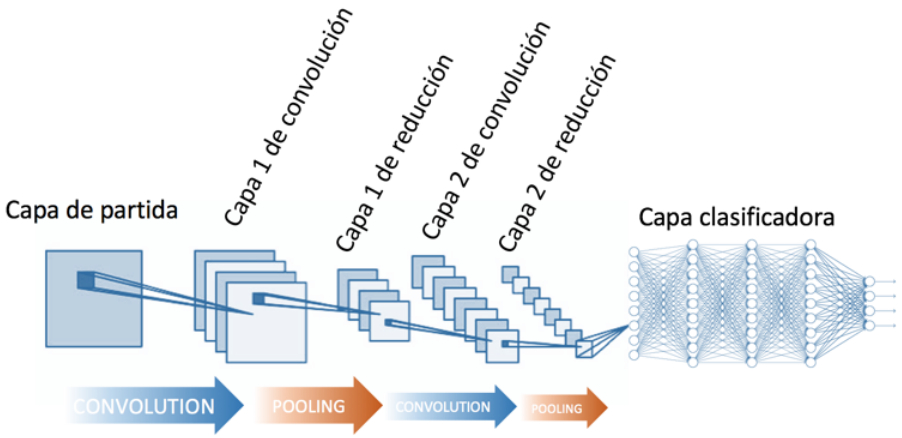


Figura 2.3: Arquitectura general de una ConvNet.[2]

Capa de convolución

Esta capa se encarga de aplicar la operación llamada convolución. Recibe como entrada una serie de matrices (tensores) que se deben procesar y le aplica la operación mencionada con una serie de filtros (kernels), obteniendo como resultado un mapa de características de la entrada respectiva. En esta capa, las neuronas son representadas por los filtros y por tanto no reciben un valor numérico único, sino que tensores a los cuales aplicar la convolución. La salida de cada neurona de esta capa se calcula con la siguiente ecuación:

$$Y_j = g\left(b_j + \sum_i K_{ij} \otimes Y_i\right) \tag{2.1}$$

Donde Y_j, que representa la salida de la neurona j, es una matriz que se calcula por medio de la combinación lineal de las salidas Y_i de las neuronas en la capa anterior cada una de ellas operadas con el kernel de convolución K_{ij} correspondiente a esa conexión. La cantidad obtenida es sumada a un bias b_j y luego se aplica la función de activación $g(\cdot)$.

Aplicar la operación de convolución produce como resultado filtrar la entrada con un kernel previamente entrenado. Esto transforma los datos de tal manera que ciertas características

[2]Diego Calvo (Julio de 2017), *Red Neuronal Convolucional CNN*, `http://www.diegocalvo.es/red-neuronal-convolucional/`

(determinadas por la forma del kernel) se vuelven más dominantes en la salida. Estos kernel tienen habilidades de procesamiento específicos, como por ejemplo, la detección de bordes dentro de una imagen. Sin embargo, los kernel que son entrenados por una red convolucional son más complejos con el objetivo de extraer características más abstractas y no triviales[3].

Capa de pooling

Esta capa, llamada también capa de reducción de muestreo, se ubica generalmente después de una capa convolucional y cumple la función de disminuir las dimensiones espaciales (ancho x alto) de la entrada. Su objetivo es reducir la cantidad de parámetros y por tanto, de computo, dentro de la red convolucional, así como también de extraer características más generales de los patrones presentes en la entrada. Un ejemplo de implementación de la capa de pooling consiste en la aplicación de la operación MAX en sub espacios de la matriz de entrada. A esta aplicación se le conoce como max-pooling y su procedimiento se puede analizar de mejor manera en la siguiente imagen, donde en cada sub-cuadrado de la matriz original se seleccionan los valores más altos[4] [5].

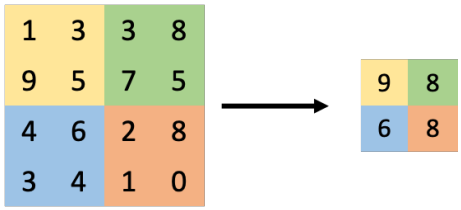


Figura 2.4: Ejemplo de aplicación de max-pooling

La imagen de la Figura 2.4 muestra la aplicación de un max-polling con un filtro de 2×2 y un paso de 2, a una matriz de 4×4 obteniendo una matriz de 2×2 como resultado de la operación.

Capa completamente conectada

La característica principal de esta capa, es que las neuronas que la conforman, poseen conexiones a todas las salidas de la capa anterior. Por otra parte las neuronas de esta capa se comportan de igual modo que las neuronas de una MLP y por tanto su salida esta dada por la siguiente formula:

[3]Diego Calvo (Julio de 2017), *Red Neuronal Convolucional CNN*, `http://www.diegocalvo.es/red-neuronal-convolucional/`

[4]*Convolitional Neural Network for Visual Recognition*, `http://cs231n.github.io/convolutional-networks/`

[5]Raul E. Lopez Briega (Agosto de 2016), *Redes neuronales convolucionales con TensorFlow*, `http://relopezbriega.github.io/blog/2016/08/02/redes-neuronales-convolucionales-con-tensorflow/`

$$y_j = g\left(b_j + \sum_i w_{ij} \cdot x_i\right) \tag{2.2}$$

Donde y_j, que representa la salida de la neurona j, es un valor que se calcula por medio de la combinación lineal de las salidas x_i de las neuronas en la capa anterior cada una de ellas multiplicadas con un peso w_{ij} correspondiente a la conexión respectiva. Esta cantidad es sumada a un bias b_j y luego se aplica la función de activación $g(\cdot)$.

2.2.1. VGG

La red convolucional VGG fue diseñada por Karen Simonyan y Andrew Zisserman, y publicada a principios del año 2015 en la investigación denominada "*Very Deep Convolutional Networks for Large-Scale Image Recognition*". En la época en la que fue diseñada esta red, el estado del arte dentro de los problemas de reconocimiento de imágenes eran las redes convolucionales. Sin embargo, los resultados dentro de la competencia ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) aún se podían mejorar. Por otra parte, las redes utilizadas hasta ese entonces no eran muy profundas y los filtros utilizados para la convolución eran más grandes que los implementados actualmente.

Dentro de la investigación, proponen una red más profunda (16-19 capas) en las cuales los filtros utilizados son de menor tamaño (3×3). De este modo, innovan en lo que es la arquitectura de una red convolucional. En la imagen de la Figura 2.5, se observa la arquitectura de la red VGG. En ella se muestran la cantidad de filtros utilizados en cada una de las capas y los tamaños de las imágenes resultantes luego de cada capa de polling.

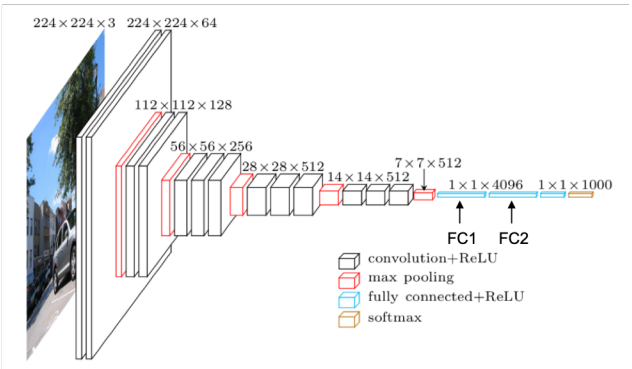


Figura 2.5: Arquitectura de la red VGG

Cabe destacar, que dentro de las capas de convolución utilizan filtros de 1×1 para realizar una transformación de las imágenes, utilizan ventanas de tamaño 2×2 con paso 2 para realizar los max-pooling entre capas, disminuyendo a la mitad el tamaño de la imagen y aumentando en 2 la cantidad de filtros, y utilizan filtros de tamaño 3×3 para capturas las nociones de izquierda/derecha, arriba/abajo y centro, dentro de las imágenes. Finalmente, la red cuenta con 2 capas fully connected que permiten realizar el proceso de clasificación, obteniendo como salida final, un vector de 1000 dimensiones correspondiente a la cantidad de clases de

la tarea de clasificación de la competencia en la que se participó. Por otro lado, las capas fully connected antes mencionadas, son aquellas desde las cuales es posible obtener un mapa de características de una imagen procesada por esta red.

Esta nueva arquitectura de red convolucional, les permite obtener el primer puesto en la competencia de localización y clasificación en ImageNet del año 2014, conformando de este modo el estado del arte en el reconocimiento de imágenes.

2.2.2. ResNet

La red convolucional ResNet fue diseñada por Kaiming He, Xiangyu Zhang, Shaoqing Ren y Jian Sun, y publicada a fines del año 2015 en la investigación denominada "*Deep Residual Learning for Image Recognition*". Esta red fue diseñada para resolver un problema que muchas redes convolucionales profundas estaban sufriendo para ese entonces, el costo del entrenamiento y el desvanecimiento del gradiente en la propagación del error durante el aprendizaje. Es por ello que dentro de su investigación proponen una nueva forma de aprendizaje que facilita el entrenamiento de redes convolucionales profundas, reformulando el diseño de las capas de convolución.

Dentro de la investigación proponen la utilización de bloques de convolución compuestos por 2 capas, en la cual se propaga la entrada del bloque, hacia la salida de este. En la imagen de la Figura 2.6, se observa la arquitectura de un bloque de convolución y la propagación de la identidad hacia la salida del bloque de convolución. La función $\mathcal{F}$, representa la función residual de la salida de una capa de convolución, la cual debe ser aprendida durante el entrenamiento.

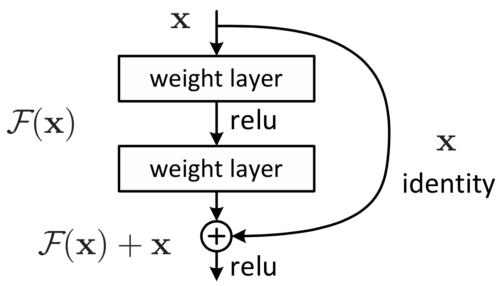


Figura 2.6: Bloque de convolución de la red convolucional ResNet

Por otra parte, se define la salida de un bloque por la formula de la ecuación 2.4, donde σ representa la función de activación ReLu y W_1, W_2 representan los pesos de las capas 1 y 2 del bloque de convolución.

$$y = \mathcal{F}(x, W_i) + x \tag{2.3}$$

$$y = W_2\sigma(W_1 x) + x \tag{2.4}$$

Esta nueva técnica de aprendizaje, permite aumentar la cantidad de capas de convolución de las redes, llegando a superar hasta por ocho veces las redes más profundas hasta ese entonces y manteniendo una complexidad menor, es decir menos operaciones de punto flotante.

Además, esta nueva arquitectura logra obtener el primer puesto en la competencia de clasificación ILSVRC del año 2015 y en las competencias de: detección en ImageNet, localización en ImageNet, detección en COCO y segmentación en COCO del mismo año. Es por ello que esta arquitectura pasa a conformar el estado del arte en el reconocimiento de imágenes.

En la Figura A.2, se muestran las distintas arquitecturas de redes convolucionales ResNet. En ella se observa que a medida que la imagen se adentra en la red, la cantidad de filtros de convolución aumenta y el tamaño de la imagen disminuye, llegando a una salida de 2048 filtros con una imagen de tamaño 1×1 para el caso de la red ResNet50.

2.3. Deep Features

Un deep feature, es aquel vector obtenido tras la intercepción de la salida de un modelo jerárquico como lo son las MLP o las ConvNet. Ahora bien, un vector es considerado más profundo, dependiendo de que tan pronto es interceptada la red utilizada para procesar el objeto.

Para el caso de las redes diseñadas para problemas de reconocimiento visual, estos vectores son capaces de extraer caracteristicas abstrartas de los elementos presentes en la imagen. En la Figura 2.3 se observa la arquitectura general de una ConvNet. Esta se divide en 2 partes: una que permite la extracción de características y otra que permite la clasificación. En particular, los deep features de imágenes se obtienen de interceptar la ConvNet en la capa de entrada de la capa clasificadora.

2.4. Bag of Words

Bag of Words[6] (BoW) es un modelo de vectorización de texto que consiste en almacenar en un vector, la cantidad de apariciones de una palabra dentro de una frase. Es importante mencionar que cada una de las posiciones del vector, corresponde a una palabra específica presente en el vocabulario definido. Por otra parte, BoW describe un documento según las palabras que contiene, no representando la similitud semántica entre palabras. Es por ello que por ejemplo: "zapato" y "sombrero", tienen una distancia igual a "zapatos" y "botas", pese a que estas dos últimas poseen mayor similitud semántica.

