

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO - CHILE



“EVALUACIÓN DE MÉTODOS DE RECUPERACIÓN DE IMÁGENES BASADAS EN CONTENIDO”

JEAN FRANCO ARAVENA ORELLANA

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Ricardo Ñanculef
Profesor Correferente: Carlos Valle

Agosto - 2023

DEDICATORIA

A mi familia, amigos, compañeros y profesores por haber sido parte de este largo proceso.

AGRADECIMIENTOS

Quiero dedicarle estas palabras a mi familia para agradecerles por todo su apoyo y cariño, por siempre incentivarme a seguir adelante y estar presentes en todo momento. Ha sido un largo camino, sobre todo la parte final.

También quiero agradecer a mi pareja Virginia Aretuo por haberme motivado a terminar este proceso y apoyarme siempre, a mis dos grandes amigos Jorge Curinao y Renzo Forteza, por haber estado ahí motivándome y ayudando a hacer el proceso un poco más ameno, mirando siempre el horizonte.

Finalmente, agradecer a los profesores Ricardo Ñanculef y Carlos Valle por haber estado ahí a pesar de alargar mucho este proceso. Además, agradecer a todos los profesores que tuve en la universidad, se aprendió mucho y ese aprendizaje últimamente me ha sido de gran ayuda.

RESUMEN

Resumen— La recuperación de imágenes basadas en contenido (CBIR) apunta a obtener imágenes similares a una imagen de entrada. El contenido hace referencia a colores, formas, texturas u otra información derivada de una imagen. La importancia de CBIR radica en que permite la búsqueda eficiente y eficaz de imágenes dentro de una base de datos aprovechando el contenido de la imagen. Si bien existe la búsqueda por imágenes utilizando etiquetas o metadatos, no siempre se cuenta con dicha información, incluso puede ser incorrecta o difícil de obtener. Existe una amplia gama de aplicaciones relacionadas con CBIR, una de ellas es el apoyo a médicos para la detección de enfermedades. Proximity es uno de los tantos métodos desarrollados en el área de CBIR, se basa en la concatenación de una red neuronal de segmentación y otra de clasificación. Lamentablemente, Proximity y muchos otros sistemas CBIR asumen la disponibilidad de un gran número de imágenes etiquetadas por médicos expertos. Este supuesto es difícil de satisfacer en la práctica y fuerza a los desarrolladores a entrenar el sistema con conjuntos de datos internacionales cuyas características pueden diferir de los datos nacionales para los que el sistema está siendo construido. Self-supervised Learning es un concepto que ha ganado popularidad últimamente al ser métodos que no requieren de imágenes etiquetadas, es por ello que se investiga su uso para romper las limitaciones mencionadas anteriormente, obteniendo mejoras tanto en precisión como en tiempo y tamaño de embeddings, sin la necesidad de utilizar etiquetas.

Palabras Clave— Redes Neuronales, Auto-supervisado, Recuperación de imágenes.

ABSTRACT

Abstract— Content-Based Image Retrieval (CBIR) aims to obtain images that are similar to an input image. Content refers to colors, shapes, textures, or other information derived from an image. The importance of CBIR lies in its ability to efficiently and effectively search for images within a database by leveraging the content of the image. While image search using tags or metadata exists, such information is not always available, and it can be incorrect or difficult to obtain. There is a wide range of applications related to CBIR, one of which is supporting doctors in disease detection. Proximity is one of the many methods developed in the field of CBIR, based on the concatenation of a segmentation neural network and a classification neural network. Unfortunately, Proximity and many other CBIR systems assume the availability of a large number of images labeled by expert doctors. This assumption is difficult to meet in practice and forces developers to train the system using international datasets, which may differ from the national data for which the system is being built. Self-supervised learning is a concept that has gained popularity recently as it doesn't require labeled images. Therefore, research is being conducted on its use to overcome the aforementioned limitations, achieving improvements in accuracy, time and size of embeddings, without the need for labels.

Keywords—Neural Network, Self-supervised, Image Retrieval

GLOSARIO

BERT: Bidirectional Encoder Representations from Transformers.

BYOL: Bootstrap Your Own Latent.

CNN: Convolutional Neural Network (Redes Convolucionales).

COVID-19: Coronavirus disease (Enfermedad respiratoria muy contagiosa causada por el virus SARS-CoV-2).

EMA: Exponential Moving Average.

FONDEF: Fondo de Fomento al Desarrollo Científico y Tecnológico.

IA: Inteligencia Artificial.

MLM: Masked Language Model.

NLP: Natural Language Processing (Procesamiento del lenguaje natural).

NN: Neural Network (Redes Neuronales).

NSP: Next Sentence Prediction.

ReLU: Rectified linear unit.

SIFT: Scale-invariant Feature Transform.

SSL: Self Supervised Learning.

ÍNDICE DE CONTENIDOS

RESUMEN	IV
ABSTRACT	V
GLOSARIO	VI
ÍNDICE DE FIGURAS	IX
ÍNDICE DE TABLAS	X
INTRODUCCIÓN	1
CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA	3
1.1 Contexto	3
1.2 Objetivos	5
1.2.1 Objetivo General	5
1.2.2 Objetivos Específicos	5
CAPÍTULO 2: MARCO CONCEPTUAL	6
2.1 Redes Neuronales	6
2.2 Funciones de Activación	8
2.3 Redes Feedforward	9
2.4 Redes Convolucionales	10
2.4.1 Arquitecturas Clásicas De Redes Convolucionales	13
2.5 Tipos de entrenamiento en una red neuronal	21
2.5.1 Self-supervised Learning	24
CAPÍTULO 3: PROPUESTA DE SOLUCIÓN	32
3.1 Descripción General	32
3.2 Proximity	32
3.3 Métodos Self-Supervised	37
3.3.1 SimCLR	37
3.3.2 BYOL	40
3.3.3 SiamSiam	43
3.3.4 Barlow Twins	45
CAPÍTULO 4: EXPERIMENTOS	48
4.1 DATASETS	48
4.2 MÉTRICAS	52
4.3 RESULTADOS	54
4.3.1 Proximity	55
4.3.2 Métodos Self-Supervised	66

CAPÍTULO 5: CONCLUSIONES	73
5.1 Conclusiones generales y contribuciones	73
5.2 Alcance y limitaciones	74
5.3 Trabajo futuro	74
ANEXOS	75
REFERENCIAS BIBLIOGRÁFICAS	99

ÍNDICE DE FIGURAS

1	Principales problemas con el etiquetado manual.	3
2	Ejemplo de un sistema CBIR.	4
3	Ejemplo de una red neuronal.	6
4	Modelo de neurona MP.	7
5	Función sigmoide.	8
6	Función tangente hiperbólica.	8
7	Función ReLU.	9
8	Ejemplo de una red Feedforward.	9
9	Arquitectura típica de una red convolucional.	11
10	Ejemplo de una convolución discreta.	11
11	Ejemplo de average pooling.	12
12	Ejemplo de max pooling.	12
13	Timeline de algunas de las redes convolucionales más importantes y populares de su época.	13
14	Arquitectura de LeNet-5.	14
15	Arquitectura de Alexnet	15
16	Arquitectura de GoogLeNet.	16
17	Modulo inception.	17
18	Arquitectura de VGG-11 hasta VGG-19.	17
19	Bloque residual (Skip connection).	18
20	Arquitectura de ResNet-18 hasta ResNet-154.	18
21	Pointwise Convolution seguida de Depthwise Convolution.	19
22	Arquitectura de Xception.	19
23	Bloque convolucional típico en ResNeXt.	20
24	Arquitectura de ResNet vs ResNeXt.	21
25	Aprendizaje Supervisado.	22
26	Aprendizaje No Supervisado.	22
27	Aprendizaje Semi-Supervisado.	23
28	Caso de uso del aprendizaje semi-supervisado.	23
29	Aprendizaje Reforzado.	24
30	Representación útil Self-supervised.	25
31	Pre-training y fine-tuning en BERT.	26
32	Arquitectura del autoencoder.	27
33	Color Jitter.	28
34	Random Grayscale.	28
35	Random Horizontal Flip.	29
36	Gaussian Blur.	29
37	Random Resized Crop.	29
38	Encoder en SSL.	30
39	Proceso completo de SSL en imágenes.	30
40	Xception en Proximity.	33
41	CE-Net en Proximity.	34

42	With concatenation en Proximity.	35
43	Without concatenation en Proximity.	36
44	Obtención de pares positivos en SimCLR.	37
45	Base encoder en SimCLR.	38
46	Projection head en SimCLR.	38
47	Arquitectura de SimCLR.	39
48	Obtención de la loss en SimCLR.	40
49	Arquitectura de BYOL.	41
50	Arquitectura de BYOL con énfasis en la estructura.	42
51	Arquitectura de SiamSiam.	44
52	Arquitectura de Barlow Twins	46
53	Proporción de géneros y cantidad de pacientes (SARS CoV2-CT scan).	48
54	Ejemplos de imágenes con y sin COVID-19 (SARS CoV2-CT scan).	48
55	Ejemplos de imágenes con y sin COVID-19 (COVID-19 CT).	49
56	Proceso de obtención de imágenes CT (COVID-CT Dataset).	50
57	Ejemplos de imágenes con y sin COVID-19 (COVID-CT Dataset).	50
58	Distribución de la edad para pacientes con COVID-19 (COVID-CT Dataset).	51
59	Proporción de géneros para pacientes con COVID-19 (COVID-CT Dataset).	51
60	Ejemplos de imágenes con y sin COVID-19 (Query Dataset).	52
61	Resumen de los dataset que contienen imágenes médicas.	52
62	Ejemplo para calculo de la precisión en un sistema CBIR.	53
63	Proximity: Median in seconds to retrieve k images across the whole query dataset.	58

ÍNDICE DE TABLAS

1	Proximity: mAP@k all images (mean between five different seeds).	55
2	Proximity: mAP@k all images (stdev between five different seeds).	55
3	Proximity: mAP@k images with covid (mean between five different seeds).	56
4	Proximity: mAP@k images with covid (stdev between five different seeds).	56
5	Proximity: mAP@k images without covid (mean between five different seeds).	57
6	Proximity: mAP@k images without covid (stdev between five different seeds).	57
7	Proximity: Median in seconds to retrieve k images across the whole query dataset.	58
8	Proximity: Embeddings dimension and size of every method.	59
9	Proximity Experiment: mAP@k all images (mean between five different seeds).	60
10	Proximity Experiment: mAP@k all images (stdev between five different seeds).	61
11	Proximity Experiment: mAP@k images with covid (mean between five different seeds).	62
12	Proximity Experiment: mAP@k images with covid (stdev between five different seeds).	63
13	Proximity Experiment: mAP@k images without covid (mean between five different seeds).	64

14	Proximity Experiment: mAP@k images without covid (stdev between five different seeds).	65
15	All methods: mAP@k all images (mean between five different seeds).	66
16	All methods: mAP@k all images (stdev between five different seeds).	67
17	All methods: mAP@k images with covid (mean between five different seeds).	68
18	All methods: mAP@k images with covid (stdev between five different seeds).	68
19	All models: mAP@k images without covid (mean between five different seeds).	69
20	All models: mAP@k images without covid (stdev between five different seeds).	70
21	All methods: Median in seconds to retrieve k images across the whole query dataset.	71
22	All methods: Embeddings dimension and size of every method.	72
23	Proximity Experiment: mAP@k all images seed 13.	75
24	Proximity Experiment: mAP@k all images seed 134.	76
25	Proximity Experiment: mAP@k all images seed 2023.	77
26	Proximity Experiment: mAP@k all images seed 233.	78
27	Proximity Experiment: mAP@k all images seed 44.	79
28	Proximity Experiment: mAP@k images with covid seed 13.	80
29	Proximity Experiment: mAP@k images with covid seed 134.	81
30	Proximity Experiment: mAP@k images with covid seed 2023.	82
31	Proximity Experiment: mAP@k images with covid seed 233.	83
32	Proximity Experiment: mAP@k images with covid seed 44.	84
33	Proximity Experiment: mAP@k images without covid seed 13.	85
34	Proximity Experiment: mAP@k images without covid seed 134.	86
35	Proximity Experiment: mAP@k images without covid seed 2023.	87
36	Proximity Experiment: mAP@k images without covid seed 233.	88
37	Proximity Experiment: mAP@k images without covid seed 44.	89
38	Proximity Experiment: mAP@k all images seed 13.	90
39	Proximity Experiment: mAP@k all images seed 134.	90
40	Proximity Experiment: mAP@k all images seed 2023.	90
41	Proximity Experiment: mAP@k all images seed 233.	91
42	Proximity Experiment: mAP@k all images seed 44.	91
43	Proximity Experiment: mAP@k images with covid seed 13.	91
44	Proximity Experiment: mAP@k images with covid seed 134.	92
45	Proximity Experiment: mAP@k images with covid seed 2023.	92
46	Proximity Experiment: mAP@k images with covid seed 233.	92
47	Proximity Experiment: mAP@k images with covid seed 44.	93
48	Proximity Experiment: mAP@k images without covid seed 13.	93
49	Proximity Experiment: mAP@k images without covid seed 134.	93
50	Proximity Experiment: mAP@k images without covid seed 2023.	94
51	Proximity Experiment: mAP@k images without covid seed 233.	94
52	Proximity Experiment: mAP@k images without covid seed 44.	94
53	Proximity Experiment: mAP@k all images seed 13.	95
54	Proximity Experiment: mAP@k all images seed 134.	95
55	Proximity Experiment: mAP@k all images seed 2023.	95

56	Proximity Experiment: mAP@k all images seed 233.	95
57	Proximity Experiment: mAP@k all images seed 44.	95
58	Proximity Experiment: mAP@k images with covid seed 13.	96
59	Proximity Experiment: mAP@k images with covid seed 134.	96
60	Proximity Experiment: mAP@k images with covid seed 2023.	96
61	Proximity Experiment: mAP@k images with covid seed 233.	96
62	Proximity Experiment: mAP@k images with covid seed 44.	96
63	Proximity Experiment: mAP@k images without covid seed 13.	97
64	Proximity Experiment: mAP@k images without covid seed 134.	97
65	Proximity Experiment: mAP@k images without covid seed 2023.	97
66	Proximity Experiment: mAP@k images without covid seed 233.	97
67	Proximity Experiment: mAP@k images without covid seed 44.	97

INTRODUCCIÓN

Recuperación de imágenes basadas en contenido, o bien por sus siglas en inglés CBIR (Content-Based Image Retrieval), corresponde a la consulta de imágenes dentro de una base de datos mediante el uso de un input, que es usado como referencia basándose en su estructura y/o características para recuperar imágenes similares a la entrada, dentro de la base de datos. La aplicación de este sistema es muy útil en la actualidad, incluso ya es implementado en varias aplicaciones como AliExpress, que cuenta con su búsqueda por imágenes. En este sistema, es posible tomar una imagen a un producto de internet y obtener imágenes similares a lo buscado. Si bien, probablemente ese caso no tenga tanto impacto en la actualidad, existen otros que sí, como lo es el caso de recuperación de imágenes médicas, para el apoyo en la toma de decisiones. Ayudando directamente a los médicos a diferenciar qué tipo de problema y/o enfermedad presenta un paciente basándose en casos similares.

Anteriormente, se desarrolló un sistema para abordar este último problema, denominado Proximity [Sotomayor *et al.*, 2021] el cual consta de la concatenación de dos redes neuronales: una de segmentación y otra de clasificación. Lamentablemente, Proximity y muchos otros sistemas CBIR asumen la disponibilidad de un gran número de imágenes etiquetadas por médicos expertos. Este supuesto es difícil de satisfacer en la práctica y fuerza a los desarrolladores a entrenar el sistema con conjunto de datos internacionales cuyas características pueden diferir de los datos nacionales para los que el sistema está siendo construido.

Proximity es tomado como baseline, replicando los experimentos y los modelos utilizados. Debido al gran crecimiento en popularidad de los métodos de tipo *Self-supervised Learning* para el área de visión por computadora y la nula necesidad de etiquetas en fase de entrenamiento, se decide implementar cuatro modelos distintos: *SimCLR* [Chen *et al.*, 2020], *BYOL* [Grill *et al.*, 2020], *SiamSiam* [Chen y He, 2021] y *Barlow Twins* [Zbontar *et al.*, 2021]. Replicando los experimentos de Proximity con estos métodos, se contrastan los resultados en términos de: precisión, tiempo y tamaño de embeddings.

Utilizando los modelos *SSL* (Self-supervised Learning), se obtiene una mejora considerable en relación con lo obtenido por Proximity, sin la necesidad de requerir de etiquetas ni de pesos preentrenados. Esto dota al modelo de la capacidad de ser utilizado posteriormente en cualquier otro tipo de problema, ya sea clasificación multi-clase, clustering, etc. Esto se debe a la naturaleza intrínseca de los métodos de *SSL*, que no solo se obtiene un modelo que puede ser utilizado para recuperación de imágenes, sino que también se puede expandir a múltiples tareas secundarias (downstream tasks) relacionadas con imágenes médicas.

La memoria se compone de 5 capítulos, donde se desarrolla la investigación realizada. La primera parte tiene relación con la definición del problema, la siguiente hace referencia al contexto en donde se enmarca la memoria, lo que corresponde a inteligencia artificial y al conocimiento básico necesario para seguir la memoria. La tercera parte ya tiene relación con la propuesta de solución, que es donde se expone Proximity y los métodos *Self-supervised*

investigados. La cuarta parte corresponde a los experimentos realizados con los respectivos resultados y, finalmente, la quinta parte se relaciona con las conclusiones y trabajos futuro de la memoria.

CAPÍTULO 1

DEFINICIÓN DEL PROBLEMA

1.1. Contexto

Cada día aumenta la cantidad de datos que generamos, los cuales se pueden clasificar en dos formatos principalmente: estructurado y no estructurado. El primero corresponde a lo que se aprecia en bases de datos relacionales, donde la información se guarda con un formato predefinido, como puede ser una tabla. El segundo hace referencia a información que no tiene un modelo de datos preestablecido, estos pueden ser imágenes, video, audio, texto, entre otros.