[6]Wikipedia, Bag-of-words model (BoW), `https://en.wikipedia.org/wiki/Bag-of-words_model`

2.4.1. TF-IDF

TF-IDF es una variación del modelo BoW que pondera la frecuencia de aparición de un término en la oración (TF) por la frecuencia inversa de aparición de dicho término en el total de documentos (IDF). Formalmente, cada una de estas medidas se calcula de la siguiente manera:

$$tf(t,d) = 1 + \log(f(t,d)) \tag{2.5}$$

$$idf(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{2.6}$$

Donde $f(t,d)$ corresponde a la frecuencia absoluta de aparición del término t dentro del documento d, $|D|$ corresponde al total de documentos y $|\{d \in D : t \in d\}|$ es la cantidad de documentos distintos en los que se encuentra presente el término t. De este modo:

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \tag{2.7}$$

2.5. Word Embedding

Word embedding es una técnica de procesamiento de lenguaje natural, que permite la asignación inyectiva de texto a un vector multi-dimensional, mediante la implementación de un modelo matemático. Se caracteriza por tener la capacidad de capturar el contexto y la similitud entre palabras.

Esta técnica apunta a cuantificar y categorizar las semánticas entre palabras, basándose en las características presentes en la distribución de ellas dentro de una muestra grande de datos de un lenguaje. La idea detrás de esta técnica es que una palabra está caracterizada por las palabras que la acompañan, definiendo así su contexto de uso, su significado y su similitud con otros elementos del vocabulario.

2.5.1. Word2Vec

Word2Vec [12] es un modelo de word embedding basado en redes neuronales, cuyo enfoque es extraer el significado semántico de las palabras, a través del contexto en el cual son usadas. El objetivo es generar representaciones cercanas para palabras que tengan un significado semántico similar y representaciones distantes para aquellas palabras que no comparten contexto.

A continuación se presenta el modelo skipgram, el cual corresponde al modelo usado en [12]. Este modelo se basa en el uso de redes neuronales con una capa oculta, llamada capa de proyección, debido a que es la encargada de aprender la transformación adecuada de una palabra a un vector. El objetivo del modelo es, dado un conjunto de entrenamiento, entrenar la red intentando predecir las palabras cercanas a una palabra dada.

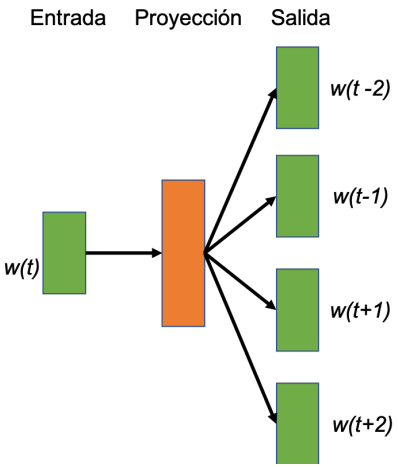


Figura 2.7: Arquitectura del modelo skipgram. El objetivo del entrenamiento es aprender una representación que permita predecir las palabras cercanas.

La imagen de la Figura 2.7, muestra la arquitectura general del modelo donde se observa la entrada (una palabra dentro del documento de entrenamiento) y su salida (las palabras anteriores y posteriores a la palabra de entrada en la ventana respectiva). El entrenamiento de este modelo consta de, dado un documento y un tamaño de ventana de entrenamiento, extraer las palabras de la ventana y entregar dichos datos como ejemplo de entrenamiento a la red. En la Figura 2.8, se muestran los datos de entrenamiento obtenidos de un documento de ejemplo.

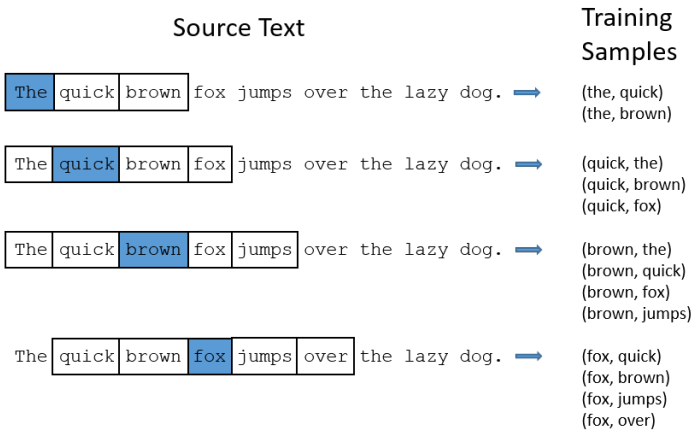


Figura 2.8: Ejemplo de obtención de datos de entrenamiento.[7]

Una vez entrenado el modelo es posible generar los vectores de las palabras presentes en los documentos de entrenamiento, generando un espacio multidimensional que mantiene la semántica de las palabras. En la imagen de la Figura 2.9, se observa una representación gráfica de las distancias entre palabras, donde se destaca una similitud de género según la distancia de los vectores, así al realizar una operación como: `king - man + woman`, se obtiene como resultado: `queen`.

[7]Gonzalo Ruiz de Villa (Mayo de 2018), *Introducción a Word2vec (skip gram model)*, `https://medium.com/@gruizdevilla/introducción-a-word2vec-skip-gram-model-4800f72c871f`

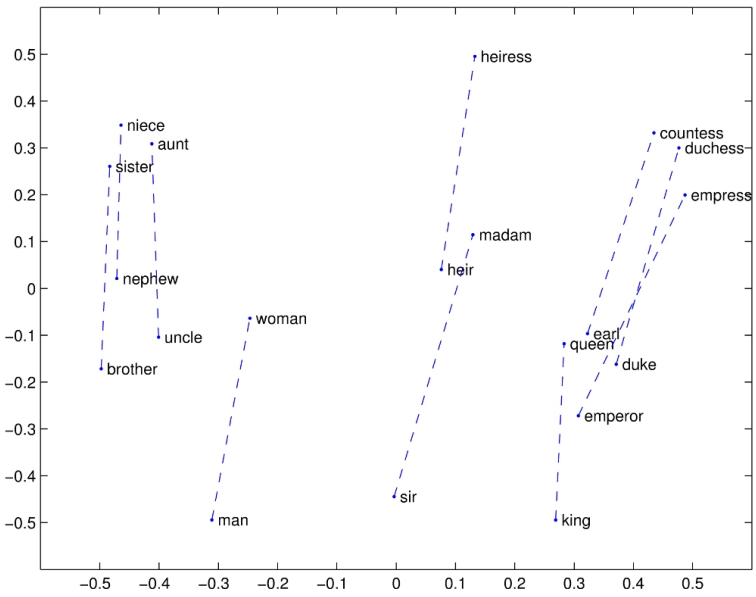


Figura 2.9: Ejemplo vectorización de palabras utilizando Word2Vec

2.5.2. FastText

FastText [1] es un modelo de word embedding cuyo enfoque está en la representación eficiente de texto en lenguaje natural mediante un entrenamiento no supervisado, que además ha sido implementado para trabajar con distintos lenguajes.

Un n-grama corresponde a un conjunto de subsecuencias de n caracteres de una palabra dada. El modo por el cual fastText realiza la representación vectorial consta de: primero incorporar los símbolos "<" y ">" al comienzo y final de la palabra, posteriormente extraer los n-gramas de una palabra dada, consiguiendo así un conjunto G de tamaño g, para finalmente calcular el centroide de G realizando un promedio de sus elementos, obteniendo de este modo el descriptor de la palabra original.

n-grama	camión
3-grama	<ca, cam, ami, mió, ión, ón>
4-grama	<cam, cami, amió, mión, ión>
5-grama	<cami, camió, amión, mión>
	<camión>

Figura 2.10: Ejemplo de extracción de n-gramas para la palabra "camión".

En la Figura 2.10, se observa un conjunto G de tamaño 15 correspondiente a la unión de 3 n-gramas con valores de n iguales a 3, 4 y 5. Se define G_i como el elemento i del conjunto G. Por lo tanto, la vectorización de la palabra "camión" estará dada por la formula 2.8, según la definición dada en [1]

$$embedding(camion) = \frac{\sum_{i=1}^{15} embedding(G_i)}{15} \tag{2.8}$$

Para la obtención de la vectorización de una frase, primero se calculan los vectores de cada una de las palabras que la componen, posteriormente se normalizan y finalmente se promedian.

2.6. Indexación Semántica Latente (LSI o LSA)

Latent Semantic Indexing (LSI) o también conocido como Latent Semantic Analisis (LSA), es un proceso cuyo propósito es eliminar términos correlacionados para reducir la cantidad de términos relevantes dentro del vocabulario utilizado para representar los documento de un conjunto dado. Este proceso consta de 4 etapas:

1. Calcular los descriptores de los n documentos entregados.
2. Utilizar el método SVD para factorizar la matriz de descriptores a su equivalente $A = USV^T$.
3. Seleccionar los k valores singulares de mayor magnitud y reducir la dimensión de las matrices de r a k
4. Convertir los descriptores de documentos a su representación reducida utilizando la matriz de transformación $U'S'^{-1}$

La primera etapa, consiste en calcular los descriptores de documentos utilizando un modelo de vectorización One Hot Encodding, por ejemplo BoW con TF-IDF, para obtener la matriz A de dimensiones $t \times n$, donde los descriptores de dimensión t se encuentran almacenados como columnas. En la segunda etapa, se debe utilizar el método SVD para factorizar la matriz de descriptores a su equivalente $A = USV^T$, donde U es de dimensiones $t \times r$, S es una matriz diagonal de dimensiones $r \times r$ y V es de dimensiones $n \times r$. Esta representación permite seleccionar los k valores singulares de mayor magnitud, es decir que maximizan la cantidad de varianza del conjunto.

Luego de seleccionar los k valores presentes en la matriz S y reducir la dimensión de las matrices de r a k, se obtiene una matriz reducida $A' = U'S'V'^T$ donde las dimensiones de las matrices U', S' y V' son $t \times k$, $k \times k$ y $n \times k$. Finalmente, es posible convertir los descriptores de documentos a su representación reducida mediante la matriz de transformación calculada, es decir, $A' = A^T U' S'^{-1}$.

Una vez calculadas las matrices mencionadas, es factible transformar vectores de consulta q a su versión reducida mediante la multiplicación por la matriz de transformación encontrada. De este modo el vector q' se define como: $q' = qU'S'^{-1}$.

La aplicación de LSA dentro de un conjunto de documentos, permite una mejor representación de estos al usar conceptos semánticos en vez de términos o palabras. Sin embargo, conlleva un costo adicional en el cálculo de las matrices de transformación, ya que dependiendo del tamaño del vocabulario, puede haber operaciones entre matrices muy grandes. Además, incorpora un costo adicional de memoria, dado que los vectores dejan de ser dispersos.

2.7. Función de similitud

Las funciones de similitud son funciones que permiten medir el grado de cercanía que existe entre 2 documentos u objetos descritos por su vector correspondiente. Así mismo, una función de disimilitud mide el grado de lejanía entre 2 vectores. Existen distintas propiedades que se deben cumplir para que una función de disimilitud pueda ser definida como métrica, estas son:

1. No-negatividad $disim(x, y) >= 0$
2. Reflexividad $disim(x, y) = 0 <=> x = y$
3. Simetría $disim(x, y) = disim(y, x)$
4. Desigualdad triangular $disim(x, z) <= disim(x, y) + disim(y, z)$

Existen diversas funciones de similitud que permiten comparar el contenido de un descriptor, entre ellas: las distancias de Minkowski, donde la más conocida es la distancia Euclidiana, y la similitud coseno.