El proceso de etiquetado de los datos no estructurados por humanos se vuelve relativamente complejo, particularmente si consideramos el caso de las imágenes. Esta es una tarea monótona que presenta altos costos en términos de tiempo y dinero. También requiere que sea realizada por un especialista en el área e, incluso, a pesar de contar con un especialista, la etiqueta se encuentra propensa al error humano, causando que un enfoque tradicional de recuperación de información basado en etiquetas pierda datos que pueden ser relevantes para cierto problema.

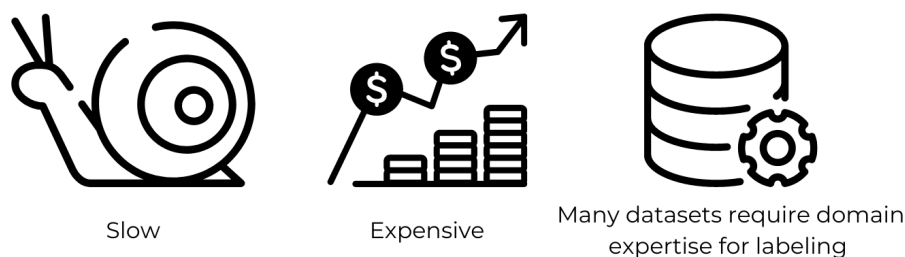


Figura 1: Principales problemas con el etiquetado manual.
Fuente: Elaboración propia.

Content-Based Image Retrieval (CBIR) intenta solucionar el problema mencionado anteriormente mediante la búsqueda de imágenes en grandes bases de datos, basándose únicamente en el contenido de estas. Esto se refiere a colores, formas, texturas u otra información derivada de una imagen que pueda servir como método de diferenciación. Para lograr esto, típicamente se utiliza una *query* que describe lo que el usuario está buscando, la cual puede ser representada como una imagen, un dibujo, una palabra clave o una combinación de éstas.

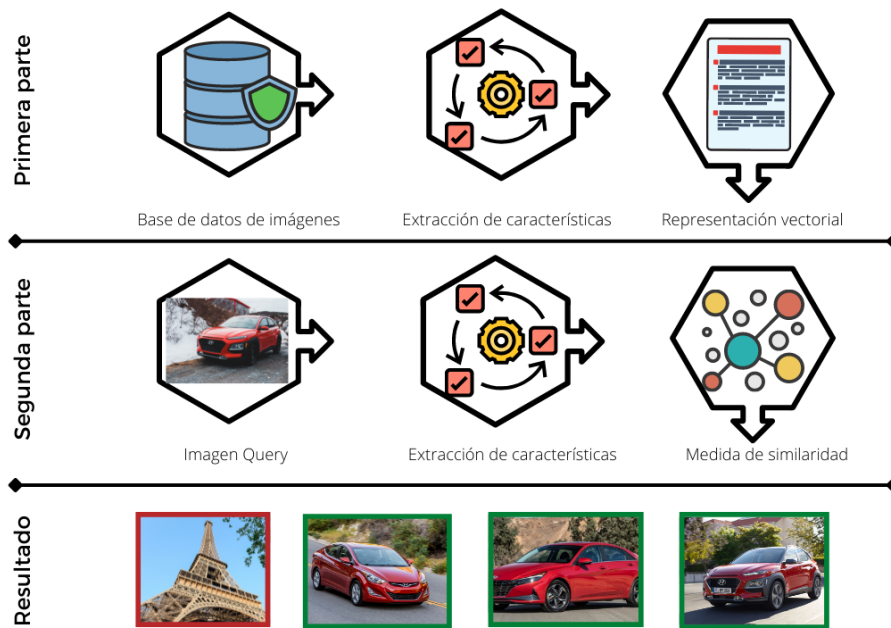


Figura 2: Ejemplo de un sistema CBIR.
Fuente: Elaboración propia.

En la figura 2 se muestra el proceso que se realiza en un sistema CBIR. Este proceso se puede considerar como un proceso de dos pasos.

En el primer paso, se toman todas las imágenes y se utiliza algún método, como pueden ser las redes neuronales, para realizar la extracción de características. Esto implica obtener una representación vectorial para cada una de las imágenes. La idea es que imágenes similares, como dos imágenes de autos rojos, tengan representaciones vectoriales similares y se encuentren cercanas en un espacio determinado. Por otro lado, imágenes diferentes, como un auto rojo y la Torre Eiffel, se encuentren alejadas en dicho espacio.

En el segundo paso, se utiliza una *query*, en este caso, un imagen de un auto de color rojo. Esta imagen de consulta se pasa por la misma red que realizó el mapeo en la primera parte, obteniendo así una representación vectorial para la *query*.

Finalmente, teniendo el vector de la *query* y los vectores correspondientes a las imágenes dentro de la base de datos, se calcula alguna medida de distancia o similitud, como la distancia euclidiana entre los vectores. En el caso de la figura 1, se obtienen tres imágenes que son relevantes para la *query* y una que no lo es.

El procedimiento mencionado anteriormente se puede aplicar en distintos ámbitos, como por ejemplo en imágenes médicas para ayudar a los patólogos a decidir en base a casos o patrones similares, qué tipo de enfermedad presenta un paciente [Li *et al.*, 2018]. Además se usa en re-identificación de personas [Ye *et al.*, 2021], lo que contribuye a la seguridad pública e incluso tiene aplicaciones prácticas en comercios online para generar principalmente

recomendaciones de ropa a los clientes [Liu *et al.*, 2016].

1.2. Objetivos

1.2.1. Objetivo General

Evaluar diferentes tipos de métodos actualmente disponibles en el estado del arte relacionados con el *Self-supervised Learning* para CBIR, con la finalidad de comparar los resultados obtenidos con un sistema ya existente de recuperación de imágenes médicas (Proximity) que depende fuertemente de datos etiquetados.

1.2.2. Objetivos Específicos

- Seleccionar al menos 4 métodos distintos relacionados con el *Self-supervised Learning*.
- Diseñar e implementar un conjunto de experimentos que permita evaluar los métodos seleccionados en términos de eficiencia y eficacia en tareas de CBIR.
- Seleccionar e implementar un conjunto de métricas ampliamente utilizadas en la literatura actual para CBIR.
- Contrastar los resultados obtenidos por los modelos que no requieren de etiquetas con los obtenidos por Proximity.

CAPÍTULO 2

MARCO CONCEPTUAL

2.1. Redes Neuronales

Una red neuronal puede expresarse simplemente como un sistema de procesamiento de información diseñado para imitar la estructura y las funciones del cerebro humano. Debido a que nuestras neuronas se han entrenado durante años para procesar información visual, generalmente, cuando se observa a un animal es fácil saber de qué tipo es, ya que desde pequeño se enseña a diferenciar a los animales en base a distintos patrones. Como las neuronas del cerebro son las que se encargan de procesar la información visual que se percibe y han sido entrenadas durante años, para una persona discernir lo que se está observando se vuelve una tarea trivial.

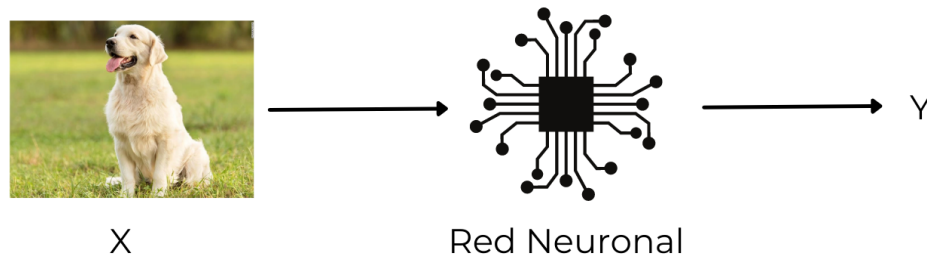


Figura 3: Ejemplo de una red neuronal.
Fuente: Elaboración propia.

En el ejemplo anterior, se recibe como input la imagen x , que representa lo que la persona está observando. El cerebro es la red neuronal encargada de mapear la entrada x a una salida y , que en este caso correspondería a “perro”, si consideramos que la tarea es reconocer qué tipo de animal se encuentra presente en la imagen.

La neurona es la unidad básica de procesamiento que se encuentra dentro de una red neuronal. Cada neurona, excepto la de entrada, recibe ciertos valores e implementa una regla de cálculo que permite determinar el estado de la neurona. El primer modelo computacional de neurona fue propuesto por McCulloch y Pitts en el año 1943 [McCulloch y Pitts, 1943].

Si denotamos por $x_1, x_2, \dots, x_d$ los datos que la neurona recibe en un instante de tiempo t , y por y_1 a la salida de la neurona después de recibir la información, el modelo MP (McCulloch y Pitts) se representa como:

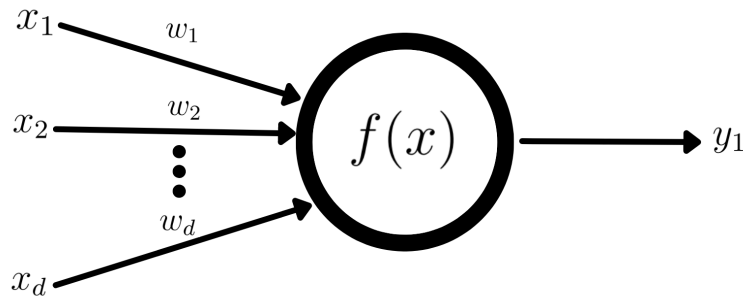


Figura 4: Modelo de neurona MP.