2.7.1. Distancias de Minkowski

Las distancias de Minkowski (L_p) son métricas que se calculan mediante la formula de la ecuación 2.9

$$L_p(U, V) = \left(\sum_{i=0}^{n} |U_i - V_i|^p \right)^{\frac{1}{p}} \tag{2.9}$$

Cuando $p = 2$ esta métrica pasa a ser la distancia Euclidiana, descrita por la formula de la ecuación 2.10. A pesar de que esta métrica permite calcular la distancia de 2 puntos en un espacio de N dimensiones, es poco utilizada como medida de similitud entre descriptores de texto, ya que pierde el sentido semántico de estos al asignar distancias iguales a vectores que poseen definiciones distintas.

$$L_2(U, V) = \sqrt{\sum_{i=0}^{n} (U_i - V_i)^2} \tag{2.10}$$

2.7.2. Similitud coseno

La similitud coseno es una medida de la distancia entre dos vectores U, V que evalúa el coseno del ángulo comprendido entre ellos. Esta medida es utilizada para la implementación de sistemas de recuperación de información, como también para evaluar la semejanza semán-

tica entre textos vectorizados mediante alguno de los modelos antes mencionados. Se destaca por su rendimiento y eficiencia, ya que consiste en calcular el producto punto entre ambos vectores y luego dividir este resultado por la norma de cada vector. La formula de la ecuación 2.11, muestra cómo calcular la similitud coseno entre 2 vectores de N dimensiones.

$$sim(U, V) = \frac{U \cdot V}{||U||\ ||V||} = \frac{\sum_{i=0}^{n} U_i V_i}{\sqrt{\sum_{i=0}^{n} {U_i}^2}\sqrt{\sum_{i=0}^{n} {V_i}^2}} \tag{2.11}$$

Cuando los vectores U, V se encuentran normalizados, el cálculo de la similitud coseno es aún más simple, ya que el divisor de la ecuación anterior pasa a ser 1 y por ende sólo es requerido calcular el producto punto entre ambos vectores.

$$sim(U, V) = U \cdot V = \sum_{i=0}^{n} U_i V_i \tag{2.12}$$

Esta medida de similitud se encuentra dentro del rango $[-1, 1]$, donde 1 indica similitud absoluta y -1 indica que ambos vectores son totalmente opuestos. Cuando los vectores se encuentran en un espacio positivo, es decir que cada una de sus componentes es mayor que 0, la medida de similitud se encuentra dentro del rango $[0, 1]$ donde 0 indica disimilitud absoluta (vectores ortogonales) y 1 indica similitud absoluta (vectores idénticos).

2.8. Métodos de Evaluación

Para los sistemas de recuperación de información existen diversas métricas para medir la efectividad. El modo por el cual estos son evaluados consta de obtener una lista de k elementos ordenados por relevancia, para luego calcular la precisión y la exhaustividad (*recall*) de la consulta realizada. La precisión mide la proporción de elementos correctos entregados sobre la totalidad de los elementos entregados por el sistema, es decir C/A, según el diagrama de la Figura 2.11. Por otro lado, la exhaustividad mide la proporción de elementos correctos entregados sobre la totalidad de elementos correctos esperados, es decir C/B.

[7]Jiawei Han, Micheline Kamber, Jian Pei (2012), *Data Mining (Third Edition)*, `https://www.sciencedirect.com/topics/computer-science/cosine-similarity`

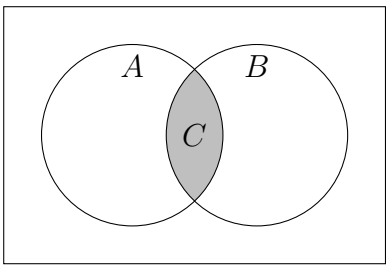


Figura 2.11: Diagrama de consulta de un sistema de recuperación. El conjunto A representa las respuestas entregadas para una consulta y el conjunto B las respuestas que se esperan recibir.

Hay ocasiones donde es requerido analizar cómo cambia la exhaustividad en función de la precisión o viceversa, para esto se calculan ambas métricas para cada uno de los elementos correctos encontrados en las respuestas entregadas por el sistema, siguiendo el modelo de la Figura 2.12.

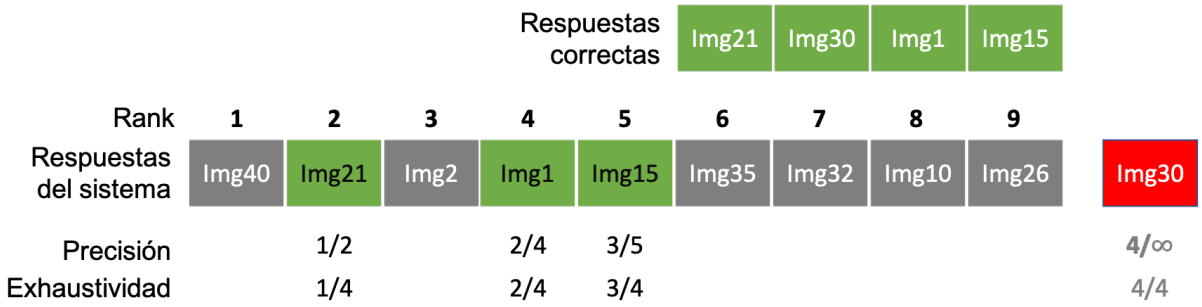


Figura 2.12: Ejemplo de consulta, cálculo de precisión y cálculo de exhaustividad

2.8.1. Average Precision at Rank (AP@r)

El AP@r es una métrica obtenida de promediar los Precision at Rank (P@r) de las consultas hechas a un sistema de recuperación de información. El P@r se calcula como $\frac{p}{r}$ donde p corresponde a la cantidad de documentos relevantes entre las posiciones 1 y r, y r es la posición de interés. De este modo, para el ejemplo de la Figura 2.12, el P@5 es el siguiente:

$$P@5 = \frac{|\{Img21, Img1, Img15\}|}{5} = \frac{3}{5} = 0{,}6 \tag{2.13}$$

2.8.2. Mean Average Precision (MAP)

El MAP es una métrica para la evaluación de un sistema que consta de promediar los Average Precision (AP) de todas las consultas realizadas al sistema. El AP corresponde al promedio de las precisiones de una consulta calculadas, según el modelo de la Figura 2.12, así para el ejemplo mostrado el AP es el siguiente:

$$AP = \frac{\frac{1}{2} + \frac{2}{4} + \frac{3}{5} + \frac{4}{\infty}}{4} = \frac{0{,}5 + 0{,}5 + 0{,}6 + 0}{4} = 0{,}4 \tag{2.14}$$

Dado lo anterior, mientras más cercano a 1 sea el MAP mejor es el sistema de recuperación.

2.8.3. Mean Reciprocal Rank (MRR)

El MRR es una métrica que consiste en promediar los Reciprocal Rank (RR) de las distintas consultas realizadas al sistema, donde el RR es la precisión del primer elemento correcto encontrado. Para el ejemplo de la Figura 2.12, el RR es:

$$RR = \frac{1}{2} = 0{,}5 \tag{2.15}$$

Dentro de esta métrica, es posible restringir la máxima posición donde encontrar una respuesta correcta. De este modo se hace más estricta la evaluación del sistema, ya que si el primer elemento encontrado excede la posición máxima definida, el RR es 0 para esa consulta.

Capítulo 3

Revisión bibliográfica

3.1. Estado del Arte

Para el estudio, desarrollo e implementación del trabajo de memoria, se utiliza como guía una investigación reciente que tiene por objetivo crear un modelo predictivo que permita describir el contenido visual de una imagen [7]. Dentro de esta investigación proponen traducir el texto de una descripción a un descriptor visual que represente la imagen descrita, planteando de esta manera una forma distinta de abordar el problema de la descripción de imágenes. Este cambio de foco genera resultados más precisos (según los experimentos realizados) en comparación con las investigaciones actuales que proponen realizar el proceso inverso [5], es decir, traducir una imagen a su descripción correspondiente. Es por ello que Word2VisualVec [7] pasa a conformar el estado del arte dentro de la descripción autónoma de imágenes.

Dentro de Word2VisualVec utilizan distintas técnicas de aprendizaje de maquina, entre ellas se encuentra el uso de redes neuronales convolucionales (Convolutional Neural Networks o ConvNet) [2], [8], [11], [13], redes neuronales multicapa (MLP) y distintos modelos de word embedding [6], [12], [1]. Es importante destacar que las ConvNet constituyen el estado del arte dentro de lo que es el análisis visual, ya que permiten la extracción de características propias de una imagen. Por otro lado, dentro de ellas existen distintas arquitecturas, unas más profundas que otras, pero que sin embargo comparten la estructura general, es decir, la utilización de capas de convolución, capas de pooling y capas completamente conectadas.

Word2VisualVec propone trabajar dentro del espacio de los descriptores visuales; y para esto es requerido traducir un vector de texto a un vector visual procurando mantener las características propias de la imagen que se indican en la oración respectiva. Para lograr lo anterior, su trabajo lo dividen en 3 partes: la primera corresponde a la extracción de características presentes en una imagen mediante el uso de ConvNets, la segunda es realizar una vectorización de un texto en lenguaje natural y la tercera consiste en entrenar una MLP que realice una regresión lineal para traducir un vector de texto a un descriptor visual.

En la primera etapa, son utilizadas distintas arquitecturas de ConvNet, entre ellas se encuentran CaffeNet [13], GoogLeNet [2], GoogLeNet-shuffle [11] y ResNet-152 [8], siendo estas 2 últimas aquellas que generan mejores resultados en la precisión de la descripción de imágenes, según los experimentos realizados en su investigación. En la segunda etapa utilizan distintos modelos de word embedding, entre estos se encuentra Word2Vec [12], Bag of Words (BoW), Gated Recurrent Unit [6] (GRU) como red recurrente especializada en vectorización de sentencias y un modelo multi-escala que combina los 3 modelos antes mencionados. Finalmente, para el entrenamiento y la realización de pruebas de precisión de la MLP que traduce un descriptor de texto a un descriptor visual, utilizan el conjunto de datos Flickr8k [9] y Flickr30k [10] que consisten en imágenes con distintas descripciones del contenido de ellas.

Dentro de la investigación guía no se utiliza el modelo de word embedding FastText, sin embargo dentro de lo que es el procesamiento de lenguaje natural, se caracteriza por su buen desempeño, su gran eficiencia y la capacidad de ser efectivos pese a enfrentarse con palabra con errores ortográficos[1]. Del mismo modo, la red convolucional VGG no fue considerada en la investigación, sin embargo ha sido destacada por su desempeño en las competencias de clasificación ILSVRC. Por ende, se evalúan ambos modelos durante el trabajo realizado.

Una investigación reciente[4] cuyo enfoque se encuentra en la recuperación de videos a través de textos descriptivos, proponen un nuevo método de vectorización de 3 niveles para sentencias y videos. Este método de vectorización busca obtener un descriptor global para ambos casos, que represente de la mejor forma las características de cada uno. Posteriormente, los descriptores obtenidos los proyectan en un espacio común, para de este modo encontrar la relación entre videos y sentencias.

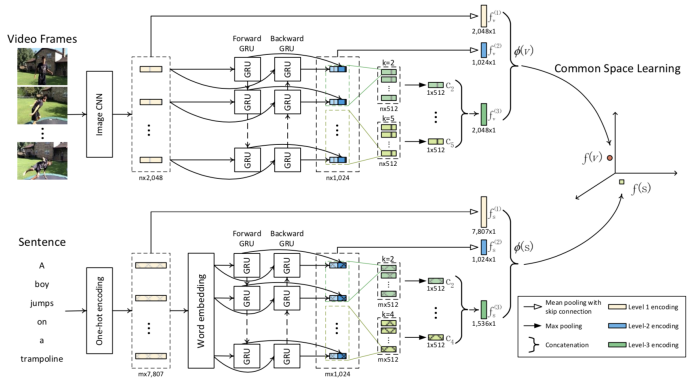


Figura 3.1: Arquitectura general del modelo dual encodding

El método de codificación del texto consta de: vectorizarlo en un primer nivel mediante un módelo One Hot Encodding (BoW), luego en el segundo nivel obtener los word embedding de cada palabra y procesarlos en una red neuronal recurrente (GRU), para en el tercer nivel procesar la salida de la GRU con redes convolucionales. Finalmente, el descriptor global del texto consiste en la concatenación de las salidas de los niveles 1, 2 y 3, es decir, la concatenación del vector obtenido con BoW, la salida de la GRU y la salida de las redes convolucionales. En la imagen de la Figura 3.1 se observa la arquitectura general del modelo dual encodding propuesto en la investigación descrita, desde la cual se realiza una adaptación para la vectorización de oraciones.

[1] `https://research.fb.com/fasttext`

Capítulo 4

Descripción e Implementación de la Solución

4.1. Descripción general de la solución

El trabajo consiste en el estudio, análisis e implementación de nuevas técnicas de aprendizaje de máquina en el área del procesamiento de texto e imágenes. El propósito es determinar si existe una relación entre ambas que permita mejorar la precisión de los resultados de una búsqueda de productos en línea utilizando ambas características, es decir, texto descriptivo e imagen asociada. Es por ello que la solución desarrollada consta de: calcular descriptores de texto, calcular descriptores de imágenes, construir y entrenar una red neuronal multicapa que transforme un descriptor de texto en un descriptor visual, implementar un buscador por imágenes y un buscador visual-semántico (texto-imagen) y evaluar ambos buscadores frente a un conjunto de consultas pre-definidas, para analizar los resultados.

4.2. Implementación

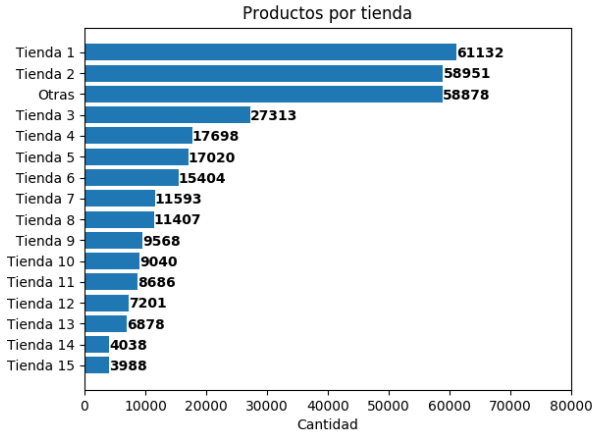
Para la implementación de la solución, es vital contar con datos que permitan entrenar distintos modelos de machine learning, junto con evaluar y comparar resultados. Por otra parte, dadas las condiciones del problema, estos datos consisten en pares texto-imagen que almacenan el título del producto y su imagen correspondiente.

4.2.1. Conjuntos de datos

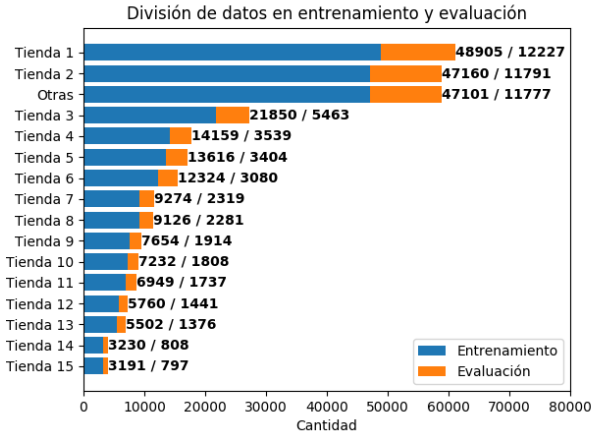
Para la implementación del sistema fueron recopilados un total de 329.399 datos de productos, dentro de los cuales 604 tuvieron que ser descartados debido a la ausencia de una imagen asociada o debido a que la imagen del producto se encuentra dañada. Es por ello que se cuenta con 328.795 datos, pertenecientes a 142 tiendas comerciales.

Una vez obtenidos los datos de cada producto, se prosigue con el procesamiento y limpieza de estos. El primer paso consiste en la eliminación de caracteres no alfanuméricos dentro de los títulos de cada producto, utilizando una expresión regular que reemplaza cada caracter no deseado por un espacio en blanco. La siguiente etapa consiste en la traducción de los títulos de cada producto al español, debido a la existencia de un subgrupo de datos que pertenece a tiendas extranjeras cuyos productos se encuentran descritos en inglés. Para realizar lo anterior se utiliza la función de traducción de las hojas de cálculo de Google Sheets. Finalmente, los productos son divididos en 2 conjuntos, uno de entrenamiento (80 %) y uno de evaluación (20 %), procurando seleccionar una proporción similar de productos de cada tienda. Esta división es realizada con el propósito de evitar el overfiting de los modelos que se utilizan, esperando obtener resultados generales sobre los cuales analizar su rendimiento.

En el gráfico de la Figura 4.1, se aprecia la distribución de productos por tienda, junto con la división de los datos en los conjuntos de entrenamiento y evaluación descritos anteriormente. Cabe mencionar que dentro de la categoría Otros se encuentran todas aquellas tiendas que no entran en el ranking de las 15 tiendas con más productos.



(a) Número de productos por tienda



(b) División del conjunto de datos en entrenamiento y evaluación

Conjunto	**Cantidad**
Entrenamiento	263033
Evaluación	65762
Total	**328795**

(c) Total de elementos por subconjunto de datos

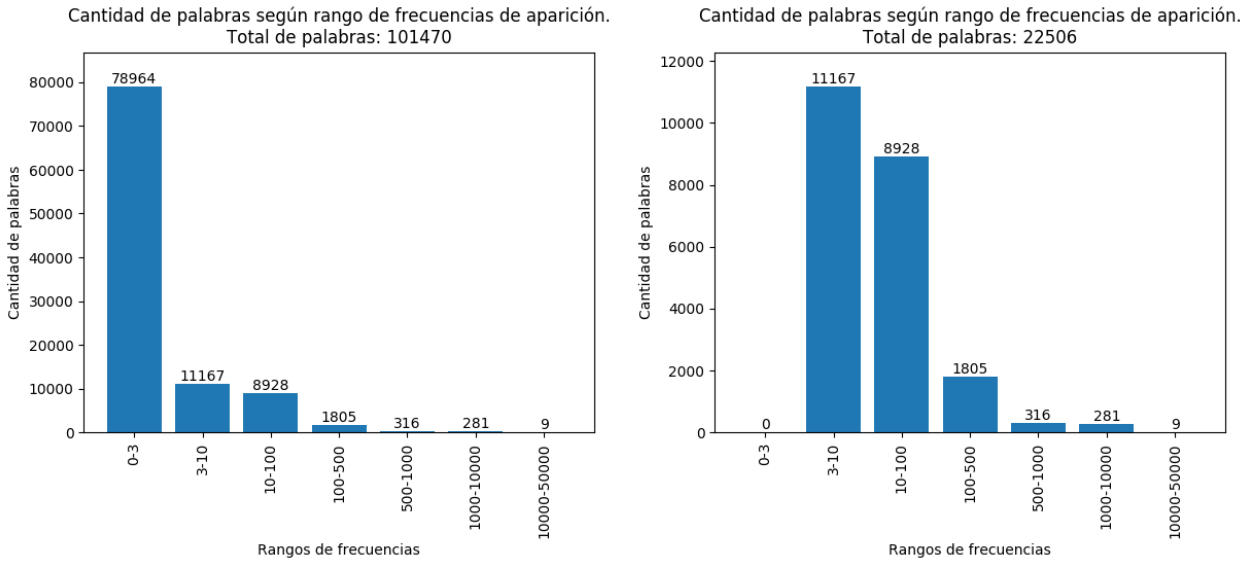
Figura 4.1: Conteo de productos por tienda y división del conjunto de datos

En la imagen de la Figura 4.2 se observan algunos ejemplos de datos pertenecientes al conjunto de entrenamiento. Dentro de lo observado se destaca la presencia de imágenes con una alta calidad que muestran claramente el objeto de interés. Sin embargo, el título del producto no posee las mismas características que las imágenes, ya que poseen una gran cantidad de palabras poco relevantes, como por ejemplo: el código del producto, el nombre de la tienda, las dimensiones del producto, entre otros.

Imagen	Texto	Imagen	Texto
	hervidor de agua electrico tetera 1 8l apagado automatico 3g		reloj naviforce nf9074m correa de pu reloj de hombre red
	arbol navidad 3 mt brunswick dear santa		bolso grande duffle cuerda ruedas negro 30 envio gratis
	mini parlante bluetooth inalambrico gatita		sandalia hombre negro art 8lx2727blac via franca

Figura 4.2: Ejemplos de productos con su respectivo título e imagen

Una vez separado el conjunto en datos, se prosigue a encontrar y eliminar las stopwords de los títulos del conjunto de datos de entrenamiento. Este proceso consiste de dos partes: la primera corresponde a la eliminación de las stopwords generales del lenguaje, es decir artículos, conectores, pronombres, etc; este conjunto de palabras fue obtenido desde github[1]. La segunda etapa corresponde a determinar las stopwords específicas del conjunto de datos de entrenamiento, para ello se comienza por calcular el vocabulario completo de los títulos de los productos, obteniendo un total de 101.470 palabras. Debido al gran tamaño del vocabulario se opta por realizar una reducción. Para ello se calcula la frecuencia de aparición de cada palabra encontrada, con el propósito de determinar que palabras son posibles de descartar sin intervenir demasiado los datos.



(a) Vocabulario total

(b) Vocabulario reducido

Figura 4.3: Distribución de palabras según rango de frecuencia de aparición

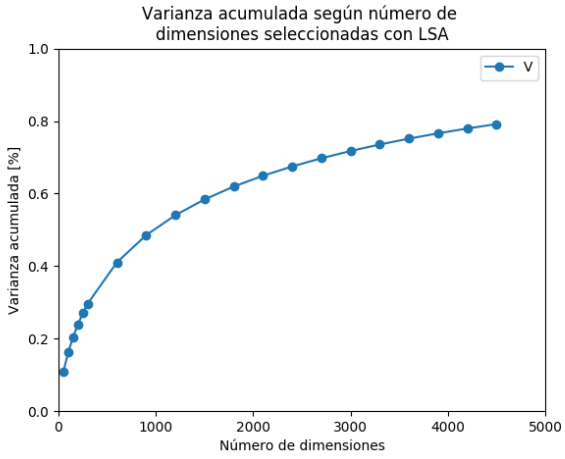
[1]Stop words español: `https://github.com/stopwords-iso/stopwords-es/blob/master/stopwords-es.txt`

La imagen de la Figura 4.3a, muestra la cantidad de palabras según frecuencias de aparición. En ella es posible observar que el rango de frecuencia de aparición 0 a 3 posee cerca de un 78 % del total del vocabulario. Producto de lo anterior, se opta por descartar todas aquellas palabras cuya frecuencia de aparición sea menor que 4, obteniendo de este modo un vocabulario reducido de 22.506 palabras, reflejado en la imagen de la Figura 4.3b. Por último, se descartan todas aquellas palabras cuya frecuencia de aparición en los documentos (document frequency) fuese mayor a la mitad del total de los documentos o menor a 4, es decir que dicha palabra no aparece en más de 3 documentos distintos. Realizado lo anterior, el tamaño final del vocabulario reducido es de 22.411 palabras.

4.2.2. Descriptores de texto

Para la obtención de los descriptores de texto es necesario entrenar un modelo de BoW con TF-IDF y disminuir la dimensión del vector utilizando LSA. Además, se debe calcular los vectores de oraciones utilizando un modelo pre-entrenado de word embedding, en este caso FastText.

Para el entrenamiento del modelo de Bow con TF-IDF se utiliza el modelo TfidfVectorizer de la librería scikitlearn, que nos permite calcular los vectores correspondientes para cada uno de los títulos de los productos del conjunto de entrenamiento. Una vez obtenidos estos vectores, es necesario realizar una reducción en el número de dimensiones. Para ello se realiza un análisis semántico latente (LSA) donde se busca extraer todas aquellas características del vector que permiten mantener una alta varianza acumulada. De este modo se genera el gráfico de la Figura 4.4, en el que se observa el porcentaje de varianza acumulada según el número de dimensiones escogidas para la reducción.



(a)

Dim	Varianza [%]	Dim	Varianza [%]
50	10,829	1800	61,938
100	16,259	2100	64,909
150	20,470	2400	67,477
200	23,996	2700	69,734
250	27,015	3000	71,739
300	29,668	3300	73,537
600	41,031	3600	75,155
900	48,531	3900	76,615
1200	54,033	4200	77,945
1500	58,390	4500	79,166

(b)

Figura 4.4: Varianza acumulada según numero de dimensiones seleccionadas

A continuación, es requerido vectorizar las oraciones utilizando un modelo pre-entrenado de FastText. Para ello, se implementaron 2 versiones: la primera, consta de calcular el IDF de cada una de las palabras del vocabulario, para luego realizar una suma ponderada de los vectores normalizados de cada palabra de la sentencia por el IDF de la palabra correspondiente.

La ecuación 4.1 representa el calculo mencionado previamente, donde $sentencia_i$ representa la i-esima palabra de la oración y la función *embedding* corresponde al calculo del vector según el modelo pre-entrenado de FastText. La segunda versión de vectorización corresponde al cálculo del promedio de los vectores normalizados de cada palabra de la sentencia, este cálculo se encuentra descrito por la ecuación 4.2.

$$embedding(sentencia) = \frac{\sum_{i=1}^{N} embedding(sentencia_i) * IDF(sentencia_i)}{\sum_{j=1}^{N} IDF(sentencia_j)} \tag{4.1}$$

$$embedding(sentencia) = \frac{\sum_{i=1}^{N} embedding(sentencia_i)}{N} \tag{4.2}$$

4.2.3. Descriptores visuales

Para la obtención de los descriptores visuales es utilizada la librería Keras, ya que ofrece una simple interacción con redes convolucionales. Además, tiene implementada la arquitectura de los modelo ResNet50 y VGG19 con los respectivos pesos del entrenamiento de dichas redes con el conjunto de datos de ImageNet. Para obtener el descriptor visual con la red ResNet50, es requerido cargar la imagen en memoria y re-dimensionar la imagen para que coincida con el tamaño de la entrada de la red. Posterior a ello, se debe interceptar la red en la salida del último bloque de convolución, justo antes de las capas clasificatorias, y realizar un global average pooling (GAP) que permite obtener el vector de 2048 dimensiones correspondiente al descriptor visual de dicha red. Por otra parte, para la obtención del descriptor visual de la red VGG19, el proceso anterior se mantiene, salvo que ahora se debe interceptar la red en la salida de primera capa clasificadora, obteniendo así el vector de 4096 dimensiones.