Fuente: Elaboración propia.

Donde:

- $f(x) = \sigma(\sum_{i=1}^d w_i x_i - b)$.
- $\sigma(\cdot)$ es la función de activación que en el modelo de MP, era la función de Heaviside, también conocida como escalón unitario.

$$\sigma(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ 0 & \text{si } x < 0 \end{cases}$$

- b corresponde a un parámetro característico de la neurona, denominado umbral de excitación o bias.
- $w_1, w_2, \dots, w_d$ corresponde a los pesos de conexión de la neurona.

La función de activación $\sigma(\cdot)$ es importante para que la red neuronal pueda modelar funciones complejas que no sean lineales. Es decir, la función de activación es la que se encarga de dotar a la red con la capacidad de aprender características no lineales. Sin funciones de activación, la respuesta de la neurona corresponde solo a funciones lineales, por lo que su complejidad es limitada y la capacidad de mapear funciones complejas es baja.

2.2. Funciones de Activación

Las funciones de activación más comunes en las redes neuronales son las siguientes:

- Sigmoid:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

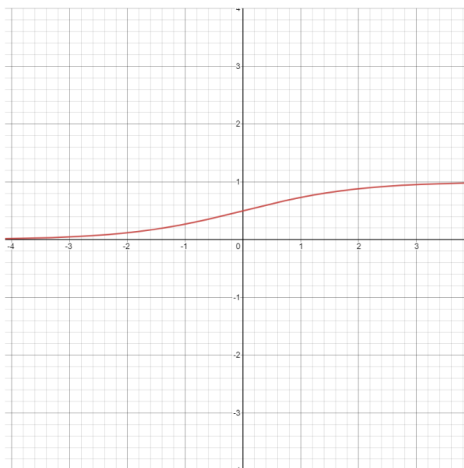


Figura 5: Función sigmoide.

Fuente: Elaboración propia.

- Tanh:

$$\sigma(x) = \frac{e^x + e^{-x}}{e^x + e^{-x}}$$

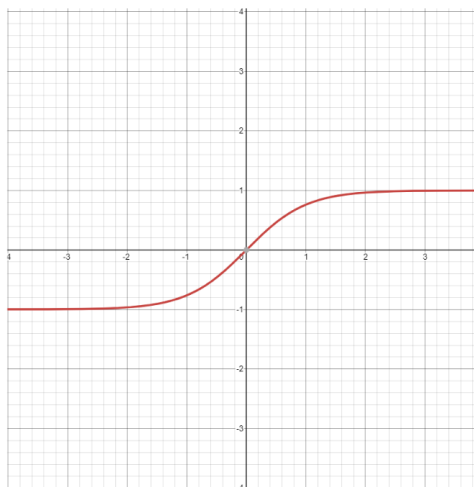


Figura 6: Función tangente hiperbólica.

Fuente: Elaboración propia.

- ReLU:

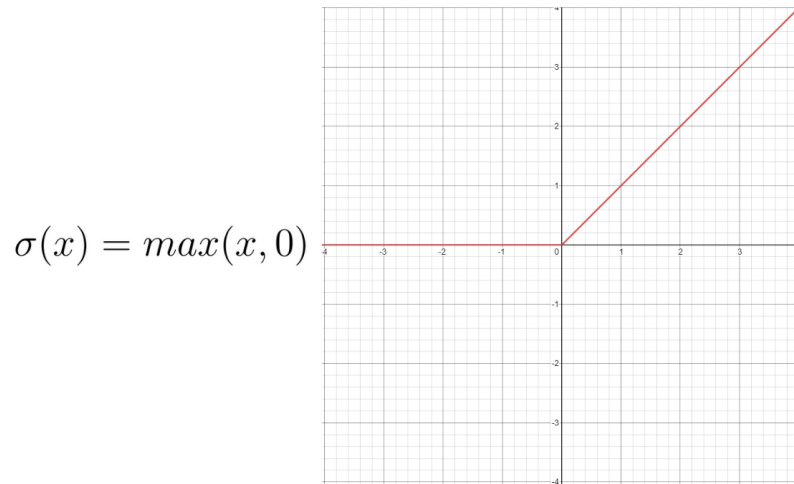


Figura 7: Función ReLU.
Fuente: Elaboración propia.

2.3. Redes Feedforward

La red más simple es la denominada Feedforward que tiene tres componentes principales: la capa de entrada (*Input*), las capas ocultas (*Hidden*) y la capa de salida (*Output*). En cada capa, la unidad fundamental es la neurona. La red cuenta con la característica de que cada neurona dentro de una capa se conecta con todas las neuronas de la siguiente capa.

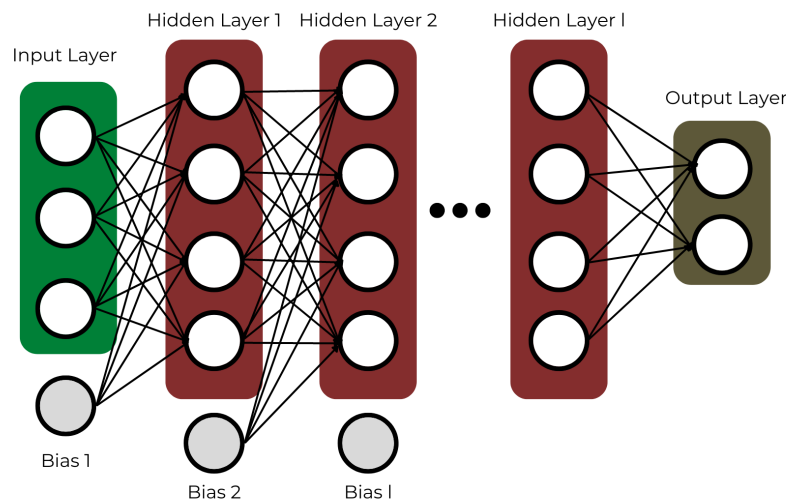


Figura 8: Ejemplo de una red Feedforward.
Fuente: Elaboración propia.

En la figura anterior, los círculos en blanco representan a las neuronas presentes en cada capa, los círculos en gris representan los sesgos (biases) de cada capa, y las conexiones entre

las capas representan los pesos de la red.

2.4. Redes Convolucionales

Una red que funciona muy bien para trabajar con datos relacionados a imágenes es la denominada CNN (*Convolutional Neural Network*), que deriva de las redes Feedforward. Antiguamente, para trabajar con imágenes en el área del machine learning se necesitaba de *handcrafted features*, es decir, que un experto del dominio se encargará de decidir qué características son importantes dentro de la imagen para obtener un mejor rendimiento al utilizar algún modelo de machine learning. Actualmente, esa tarea que se conoce como *feature engineering* para problemas con imágenes, es realizada por la red convolucional.

La red convolucional se basa en dos ideas fundamentales:

- **Campo receptivo local:** En general, se considera que la percepción humana parte desde el reconocimiento de patrones locales hacia patrones globales. Uno de los problemas que tiene una capa completamente conectada (capa Feedforward) es que cada neurona tiene acceso a la información de forma global, y como estamos trabajando con imágenes, es de esperar que las correlaciones espaciales entre los píxeles locales sean más cercanas que las que existen entre píxeles lejanos. Es por ello que cada neurona no necesita conocer toda la imagen, solo ciertas posiciones del patrón de entrada.
- **Compartición de parámetros:** Se pueden utilizar uno o más filtros/kernels para procesar el patrón de entrada. Los parámetros asociados a los filtros son los pesos y estos son compartidos. Dado que se comparten los pesos, se reducen la cantidad de parámetros aprendibles y además se consigue que el filtro trabaje en conjunto para detectar un determinado patrón de entrada.

Una red neuronal convolucional se compone típicamente de tres capas: *convolutional layer*, *pooling layer*, y *fully connected layer*.

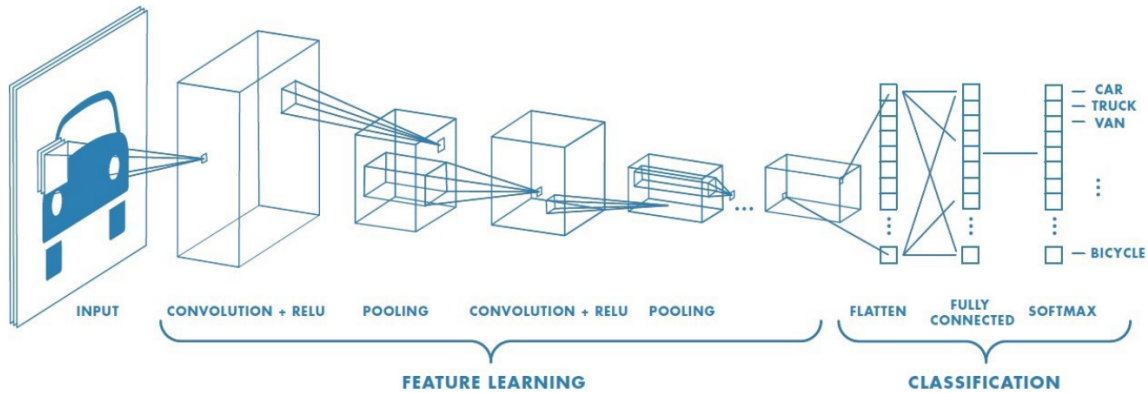


Figura 9: Arquitectura típica de una red convolucional.