4.2.4. Relación texto-imagen

Para obtener la relación texto-imagen descrita en el modelo Word2VisualVec, se propone la utilización de un descriptor de texto compuesto por la concatenación de los descriptores obtenidos con los modelos BoW con TF-IDF y FastText, con el propósito de contener en un único vector, las características semánticas y sintácticas del texto. Dicho vector nace desde la idea propuesta por el modelo dual-encodding (ver sección 3.1), ya que este modelo busca obtener un descriptor de texto que contenga la mejor representación de este. También, se propone utilizar un descriptor de imagen obtenido mediante redes convolucionales. Este último será obtenido con el modelo mejor evaluado, según los resultados obtenidos en la búsqueda visual. Finalmente, se propone la implementación de una red neuronal multicapa simple de 3 capas, que realice una regresión lineal de un descriptor de texto a un descriptor visual.

La arquitectura de la red se compone por una capa de entrada, una capa oculta totalmente conectada y una capa de salida. La dimensión de la primera capa corresponde al tamaño del descriptor de texto unificado, la dimensión de la segunda corresponde al promedio del

número de neuronas de la capa de entrada y la capa de salida, y la dimensión de la tercera capa corresponde al tamaño del descriptor visual. Como el propósito de la red neuronal es transformar un vector en otro, la función de costo utilizada para evaluar el desempeño de la transformación es el error cuadrático medio (mean squared error) descrita en la ecuación 4.5.

$$MSE(\hat{Y}, Y) = \frac{\sum_{i=1}^{n} (\hat{Y}_i - Y_i)^2}{n} \tag{4.3}$$

$$MAE(\hat{Y}, Y) = \frac{\sum_{i=1}^{n} \hat{Y}_i - Y_i}{n} \tag{4.4}$$

$$loss(D; \theta) = \sum_{(X,Y) \in D} MSE(r(X; \theta), Y) \tag{4.5}$$

Dada la arquitectura de la red, se define como θ al conjunto de parámetros dentro de la red que se deben aprender, y como $r(X; \theta)$ al vector resultante de la salida de la MLP dados los parámetros θ, es decir, la transformación del descriptor de texto al descriptor visual según la MLP. Además, se define el conjunto D como los pares (X, Y) pertenecientes al conjunto de datos de entrenamiento, donde X representa el descriptor de una oración e Y al descriptor de la imagen asociada. Producto de lo anterior, se define la ecuación 4.6 como la ecuación a optimizar durante del entrenamiento de la MLP, es decir, encontrar el conjunto de parámetros θ tal que la función 4.5 minimiza su valor para el conjunto de entrenamiento D. Por otra parte, dentro de la investigación guía utilizan el algoritmo de optimización RMSprop para resolver la ecuación, ya que genera mejores resultados durante el entrenamiento.

$$\underset{\theta}{argmin} \;\; loss(D; \theta) \tag{4.6}$$

Finalmente, para el entrenamiento de la red neuronal se divide el conjunto de entrenamiento en 2 subconjuntos: uno de entrenamiento (80 %) y uno de validación (20 %), con el propósito de disminuir el overfitting de la red. Además, se configura el algoritmo de optimización para resolver la ecuación 4.6 como RMSprop, al igual en el la investigación guía.

4.3. Diseño de Experimentos y Evaluación

Para la realización de los experimentos y su evaluación, se opta por la implementación de una aplicación web que tenga por función realizar búsquedas dentro del conjunto de evaluación definido previamente. Esta aplicación agrupa ambos tipos de búsquedas, es decir, búsqueda visual y búsqueda visual-semántica, y permite evaluar la precisión de los resultados arrojados, almacenándolos para su posterior análisis. Esta aplicación permite definir qué tipo de búsqueda se desea realizar, ya sea visual o visual-semántica, según las versiones definidas en la sección 4.3.2

4.3.1. Búsqueda visual

Para la implementación del servicio de búsqueda visual se definen 5 pasos:

1. Recepción de imagen
2. Cálculo de descriptor visual
3. Cálculo de similitud de descriptores
4. Ranking de imágenes más similares
5. Entrega de resultados y evaluación

El primer paso consiste en corroborar que la imagen se encuentre en un formato válido para el buscador, y guardar la imagen dentro del sistema con un id único. El segundo paso consiste en obtener el descriptor de la imagen mediante la utilización de una red convolucional, siguiendo los pasos definidos en la sección 4.2.3. En el tercer paso se utiliza la similitud coseno como métrica de comparación entre descriptores. Dicha métrica evalúa la cercanía de los vectores según el ángulo de desviación de estos en el espacio que los contiene. En el cuarto paso se utiliza una cola de prioridad en la que se incorporan todas las medidas de similitud calculadas, con el propósito de encontrar aquellos productos más similares al provisto en la imagen. Finalmente, dichos productos son desplegados en pantalla al usuario, solicitando que evalúe la exactitud de estos, con el fin de calcular las métricas de evaluación: precision at rank, average precision y reciprocal rank, descritas en la sección 2.8

Este servicio de búsqueda cumple 3 propósitos, el primero es comparar la precisión de los resultados mediante la utilización de 2 descriptores obtenidos con redes convolucionales distintas (ResNet50 y VGG19), en segundo lugar, definir de acuerdo a los resultados, cuál de los descriptores será utilizado en los experimentos siguientes, y tercero, obtener una línea base de un sistema de recuperación de información que utilice imágenes.

4.3.2. Búsqueda visual-semántica

Para la implementación de este servicio de búsqueda, se definen los siguientes pasos:

1. Recepción de imagen y texto de consulta
2. Cálculo de descriptores visual y de texto
3. Cálculo de similitud visual y similitud de texto
4. Ranking de productos más similares
5. Entrega de resultados y evaluación del usuario

Los pasos 1, 4 y 5 son equivalentes a los descritos en la sección anterior, mientras que los pasos 2 y 3 dependen de cada una de las versiones definidas, ya que se utilizan diferentes descriptores de texto.

Versión 1: Vector BoW con TF-IDF

La primera versión implementada para la combinación de texto e imagen en la realización de búsquedas de productos, consiste en utilizar el descriptor de texto obtenido con el modelo de BoW con TF-IDF usando el vocabulario reducido y el descriptor visual extraído desde la red convolucional con mejor desempeño según el experimento anterior. Dados estos vectores, se calculan las similitudes entre la consulta de usuario y cada producto, como la suma de las similitudes coseno entre descriptores de textos con las similitudes coseno de los descriptores de imágenes.

Versión 2: Vector BoW con TF-IDF reducido con LSA

La segunda versión implementada consiste en cambiar el descriptor de texto por el obtenido tras realizar una reducción en el número de dimensiones del vector resultante del modelo BoW con TF-IDF, según el análisis semántico latente realizado previamente. Finalmente, la relevancia de cada producto se calcula de igual manera que para la versión anterior, es decir, sumando las similitudes coseno obtenidas tras la comparación del texto de consulta con los títulos de los productos y las similitudes coseno obtenidas de la comparación de la imagen de consulta con la imagen de cada producto.

Versión 3: Transformación de descriptor de texto a descriptor de imagen

Por último, la tercera versión implementada consiste en realizar búsquedas sólo en el espacio de las imágenes. Debido a lo anterior, es requerido realizar una transformación del vector del texto de consulta a un vector en el espacio de las imágenes. De este modo, es posible calcular la similitud del vector transformado con los vectores de las imágenes de los productos existentes.

Con el propósito de refinar la búsqueda, se calcula la similitud del vector de la imagen entregada con los vectores de las imágenes existentes de manera equivalente a la realizada en una búsqueda visual. Luego, se suman las similitudes obtenidas por ambos métodos, obteniendo los productos más similares según los valores calculados. Cabe mencionar que, la transformación de un vector de texto se realiza mediante una MLP entrenada para dicho propósito.

Los tres tipos de búsqueda diseñados para este experimento tienen como propósito comparar diferentes formas de combinar texto e imágenes para la recuperación de productos. Por otra parte, cada método utiliza diferentes descriptores de texto, mientras que utilizan el mismo tipo de descriptor visual. Este último es escogido según los resultados del experimento diseñado en la sección 4.3.1

4.3.3. Métricas de evaluación

Para realizar la evaluación se calculan las métricas Average Precision at rank (AP@r) para los valores 1, 5, 10, 15, 20 de rank, la métrica Mean Average Precision (MAP) y Mean Reciprocal Rank (MRR), sobre un conjunto de 95 consultas distintas en cada uno de los servicios implementados. Calculadas las métricas, se prosigue a contrastar los resultados utilizando gráficos de barra con intervalos de confianza que permitan observar el comportamiento de estas métricas en cada buscador. Cabe destacar que mientras más cercano a 1 sean los valores obtenidos, mejor es el desempeño del buscador implementado.

Los intervalos de confianza son calculados mediante la formula de la ecuación 4.7 que corresponde al calculo del intervalo de confianza sobre una muestra con distribución normal con un nivel de confianza del 95 %. En esta formula σ representa a la desviación estándar de la muestra y n a la cantidad total de elementos.

$$error = \frac{1{,}96 * \sigma}{\sqrt{n}} \tag{4.7}$$

En las imágenes de la Figura 4.5 se observa un conjunto de ejemplos de consultas a realizar a los servicios implementados.

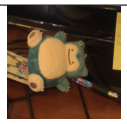
Imagen	Texto	Imagen	Texto
	Zapatillas nike		Juguete infantil
	Reloj negro		Peluche pokemon
	Billetera		Silla de auto infantil

Figura 4.5: Ejemplos de consultas

Capítulo 5

Experimentos y Análisis de Resultados

5.1. Buscador Visual

Una vez implementado el servicio de búsqueda visual, se prosigue con su evaluación realizando un total de 95 búsquedas de productos. Estas búsquedas se realizan utilizando los descriptores obtenidos con las redes ResNet50 y VGG19. Los resultados obtenidos tras la realización de las consultas se ven reflejados en la imagen de la Figura 5.1, donde se observa que ambos descriptores obtienen resultados similares. Sin embargo, los puntajes obtenidos por los descriptores calculados con la red ResNet50 son superiores para cada una de las métricas.

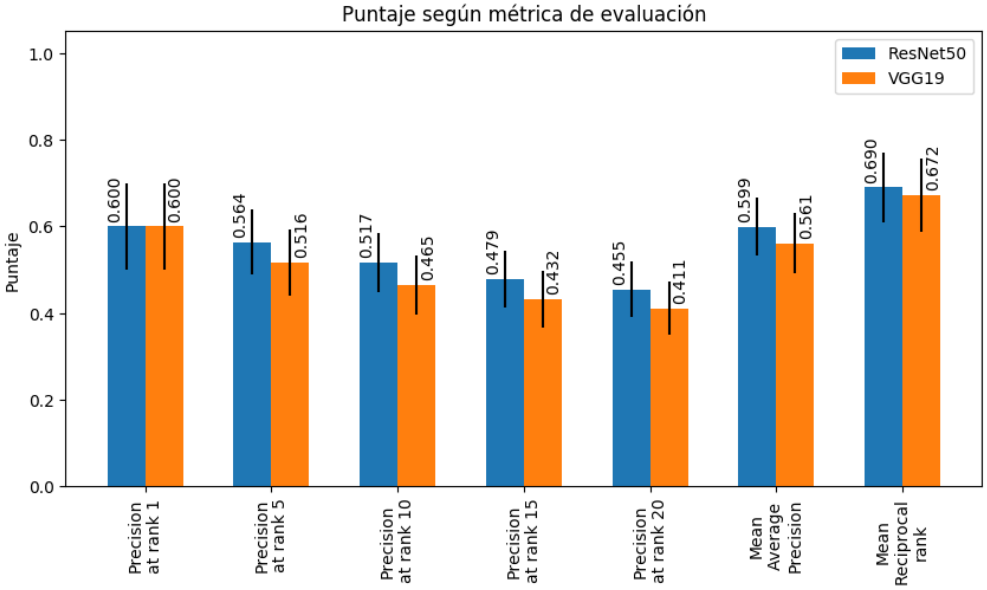


Figura 5.1: Resultados búsqueda visual

Del gráfico anterior, no es posible decir con certeza que los descriptores de la red convolucional ResNet50 son mejores que los descriptores de la red VGG19, debido a que los intervalos de confianza se cruzan en un amplio rango. Pese a ello, se opta por la utilización de los descriptores de la red ResNet50 para la implementación de las búsquedas visual-semánticas,

dado que obtiene una mayor precisión dentro de los experimentos realizados. Además, estos descriptores superan a la red VGG en distintas tareas de reconocimiento visual según los resultados expuestos en la investigación [8].

Durante la realización de los experimentos, se observa que ambos descriptores son capaces de extraer el objeto de interés desde la imagen de consulta. Sin embargo, existen grupos de objetos para los cuales los descriptores de la red ResNet50 son superiores a los descriptores de la red VGG19 y viceversa. En los ejemplos del anexo C se observan distintas consultas que muestran lo planteado, notando que en la Figura C.3 la red ResNet50 es capaz de extraer mayor detalles en el material de la correa del reloj, como también en la forma y detalles de la cajonera, mientras que en los ejemplos de la imagen C.2 la red VGG19 puede abstraer el objeto buscado a otros similares en el caso del auto de carreras y a localizar dentro de una imagen con varios objetos, el de interés. No obstante, existen casos en los que ambas redes no son capaces de encontrar resultados. En la imagen de la Figura C.1, se observan 2 casos poco favorables para ambas redes, donde el ruido en la imagen, para el caso del cinturón, provoca que ambas redes encuentren como productos más similares a aquellos que poseen una textura parecida al ruido de la imagen. De forma similar sucede con el caso del aire acondicionado, donde ambas redes buscan por forma y textura del enrejado en vez del objeto completo.

Respecto a los resultados obtenidos se puede mencionar que:

1. El AP@r de la red ResNet50 es mayor igual al AP@r de la red VGG19 para los valores de r 1, 5, 10, 15 y 20.
2. El MAP de la red ResNet50 es mayor al MAP de la red VGG19.
3. El MRR de la red ResNet50 es mayor al MRR de la red VGG19.

Según lo anterior, en conjunto con los resultados expuestos en el gráfico de la Figura 5.1, es posible afirmar que para las búsquedas realizadas con los descriptores obtenidos con la red RedNet50, la primera respuesta correcta se encuentra entre las posiciones 1 y 2, producto del valor del MRR. La probabilidad de que la primera respuesta sea correcta es de un 60 %, dado el valor del AP@1. Además, en promedio 3 de las primeras 5 respuestas son correctas, 5 de las primeras 10 respuestas son correctas, 7 de las primeras 15 respuestas son correctas y 9 de las primeras 20 respuestas son correctas, dados los valores AP@5, AP@10, AP@15 y AP@20, respectivamente.

5.2. Buscador Visual-Semántico

Para la implementación del buscador visual-semántico, se utilizan los descriptores de imagen obtenidos a través de la red convolucional ResNet50, ya que obtienen mejores resultados en los experimentos del buscador visual mostrados en la sección anterior.

Según lo expuesto en la sección 4.3.2, se proponen tres versiones para la obtención del descriptor de texto. La primera consiste en utilizar un vector BoW con TF-IDF, la segunda es reducir la dimensión del vector anterior mediante un análisis semántico latente, y la tercera es realizar una regresión lineal mediante una MLP que proyecte hacia el espacio de las imágenes,

el vector resultante de la concatenación del vector BoW con TF-IDF de dimensión reducida con el vector obtenido del modelo FastText.

Para la implementación de la segunda versión, se examinan los resultados obtenidos de la Figura 4.4, en la cual se muestra la varianza acumulada según el número de dimensiones escogidas luego de realizar un análisis semántico latente. Dado lo anterior, se selecciona un total de 4500 dimensiones para realizar la reducción del vector BoW con TF-IDF, puesto que acumula aproximadamente un 80 % de la varianza de los datos.

Para la implementación de la tercera versión, se utiliza el vector BoW con TF-IDF reducido y el vector obtenido tras calcular el sentence embedding al usar un modelo FastText pre-entrenado. Este último corresponde a un vector de 300 dimensiones que se obtiene mediante la formula 4.1. Dado lo anterior, el vector resultante para realizar la transformación posee 4800 dimensiones.

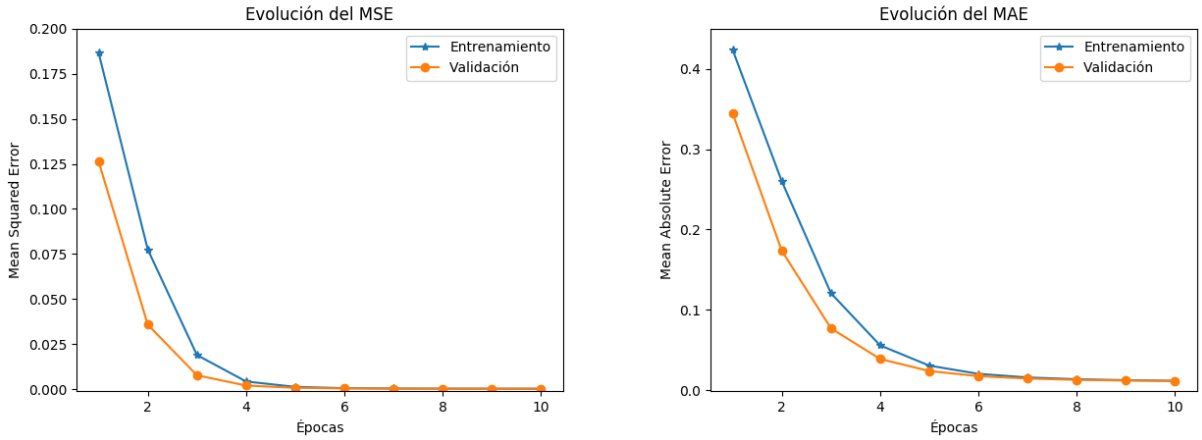
En resumen, los vectores de texto utilizados para la implementación de las versiones 1, 2 y 3 del buscador visual-semántico poseen las siguientes dimensiones:

Versión	Dimensiones
V1	22.411
V2	4.500
V3	4.800

Tabla 5.1: Dimensiones de los vectores de textos utilizados.

Una vez definidos los vectores a utilizar, se construye la MLP según las características descritas en la sección 4.2.4, quedando una red de 3 capas, donde los tamaños de la capa de entrada, capa oculta y capa de salida son 4800, 3424 y 2048, respectivamente. Una vez definidos los vectores, se prosigue con el entrenamiento según el procedimiento explicado en la sección 4.2.4, donde primero se separa el conjunto de datos de entrenamiento en dos subconjuntos, uno de entrenamiento y uno de validación. Hecho lo anterior, se configura la red para minimizar el valor de la métrica MSE, obteniendo como resultado las curvas expuestas en la Figura 5.2. Estas curvas muestran la evolución de la métricas MSE y MAE para los conjuntos definidos durante un periodo de entrenamiento de 10 épocas.

En los gráficos de la Figura 5.2, se observa una disminución progresiva de las métricas MSE y MAE, llegando a valores casi nulos en las épocas 5 y 7 respectivamente. Lo anterior, indica que la red neuronal va aprendiendo correctamente la transformación de los vectores a medida que pasan las épocas. Cabe mencionar, que una época es el término utilizado para referirse al entrenamiento de una red neuronal con el total de datos de entrenamiento, es decir, entrenar por 10 épocas corresponde a entrenar una red 10 veces con los mismos datos.



(a) MSE por época de entrenamiento

(b) MAE por época de entrenamiento

Figura 5.2: Evolución de las métricas MSE y MAE por época, dentro de los conjuntos de entrenamiento y validación

Una vez entrenada la red y calculados los vectores descriptores de texto, se prosigue a la evaluación del servicio de búsqueda visual-semántico, realizando un total de 95 consultas a cada una de las versiones implementadas. Es importante mencionar, que para que los resultados sean comparables, las búsquedas son realizadas sobre el mismo conjunto de datos utilizado para la evaluación del servicio de búsqueda visual, es decir, se usan las mismas imágenes de consulta y se busca sobre los mismos productos. En la imagen de la Figura 5.3, se observan las métricas de comparación calculadas para evaluar la precisión de los resultados obtenidos con las distintas versiones del buscador visual-semántico.

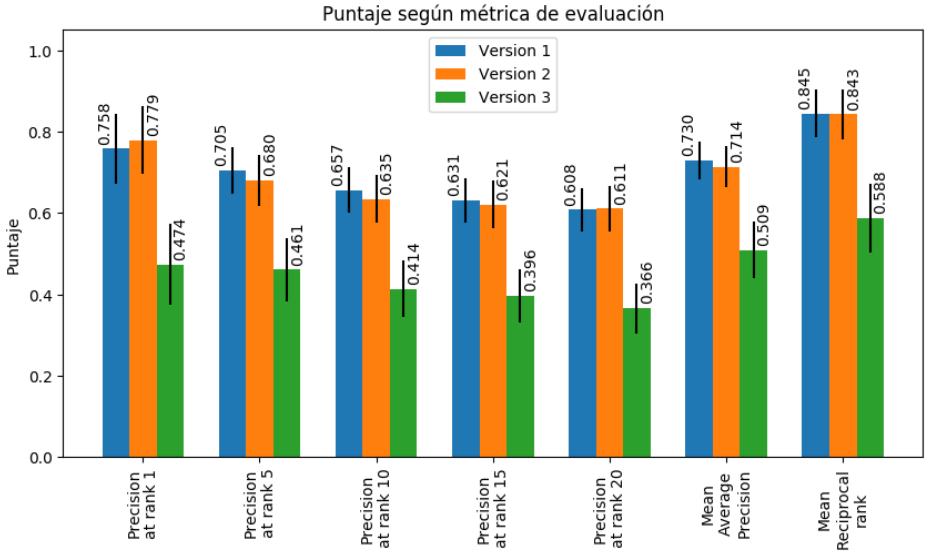


Figura 5.3: Resultados búsqueda semántica

Dentro del gráfico expuesto, es posible observar que la primera versión implementada obtiene resultados destacables respecto a su precisión en cada una de las métricas, superando ampliamente los resultados de la versión 3. Por otra parte, los resultados obtenidos por la

versión 2 son similares a los obtenidos por la versión 1 observándose una diferencia máxima de 0,025 puntos entre los resultados de ambas versiones.

Durante la realización de los experimentos, se observa que la versión 3 obtiene buenos resultados para ciertos tipos de consulta, algunos ejemplo de casos se muestran en las imágenes de la Figura C.4. En ellos es posible notar que la similitud obtenida es bastante alta, llegando a ser para algunos casos 0,552 lo que muestra que la transformación realizada por la MLP genera un vector en el espacio de las imágenes que se asemeja considerablemente al producto descrito en el texto entregado. Ahora bien, la razón por la cual los resultados son tan bajos para el resto de las consultas, se debe a los datos de entrenamiento, pudiéndose tener 2 casos, el primero es que la red se entrenó para realizar la transformación de un tipo de productos distintos a los utilizados para realizar búsquedas o los datos de entrenamiento no son útiles para entrenar la relación entre texto e imagen. Sin embargo, debido al tipo de división utilizada para separar los datos en entrenamiento y evaluación, se descarta el primer caso mencionado, dado que se procuró mantener una proporción similar de la cantidad de productos de cada tienda en ambos subconjuntos.

Según los datos del gráfico de la Figura 5.3, se puede afirmar que la primera respuesta correcta se encuentra en la primera posición con una probabilidad del 76 %, producto del valor del MRR y AP@1. Además, en promedio 4 de las primeras 5 respuestas son correctas, 6 de las primeras 10 respuestas son correctas, 10 de las primeras 15 respuestas son correctas y 12 de las primeras 20 respuestas son correctas, dados los valores AP@5, AP@10, AP@15 y AP@20, respectivamente.