Fuente: Sumit Saha.

El objetivo de la capa convolucional es extraer características de alto nivel, como lo son los bordes de una imagen. La capa convolucional requiere de la cantidad de filtros a utilizar y el tamaño del kernel. Si consideramos una imagen con un solo canal (imágenes en escala de grises), con un tamaño 5x5 y un filtro 2D con un kernel de 3x3 y strides de 1x1, una convolución discreta se vería de la siguiente forma:

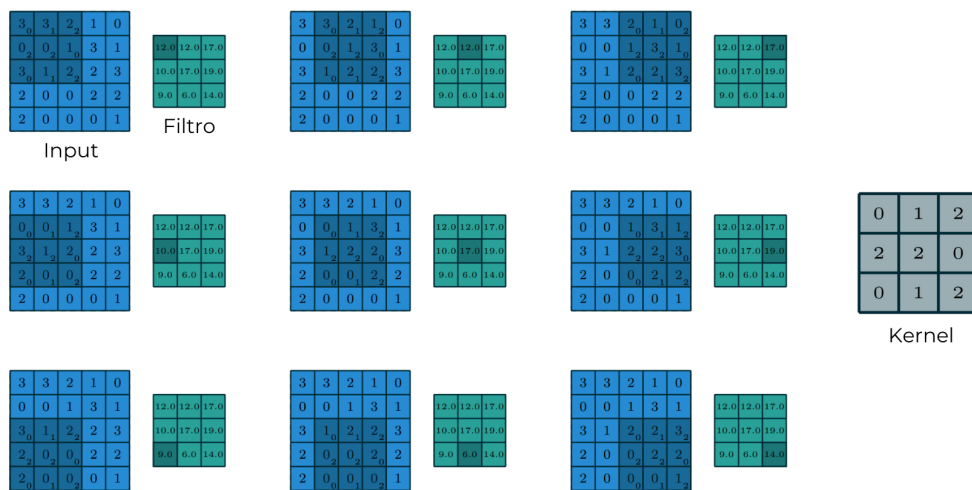


Figura 10: Ejemplo de una convolución discreta.

Fuente: [Dumoulin y Visin, 2016].

La capa de pooling se encarga principalmente de reducir la dimensión espacial del patrón de entrada manteniendo las características importantes. A diferencia de la capa convolucional, no tiene pesos por lo que no es aprendible, y no se acompaña de alguna función de activación como se aprecia típicamente en las capas convolucionales. Una capa de pooling solo requiere del tamaño del campo receptivo, stride y función a aplicar, la cual generalmente es

el promedio o máximo. Cuando se aplica como función el promedio en la capa, se le denomina como *Average Pooling* y cuando se utiliza como función el máximo, se le denomina como *Max Pooling*.

Si consideramos una imagen con un solo canal de 5x5 y aplicáramos la capa de *Average Pooling* y *Max Pooling* con una ventana de 3x3 y strides de 1x1 obtendríamos lo siguiente:

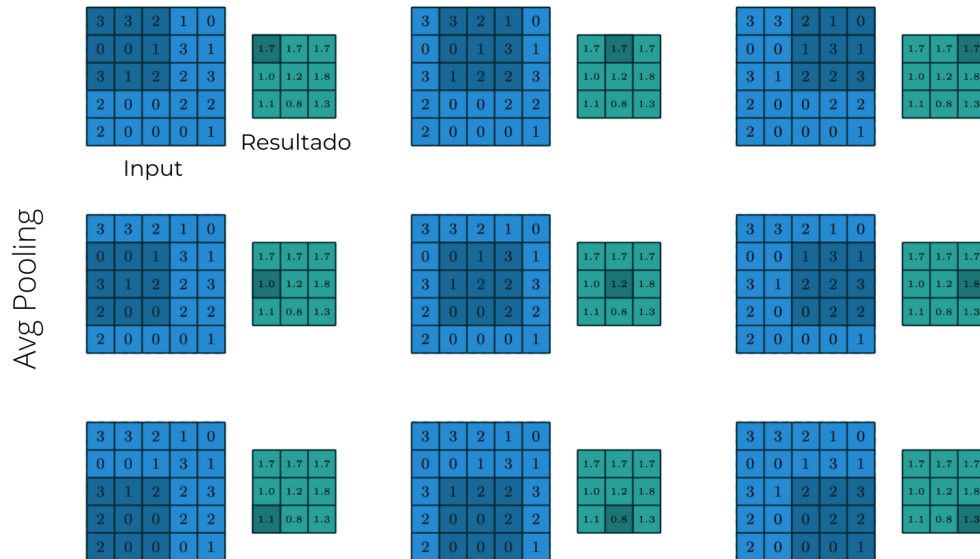


Figura 11: Ejemplo de average pooling.
Fuente: [Dumoulin y Visin, 2016].

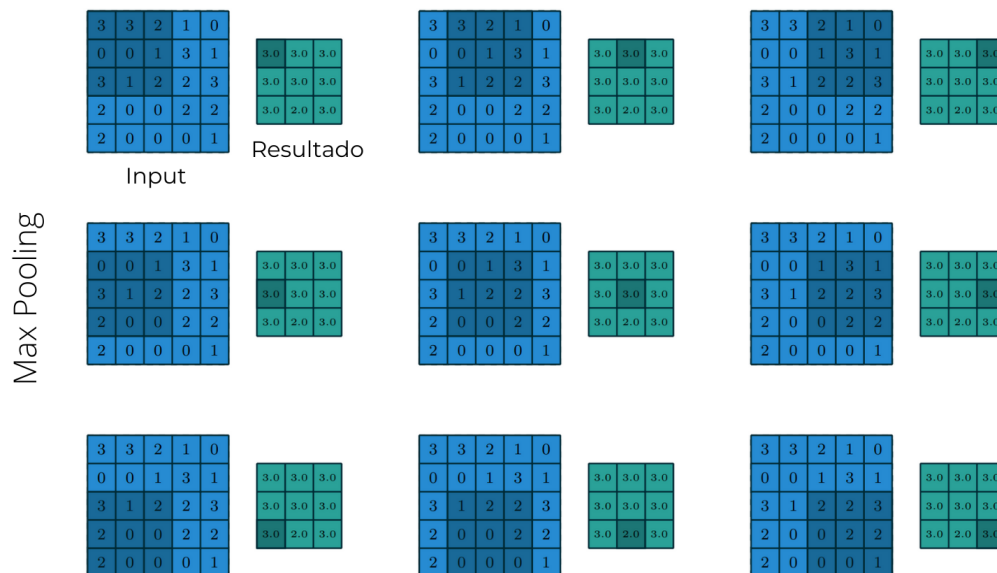


Figura 12: Ejemplo de max pooling.
Fuente: [Dumoulin y Visin, 2016].

2.4.1. Arquitecturas Clásicas De Redes Convolucionales

El campo de las redes convolucionales ha tenido grandes avances a través del tiempo. La siguiente imagen muestra una línea de tiempo con algunas de las arquitecturas más conocidas y populares de cada época.

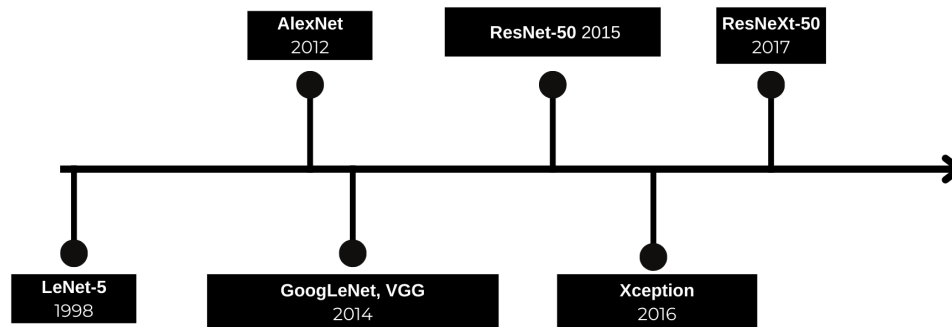


Figura 13: Timeline de algunas de las redes convolucionales más importantes y populares de su época.

Fuente: Elaboración propia.

LeNet-5 (1998)

Fue una de los primeros modelos de redes convolucionales, propuesto por Yann LeCun, Léon Bottou, Yoshua Bengio y Patrick Haffner en el año 1998. La primera vez que apareció esta arquitectura fue en el paper *Gradient-Based Learning Applied to Document Recognition* [LeCun et al., 1998], donde fue utilizada para el reconocimiento de imágenes de dígitos escritos a mano. La principal razón detrás de su popularidad fue lo simple que era la arquitectura, teniendo un total aproximado de tan solo 60.000 parámetros entrenables.