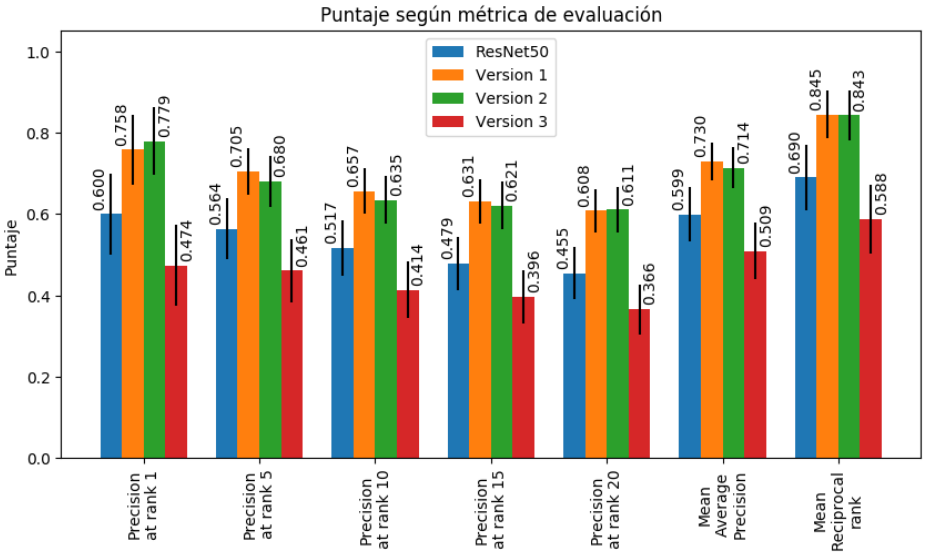


Figura 5.4: Comparación entre búsqueda visual y búsqueda semántica

Finalmente, en el gráfico de la Figura 5.4 se observa la comparación entre los resultados obtenidos entre la búsqueda visual-semántica y la búsqueda visual, notando que la búsqueda semántica supera en todas las métricas a la búsqueda visual. Por lo que se afirma que la incorporación de texto a la consulta, mejora la precisión de los resultados obtenidos en la búsqueda de productos de tiendas en línea.

Capítulo 6

Conclusiones y Trabajo Futuro

Sobre el trabajo realizado, se puede concluir que la incorporación de texto a la consulta de una búsqueda permite mejorar significativamente la precisión de los resultados entregados por un sistema de recuperación de información. Según los resultados obtenidos el modelo de vectorización de sentencias BoW con TF-IDF aumenta aproximadamente en un 16 % la precisión de los resultados obtenidos en una búsqueda visual.

La aplicación de un análisis semántico latente en los vectores descriptores de texto, permite disminuir la dimensión de estos, manteniendo las características propias del vector original. Lo anterior queda demostrado según los resultados expuesto en la sección 5.2, donde se observa que el vector de texto con dimensión reducida obtiene valores de precisión muy cercanos a los obtenidos con el vector BoW con TF-IDF. Sin embargo, no se recomienda su uso para la implementación de sistemas de recuperación de información, puesto que aumenta drásticamente el espacio en memoria utilizado por dichos vectores, pero si se recomienda su uso para la utilización de estos en el entrenamiento de una red neuronal multicapa, como fue implementado en la versión 3 del buscador visual-semántico, dado que disminuye el tamaño de la entrada a valores manejables y atenúa el overfitting del entrenamiento.

Respecto a los descriptores visuales obtenidos con redes convolucionales, se concluye que son útiles para la implementación de un sistema de recuperación de información, puesto que son capaces de extraer características profundas de los objetos presentes en una imagen. Sin embargo, estos descriptores son generales debido al entrenamiento al cual es sometida la red (conjunto de imágenes de ImageNet) es por ello que para algunos objetos los descriptores no son capaces de distinguir ni describir su contenido. Dicho lo anterior, se propone ajustar los pesos de las redes convolucionales, realizando un entrenamiento de las últimas capas, utilizando imágenes de productos de tiendas, con el propósito de especializar la red en la tarea de describir el contenido visual de productos.

Dentro de los problemas encontrados durante la experimentación se encuentran, primero, los malos resultados obtenidos en la versión 3 del buscador semántico, segundo, la presencia de productos cuyos títulos se encuentran en otro idioma y, tercero, la falta de datos de evaluación para los cuales se conozca las respuestas esperadas, es decir, un ground truth. Es por ello que se propone rehacer los experimentos utilizando un conjunto de datos más estructurado, en el

cual los títulos de los productos sean claros, concisos y que consideren sólo las características observables de la imagen del producto. Además, se recomienda incorporar una descripción detallada del producto según la imagen provista, con el propósito de generar una mejor relación entre descriptor de texto y descriptor de imagen. Por otra parte, se recomienda utilizar productos que se encuentren en un mismo idioma, ya que el uso de traductores incorpora ruido dentro del conjunto de datos de entrenamiento, pudiendo generar fallas y errores durante el aprendizaje de los modelos. Finalmente, se recomienda la definición de un ground truth para agilizar y mejorar el proceso de evaluación, pudiendo de este modo realizar más experimentos que permitan extraer información relevante sobre el proceso de combinación de textos e imágenes para la realización de búsquedas de productos de tiendas en línea.

Finalmente, se concluye que el servicio de búsqueda visual-semántico implementado con descriptores de texto obtenidos con el modelo BoW con TF-IDF en conjunto con descriptores visuales obtenidos con la red neuronal ResNet50, define la mejor forma de combinar texto e imagen para la realización de búsquedas productos, según los resultados obtenidos en la sección anterior. Pese a los malos resultados de la versión 3, esta no se descarta del buscador visual-semántico puesto que debe ser evaluado utilizando un conjunto de datos refinado según las características mencionadas anteriormente. Es por ello que queda como trabajo futuro construir o utilizar un conjunto de datos representativo en el que los textos e imágenes asociados posean una relación significativa.

Bibliografía

[1] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *CoRR*, 2016.

[2] Y.Jia P.Sermanet S.Reed D.Anguelov D.Erhan V. Vanhoucke C.Szegedy, W.Liu and A. Rabinovich. Going deeper with convolutions. *CVPR*, 2015.

[3] Jianfeng Dong, Xirong Li, and Cees G. M. Snoek. Word2visualvec: Cross-media retrieval by visual feature prediction. *CoRR*, abs/1604.06838, 2016.

[4] Jianfeng Dong, Xirong Li, Chaoxi Xu, Shouling Ji, Yuan He, Gang Yang, and Xun Wang. Dual encoding for zero-example video retrieval. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[5] Y. Tamaazousti I. Chami and H. Le Borgne. Amecon: Abstract meta-concept features for text-illustration. *ICMR*, 2017.

[6] S. Fidler I. Vendrov, R. Kiros and R. Urtasun. Order-embeddings of images and language. *ICLR*, 2016.

[7] Xirong Li Jianfeng Dong and Cees G. M. Snoek. Predicting visual features from text for image and video caption retrieval. *IEEE*, 2018.

[8] S. Ren K. He, X. Zhang and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016.

[9] P. Young M. Hodosh and J. Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *JAIR*, 47(1):853–899, 2013.

[10] M. Hodosh P. Young, A. Lai and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*, 2:67–78, 2014.

[11] Cees G. M. Snoek Pascal Mettes, Dennis C. Koelma. The imagenet shuffle: Reorganized pre-training for video event detection. *ICMR*, 2016.

[12] G. Corrado T. Mikolov, K. Chen and J. Dean. Efficient estimation of word representations in vector space. *ICLR*, 2013.

[13] J. Donahue S. Karayev J. Long R. Girshick S. Guadarrama Y. Jia, E. Shelhamer and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *MM*, 2014.

Apéndice A

Anexo A

Dentro de las imágenes adjuntas en el presente anexo, se encuentran las distintas arquitecturas, de redes convolucionales descritas en el capitulo 2, junto con la arquitectura deneral del modelo dual encodding.

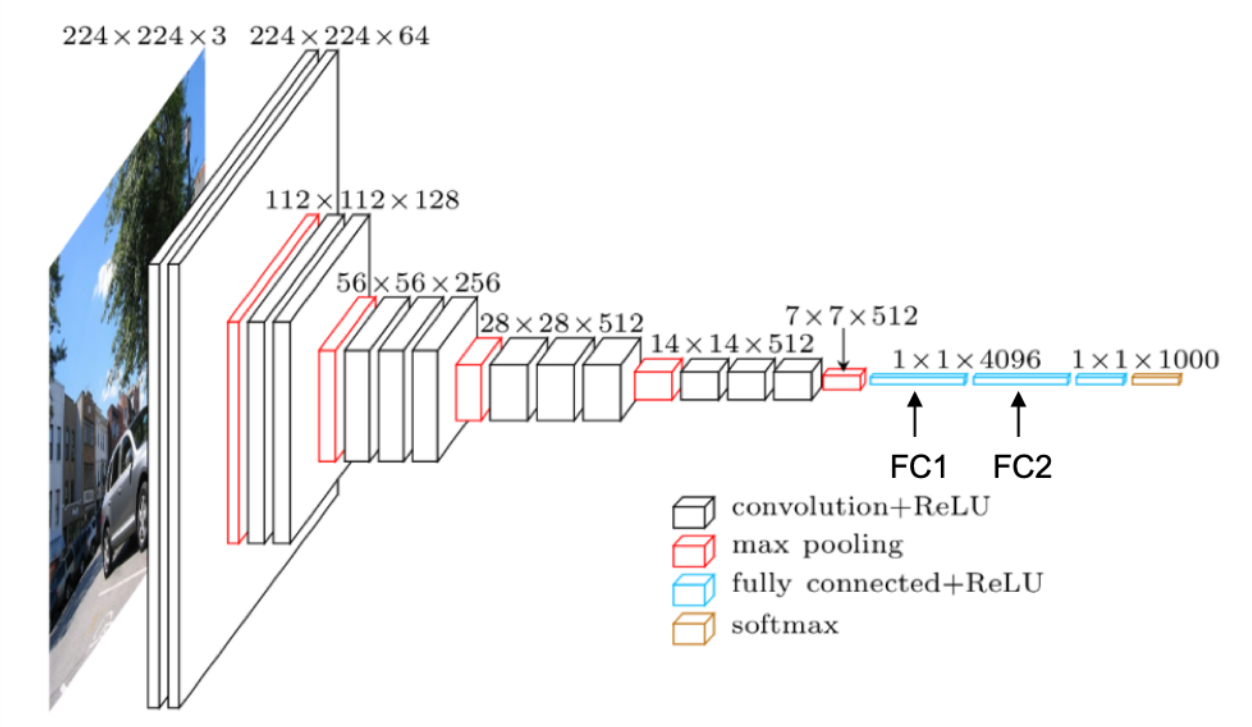


Figura A.1: Arquitectura de la red VGG

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figura A.2: Arquitecturas de redes convolucionales ResNet

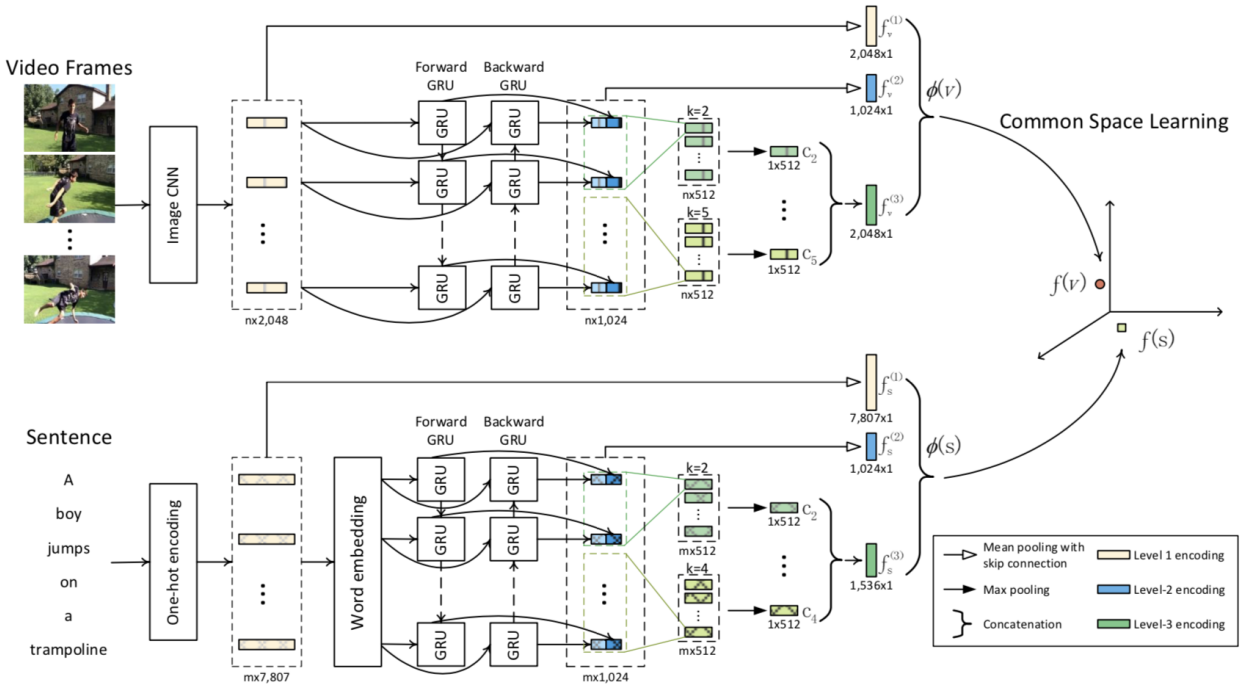


Figura A.3: Arquitectura general del modelo dual encodding

Apéndice B

Anexo B

En el presente anexo se exponen las tablas resumen de los resultados obtenidos tras la evaluación de los servicios de búsqueda, en ellas se exhiben las diferentes métricas de evaluación con sus respectivos promedio, desviación estándar y error, mostrando además los valores mínimo y máximo según el intervalo de confianza calculado.

B.1. Búsqueda Visual

Búsqueda visual: ResNet50

Metric	Precision at rank 1	Precision at rank 5	Precision at rank 10	Precision at rank 15	Precision at rank 20	Average Precision	Reciprocal rank
Mean	0,600	0,564	0,517	0,479	0,455	0,599	0,690
Standard deviation	0,490	0,379	0,343	0,324	0,317	0,334	0,400
Error	0,099	0,076	0,069	0,065	0,064	0,067	0,080
Min value	0,501	0,488	0,448	0,414	0,391	0,532	0,609
Max value	0,699	0,641	0,586	0,544	0,519	0,666	0,770

Tabla B.1: Resultados de búsquedas visuales realizadas con la red convolucional ResNet50. Total de consultas realizadas: 95

Búsqueda visual: VGG19

Metric	Precision at rank 1	Precision at rank 5	Precision at rank 10	Precision at rank 15	Precision at rank 20	Average Precision	Reciprocal rank
Mean	0,600	0,516	0,465	0,432	0,411	0,561	0,672
Standard deviation	0,490	0,383	0,341	0,325	0,310	0,350	0,417
Error	0,099	0,077	0,069	0,065	0,062	0,070	0,084
Min value	0,501	0,439	0,397	0,366	0,349	0,491	0,588
Max value	0,699	0,593	0,534	0,497	0,473	0,632	0,756

Tabla B.2: Resultados de búsquedas visuales realizadas con la red convolucional VGG19. Total de consultas realizadas: 95

B.2. Búsqueda Visual-Semántica

Búsqueda semántica: Versión 1

Metric	Precision at rank 1	Precision at rank 5	Precision at rank 10	Precision at rank 15	Precision at rank 20	Average Precision	Reciprocal rank
Mean	0,758	0,705	0,657	0,631	0,608	0,730	0,845
Standard deviation	0,428	0,283	0,275	0,269	0,268	0,234	0,286
Error	0,086	0,057	0,055	0,054	0,054	0,047	0,058
Min value	0,672	0,648	0,602	0,577	0,554	0,683	0,787
Max value	0,844	0,762	0,712	0,685	0,662	0,777	0,902

Tabla B.3: Resultados de búsquedas semánticas realizadas según la versión 1 definida en la sección 4.3.2. Total de consultas realizadas: 95

Búsqueda semántica: Versión 2

Metric	Precision at rank 1	Precision at rank 5	Precision at rank 10	Precision at rank 15	Precision at rank 20	Average Precision	Reciprocal rank
Mean	0,779	0,680	0,635	0,621	0,611	0,714	0,843
Standard deviation	0,415	0,309	0,290	0,288	0,279	0,251	0,305
Error	0,083	0,062	0,058	0,058	0,056	0,051	0,061
Min value	0,696	0,618	0,576	0,563	0,555	0,664	0,781
Max value	0,862	0,742	0,693	0,679	0,667	0,765	0,904

Tabla B.4: Resultados de búsquedas semánticas realizadas según la versión 2 definida en la sección 4.3.2. Total de consultas realizadas: 95

Búsqueda semántica: Versión 3

Metric	Precision at rank 1	Precision at rank 5	Precision at rank 10	Precision at rank 15	Precision at rank 20	Average Precision	Reciprocal rank
Mean	0,474	0,461	0,414	0,396	0,366	0,509	0,588
Standard deviation	0,499	0,382	0,341	0,326	0,303	0,348	0,417
Error	0,100	0,077	0,069	0,065	0,061	0,070	0,084
Min value	0,373	0,384	0,345	0,330	0,305	0,439	0,504
Max value	0,574	0,538	0,482	0,461	0,427	0,579	0,672

Tabla B.5: Resultados de búsquedas semánticas realizadas según la versión 3 definida en la sección 4.3.2. Total de consultas realizadas: 95

Apéndice C

Anexo C

En el presente anexo se exponen distintos resultados obtenidos tras la realización de consultas a los buscadores visual y visual-semántico. En ellos se muestran las primeras 5 respuestas entregadas por el buscador respectivo.

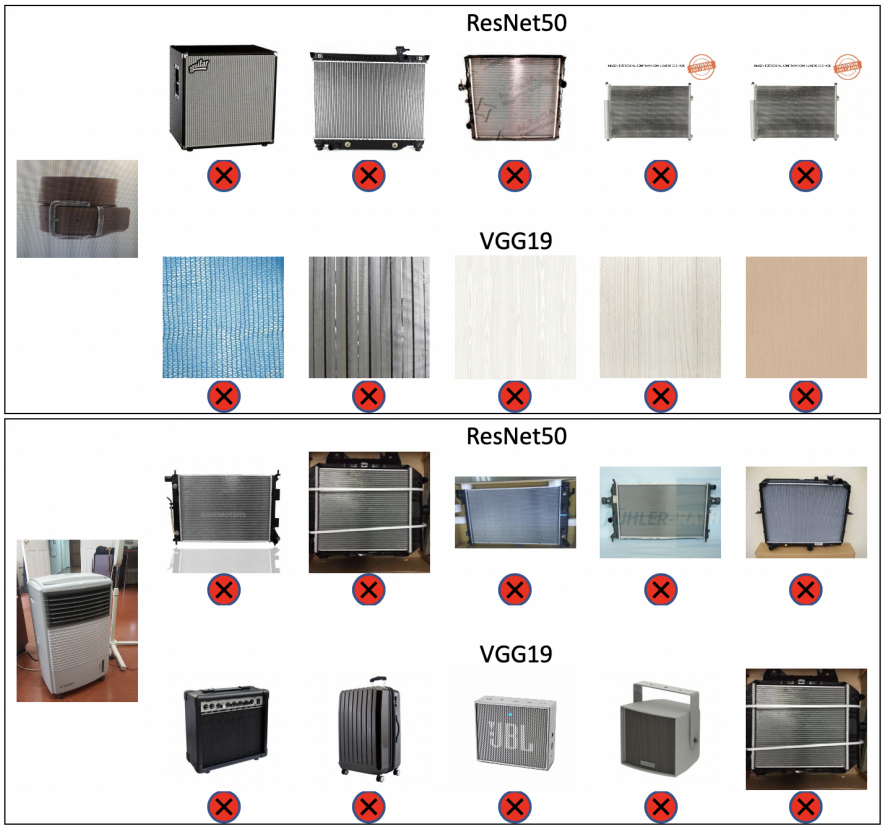


Figura C.1: Ejemplos de resultados malos en una búsqueda visual

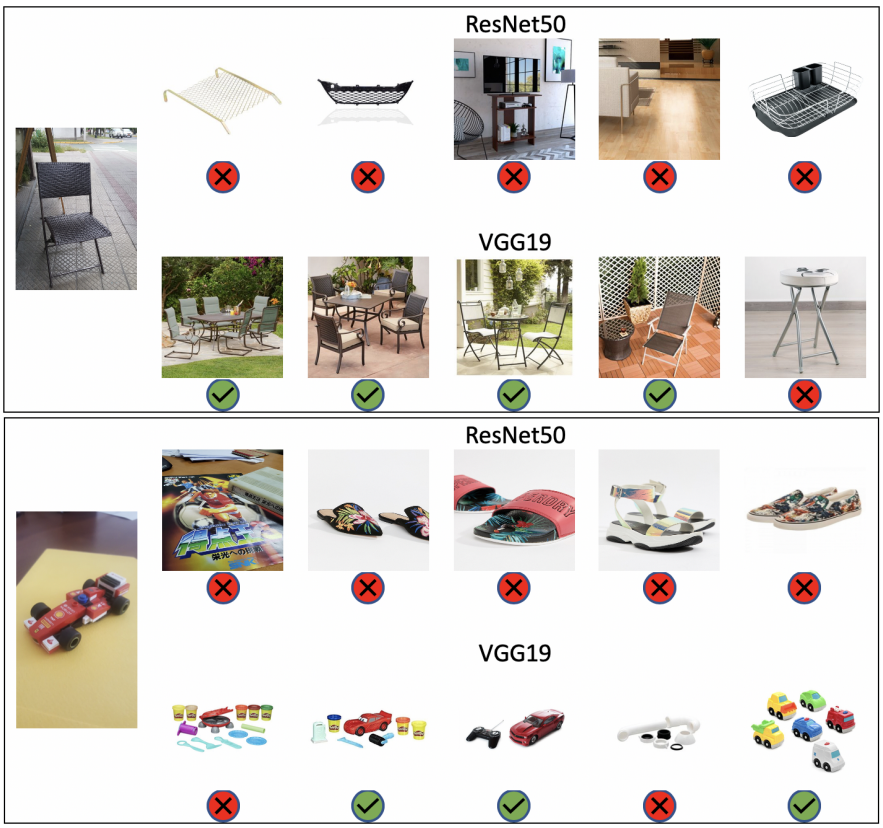


Figura C.2: Ejemplos de resultados favorables a VGG19 en una búsqueda visual

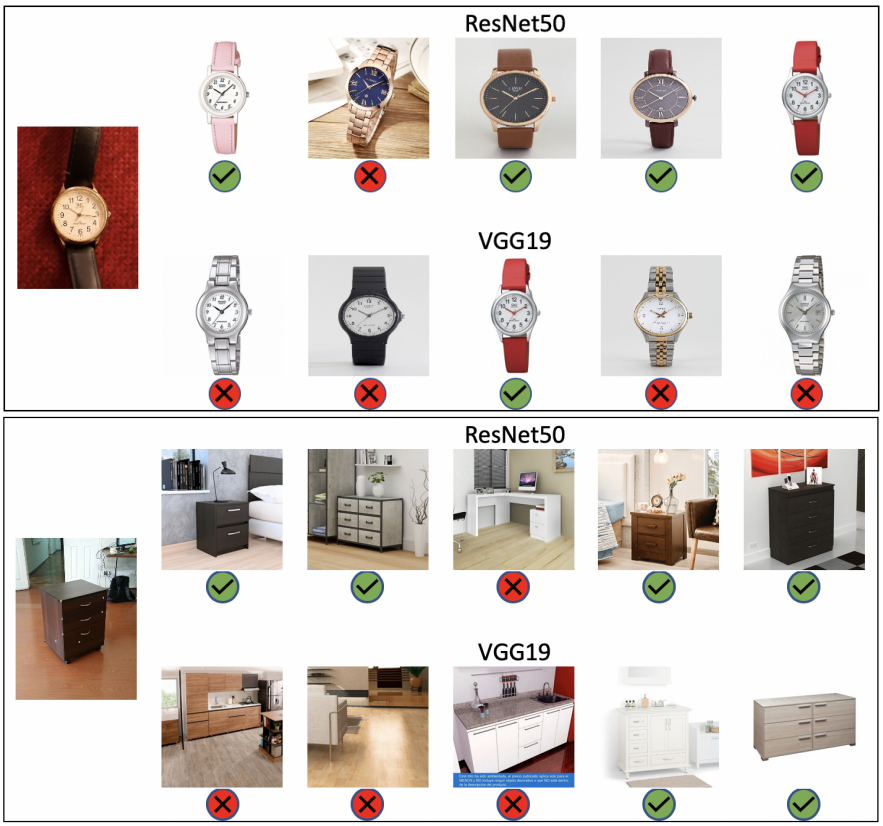


Figura C.3: Ejemplos de resultados favorables a ResNet50 en una búsqueda visual

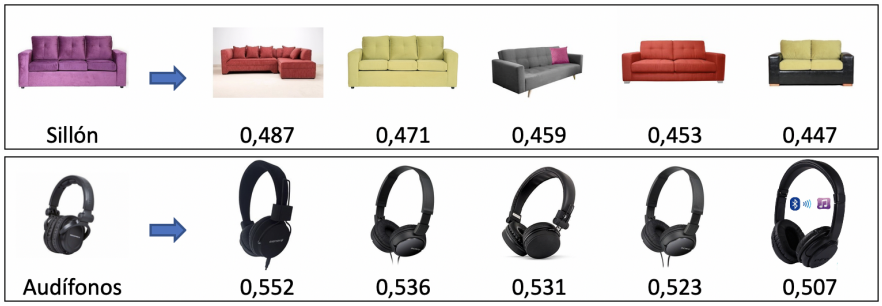


Figura C.4: Ejemplos de resultados favorables para la versión 3 del buscador semántico