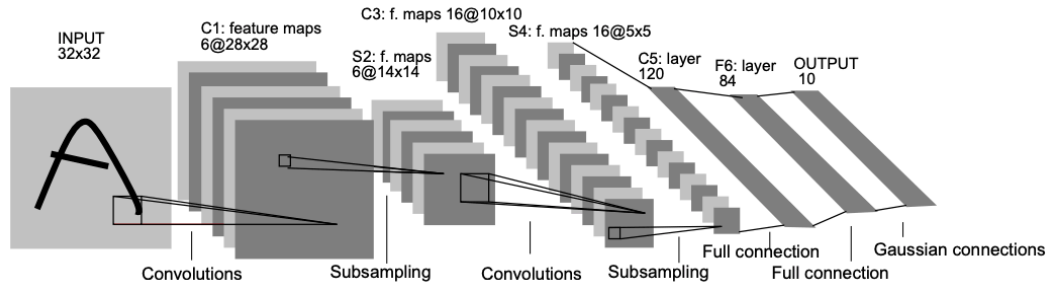


Figura 14: Arquitectura de LeNet-5.

Fuente: [LeCun *et al.*, 1998].

La arquitectura era simple considerando los modelos que serían introducidos más tarde. Como se puede ver en la imagen, constaba de solo 7 capas en total, sin contar la capa de input. La entrada consistía en imágenes de tamaño 32x32 con un solo canal, seguida de una capa de convolución con 6 filtros y kernel de tamaño (5, 5), seguida de una capa de average pooling de tamaño (2, 2), seguida por otra capa de convolución con 16 filtros y kernel de tamaño (5, 5), seguida por otra capa de average pooling con tamaño de (2, 2), y seguida finalmente por tres fully connected layer de tamaño, 120, 84, y 10 respectivamente.

Este patrón de arquitectura implementado en *LeNet-5* posteriormente se convirtió en un estándar para redes convolucionales, formando así el siguiente patrón:

1. Apilar en la red capas de convolución seguida por capas de pooling.
2. Finalizar la red con una o más capas fully connected.

Alexnet (2012)

Uno de los hitos más importantes ocurrió en 2012, cuando aparece una red llamada *AlexNet* [Krizhevsky *et al.*, 2017]. Esta red participó en el desafío de Imagenet [Deng *et al.*, 2009], consiguiendo un error Top-5 del 15.3% con aproximadamente 60 millones de parámetros, obteniendo así el primer lugar en esa competencia. Lo que fue histórico para la época, porque por primera vez una red neuronal convolucional fue capaz de superar los métodos clásicos de machine learning que se basaban en características extraídas por métodos como *SIFT* [Lowe, 1999].

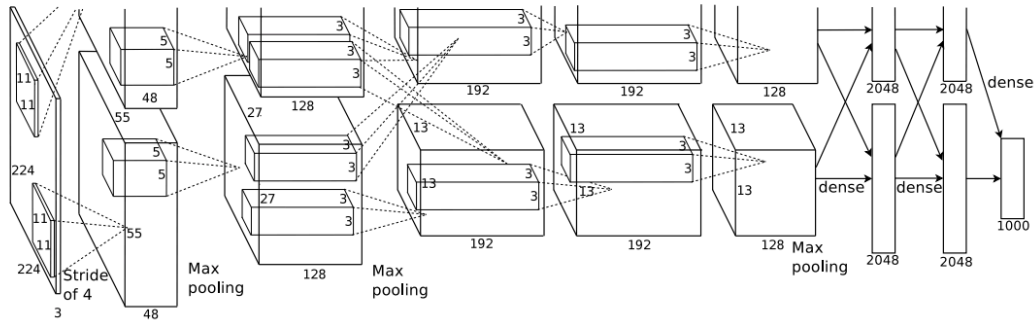


Figura 15: Arquitectura de Alexnet

Fuente: [Krizhevsky *et al.*, 2017].

Como se puede ver en la imagen, la arquitectura de *Alexnet* tenía en total 11 capas, si contamos las de pooling. Las primeras cuatro son una capa convolucional seguida por una capa max pooling, seguida de 3 capas convolucionales donde la última era seguida por otra capa de max pooling y finalmente contaba con 3 capas fully connected.

GoogLeNet, VGG (2014)

En 2014, Google presenta *GoogLeNet* [Szegedy *et al.*, 2015], también conocida como *inception* o *inception-v1*. Red convolucional que logró posicionarse como el nuevo estado del arte en la competencia de Imagenet 2014 (ILSVRC14) obteniendo un error Top-5 del 6.7 %.



Figura 16: Arquitectura de GoogLeNet.
Fuente: [Szegedy *et al.*, 2015].

La arquitectura contaba con alrededor de 22 capas. La innovación principal era que optimizaba al máximo los recursos computacionales dentro de la red y que también incorporaba un nuevo modulo que denominaron *inception*.

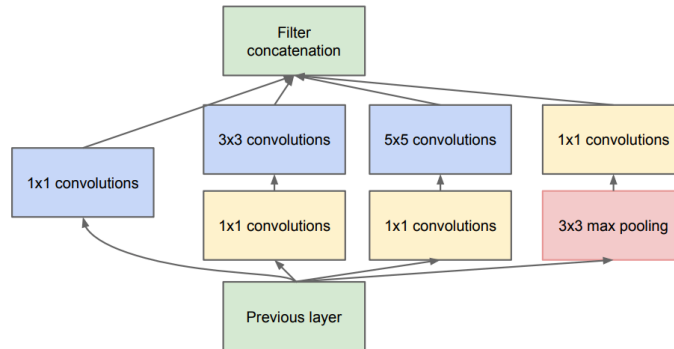


Figura 17: Módulo inception.

Fuente: [Szegedy *et al.*, 2015].

Asimismo, en ese año se presenta VGG [Simonyan y Zisserman, 2014], red convolucional que se caracterizaba por seguir un patrón similar a LeNet, múltiples capas convolucionales seguidas de capas de pooling, finalizando con capas fully connected. Se plantea una arquitectura base a la cual se le varía la profundidad (cantidad de capas) entre 11 hasta 19, lo que se conoce respectivamente como VGG-11 hasta VGG-19.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv1-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figura 18: Arquitectura de VGG-11 hasta VGG-19.

Fuente: [Szegedy *et al.*, 2015].

La cantidad de parámetros entrenables variaba entre 133 millones para VGG-11 hasta 143 millones para VGG-19 respectivamente. La principal contribución de la red VGG fue la comprobación experimental de la importancia de la profundidad en las redes convolucionales.

ResNet-50 (2015)

Si bien se sabía que la profundidad en las redes convolucionales era de gran ayuda, existía un límite principalmente por el problema del desvanecimiento del gradiente, lo cual tiene relación con que a medida que más capas se van apilando en una red que usen ciertas funciones de activación como lo es la sigmoide, los gradientes se acercan a cero, haciendo que se vuelva difícil la actualización de los pesos y por ende el entrenamiento de la red.

Es ahí donde aparece una idea bastante innovadora con *ResNet-50* [He et al., 2016], la cual introduce la idea de *skip connections* en redes convolucionales.

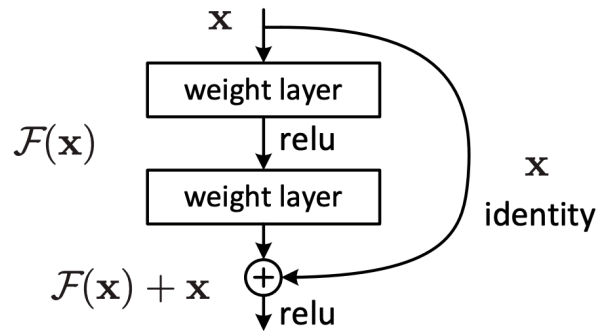


Figura 19: Bloque residual (Skip connection).
Fuente: [He et al., 2016].

Con esa idea innovadora de *skip connections* ResNet fue capaz de obtener el primer lugar en la competencia de ILSVRC y COCO en el año 2015. Las *skip connections* dotan a la red con la capacidad de crear arquitecturas con mayor profundidad sin perder mucho poder de generalización en la red por el ya conocido problema del desvanecimiento del gradiente.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	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 20: Arquitectura de ResNet-18 hasta ResNet-154.
Fuente: [He et al., 2016].

Xception (2016)

En el año 2016 *Xception* logró posicionarse como el nuevo estado del arte por encima de VGG, ResNet e Inception-V3. La mayor innovación fue la introducción del uso de *Depthwise Separable Convolutions*. Esto corresponde a un bloque convolucional que se divide en dos pasos. Primero se utiliza *Depthwise Convolution*, que es realizar una convolución por cada canal del input de forma independiente, seguido por *Pointwise Convolution* que es realizar una convolucion 1x1 proyectando el output de los canales de *Depthwise Convolution* en un nuevo espacio. Tomando esa idea decidieron invertir el orden, es decir realizar primero *Pointwise Convolution* y luego *Depthwise Convolution*.

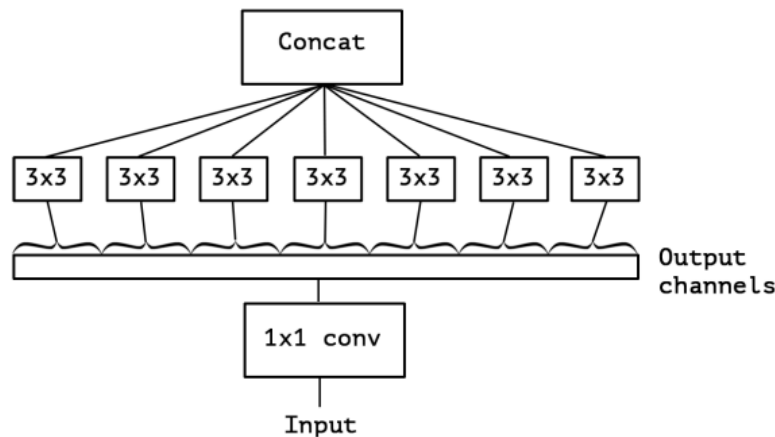


Figura 21: Pointwise Convolution seguida de Depthwise Convolution.
Fuente: [Chollet, 2017].

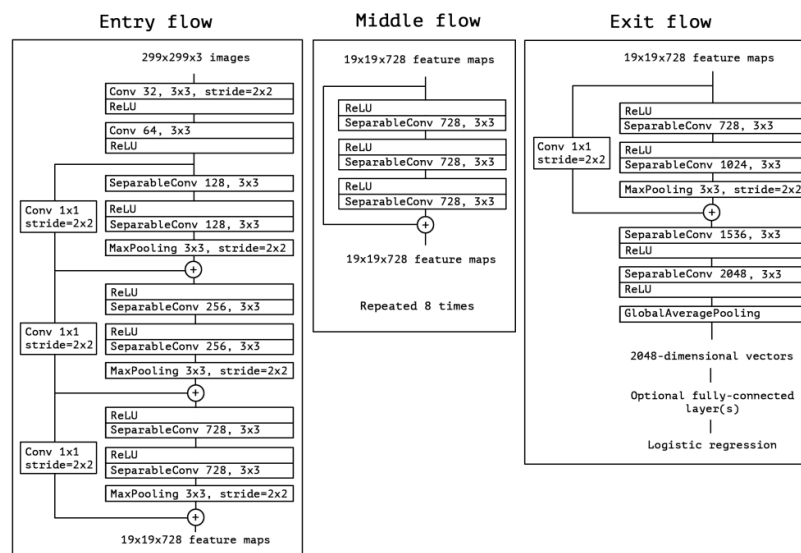


Figura 22: Arquitectura de Xception.
Fuente: [Chollet, 2017].

ResNeXt-50 (2017)

La principal innovación en *ResNeXt* [Xie *et al.*, 2017] corresponde a la incorporación de un nuevo componente en el diseño de la arquitectura de la red, lo que denominan *Cardinality*. Esta red se basa principalmente en *ResNet*, tomando la idea de las skip connections e introduciendo su nuevo concepto de cardinalidad, el cual define la cantidad de transformaciones realizadas en cada bloque de la red.

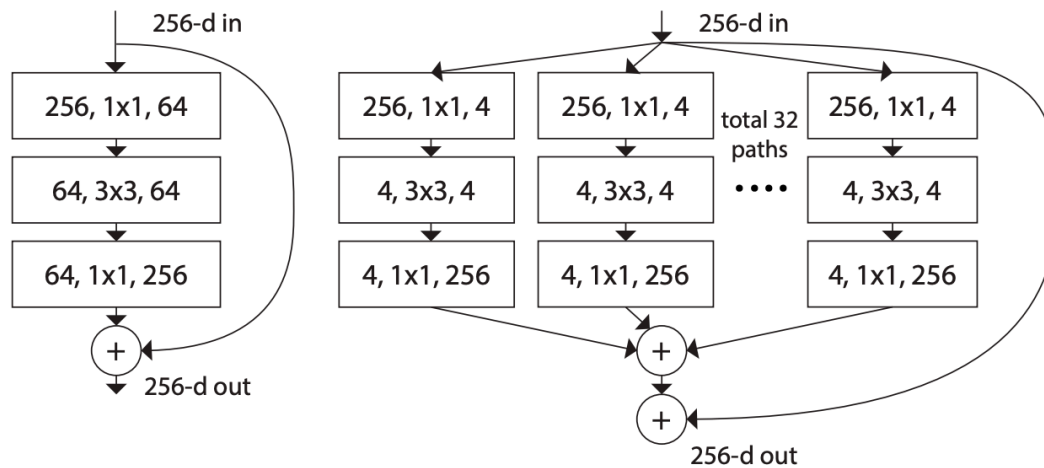


Figura 23: Bloque convolucional típico en ResNeXt.

A la izquierda se ve un bloque convolucional típico de ResNet y a la derecha su equivalente en ResNeXt con $C = 32$ (Cardinality).

Fuente: [Xie *et al.*, 2017].

La arquitectura de la red sigue prácticamente el mismo patrón que *ResNet* pero con la diferencia de incluir ese parámetro de cardinalidad. Esto significa utilizar bloques que agregan un conjunto de transformaciones con la misma topología. Experimentalmente muestran que bajo las mismas condiciones de complejidad, incrementar la cardinalidad mejora la precisión de clasificación e incluso, incrementar la cardinalidad en una red es mucho más efectivo que crear una arquitectura más profunda o más ancha. Esto último hace referencia al número de neuronas o canales en una capa particular de la red, lo cual se puede lograr incrementando la cantidad de filtros en capas convolucionales o aumentando el número de neuronas en las capas completamente conectadas.

stage	output	ResNet-50	ResNeXt-50 (32×4d)
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2	3×3 max pool, stride 2
		$\begin{bmatrix} 1\times 1, 64 \\ 3\times 3, 64 \\ 1\times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times 1, 128 \\ 3\times 3, 128, C=32 \\ 1\times 1, 256 \end{bmatrix} \times 3$
conv3	28×28	$\begin{bmatrix} 1\times 1, 128 \\ 3\times 3, 128 \\ 1\times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times 1, 256 \\ 3\times 3, 256, C=32 \\ 1\times 1, 512 \end{bmatrix} \times 4$
conv4	14×14	$\begin{bmatrix} 1\times 1, 256 \\ 3\times 3, 256 \\ 1\times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times 1, 512 \\ 3\times 3, 512, C=32 \\ 1\times 1, 1024 \end{bmatrix} \times 6$
conv5	7×7	$\begin{bmatrix} 1\times 1, 512 \\ 3\times 3, 512 \\ 1\times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times 1, 1024 \\ 3\times 3, 1024, C=32 \\ 1\times 1, 2048 \end{bmatrix} \times 3$
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
# params.		25.5 ×10 ⁶	25.0 ×10 ⁶
FLOPs		4.1 ×10 ⁹	4.2 ×10 ⁹

Figura 24: Arquitectura de ResNet vs ResNeXt.

Fuente: [Xie et al., 2017].

Las redes neuronales convolucionales han ido mejorando cada vez más a través del tiempo, principalmente por el aumento de poder de cómputo y el interés constante en el área por mejorar y/o optimizar las arquitecturas existentes.

2.5. Tipos de entrenamiento en una red neuronal

Si bien hay varios tipos de entrenamiento de redes neuronales, la mayoría se puede clasificar en cuatro grandes grupos:

- Aprendizaje Supervisado o *Supervised Learning*: Se caracteriza por tener data etiquetada al entrenar un modelo de machine learning, este caso es uno de los problemas más comunes en la actualidad y se puede dividir en dos grandes categorías:
 - Regresión: Un caso típico es cuando se tiene como objetivo tratar de predecir el precio de una casa en base a distintas características como la cantidad de dormitorios, cantidad de baños, cantidad total de metros cuadrados, etc.

- Clasificación: Considerando el primer ejemplo presentado en la sección redes neuronales, un caso de clasificación sería predecir si en una imagen se encuentra presente un perro o no.

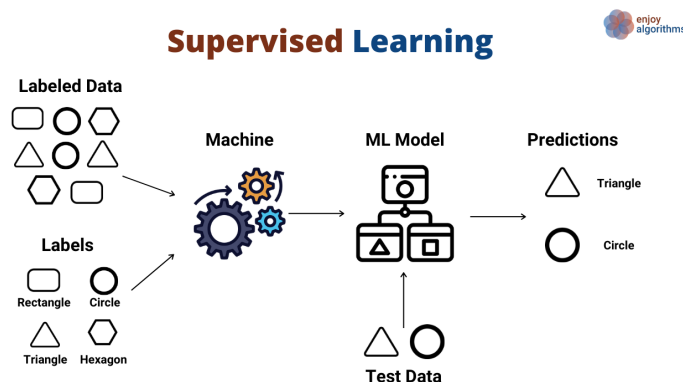


Figura 25: Aprendizaje Supervisado.
Fuente: Enjoy Algorithms.

- Aprendizaje No Supervisado o *Unsupervised Learning*: Es exactamente lo contrario a lo anterior, en este caso la data no se encuentra etiquetada, es por ello que el algoritmo utilizado de machine learning tiene que tratar de obtener patrones similares en base a la data que se pasa como input al modelo. Uno de los casos más comunes de este tipo de aprendizaje es el denominado clustering, considerando la imagen que se encuentra a continuación, la idea del clustering sería agrupar las figuras en base a su forma, de tal manera que los rectángulos se encuentren juntos, los círculos también y así sucesivamente.

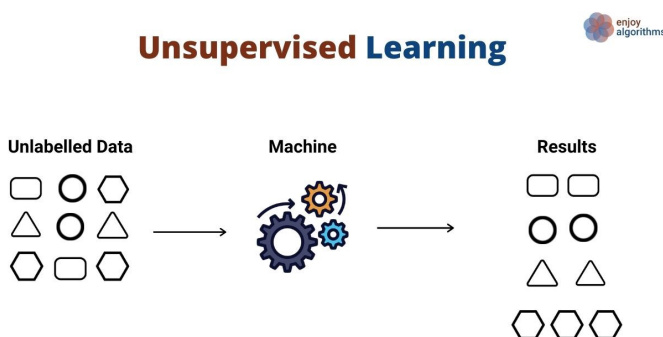


Figura 26: Aprendizaje No Supervisado.
Fuente: Enjoy Algorithms.

- Aprendizaje Semi-Supervisado o *Semi-supervised Learning*: Consiste en un aprendizaje que es una mezcla entre los dos anteriores, es decir, tenemos data etiquetada como también no etiquetada.

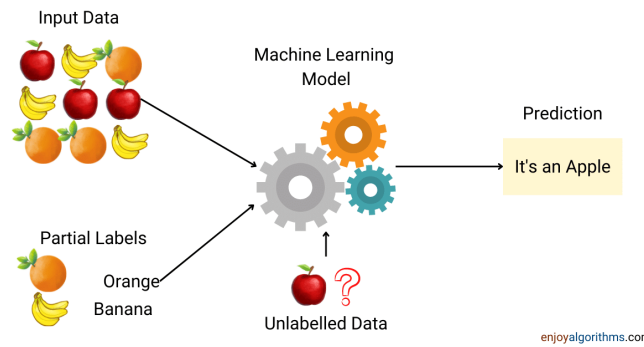


Figura 27: Aprendizaje Semi-Supervisado.
Fuente: Enjoy Algorithms.

Un caso de aplicación de este tipo de aprendizaje sería el llamado *pseudo-labeling*. Si tenemos un dataset que contiene muchas imágenes sin etiqueta pero contamos con algunas de ellas, lo que haríamos es entrenar un modelo con la data que sí tiene etiquetas y usar ese modelo para obtener etiquetas de la data que no las contiene y así generar una pseudo-etiqueta. Finalmente, se reentrena todo el modelo como si fuera aprendizaje supervisado, es decir utilizando la data con etiquetas y la que tiene las pseudo-etiquetas.

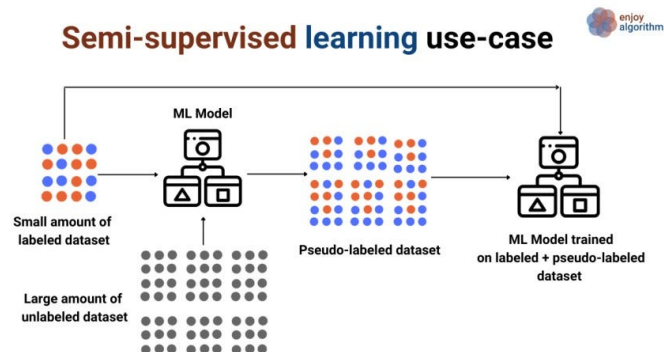


Figura 28: Caso de uso del aprendizaje semi-supervisado.
Fuente: Enjoy Algorithms.

- **Aprendizaje Reforzado o Reinforcement Learning:** En este caso el modelo de machine learning actúa como un agente virtual que se encuentra en un estado determinado dentro de un medio ambiente específico. Estos agentes tienen acciones o opciones que tomar y la idea es que seleccione la mejor opción o bien la que le otorga la mayor recompensa. El algoritmo pasa a través de ese proceso iterativo de estado, acción y recompensa, donde eventualmente termina aprendiendo.

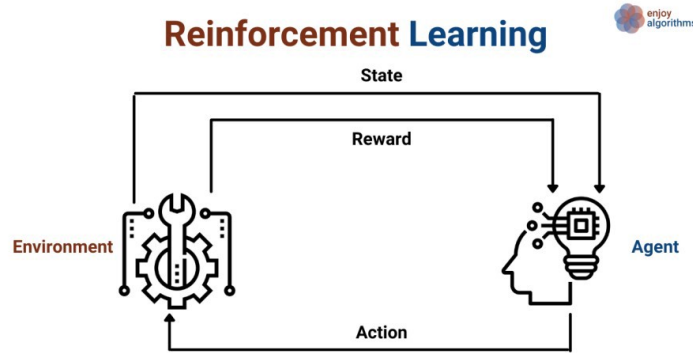


Figura 29: Aprendizaje Reforzado.
Fuente: Enjoy Algorithms.

Se popularizó en el año 1996 cuando el 10 de febrero de ese año, Deep Blue le ganó una partida de ajedrez al ruso Gari Kaspárov, número uno del ranking mundial desde 1985. Volvió a crecer la popularidad años más tarde, cuando en el año 2016, DeepMind fue capaz de crear AlphaGo usando aprendizaje reforzado. AlphaGo era una IA para jugar al juego de mesa Go que fue capaz de ganarle a un jugador profesional de Go. Hecho histórico para la época al ser la primera IA en derrotar a un humano en el juego de Go, un juego mucho más complejo que el ajedrez.

La principal ventaja del aprendizaje supervisado es que obtiene una muy buena precisión pero presenta un par de inconvenientes:

- Obtener data etiquetada en varios casos es difícil, ya sea porque se requiere de un experto del dominio para etiquetarla o bien es un proceso muy lento.
- Entrenar modelos supervisados puede tomar mucho tiempo y se requiere de mucha data etiquetada para obtener buena precisión.
- Los errores humanos o sesgo también se aprenden, así que si el etiquetador se encuentra sesgado, puede transmitir ese sesgo al modelo así como también los errores que él cometa.

2.5.1. Self-supervised Learning

También existen derivaciones de estos tipos de entrenamiento, un caso de ello es el denominado *Self-supervised Learning* (SSL). Algunos dicen que es parte del *Unsupervised Learning* y otros que no, por lo que aun no existe realmente un consenso acerca de este tema.

Este tipo de entrenamiento no es más ni menos que un aprendizaje donde se aprende una representación "útil" de la data de entrada sin depender de anotaciones humanas. (Útil hace

referencia a que se obtiene una representación que luego puede ser usada para alguna tarea en específico como clasificación, segmentación, clustering, etc.).

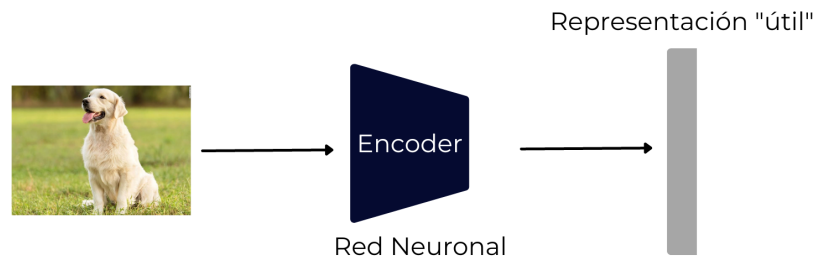


Figura 30: Representación útil Self-supervised.
Fuente: Elaboración Propia.

El encoder es usualmente una red neuronal, típicamente se utilizan modelos clásicos como alguna variante de *ResNet*, por ejemplo *ResNet-50*, pero puede ser cualquiera de las redes convolucionales, ya sean las clásicas mencionadas con anterioridad, una red convolucional propia o bien nuevas arquitecturas convolucionales del estado del arte.

Luego de obtener la representación útil, la idea es hacer *fine-tuning* con cualquier tipo de aprendizaje para alguna downstream task. En el caso del ejemplo, la downstream task podría ser el clásico problema de clasificación, es decir, predecir si la imagen de entrada corresponde a un perro o no. Para ello, se requiere tomar el encoder que en este caso hipotético es *ResNet-50* con los pesos ya entrenados al utilizar el modelo que usa *Self-supervised Learning* (esto se explica más adelante) y hacer una clasificación clásica utilizando los pesos de *ResNet-50*. Para el caso de CBIR, se puede utilizar directamente la representación útil obtenida por el método o alguna otra representación cercana a la función de pérdida, como un embedding para términos de recuperación de imágenes.

Este concepto de aprendizaje no es nuevo y ha sido un método que ha contribuido a grandes avances en el procesamiento del lenguaje natural. En el año 2000, Bengio y compañía introducen "A Neural Probabilistic Language Model" [Bengio *et al.*, 2000], un modelo que se basa en predecir una palabra desconocida asignando probabilidades a secuencias de palabras. En los años posteriores, llega uno de los modelos más populares actualmente de NLP que también se basa en auto-supervisión, BERT [Devlin *et al.*, 2018]. Para entrenar el modelo, en la tarea que denominan pre-training se utiliza la data de BooksCorpus (alrededor

de 800M de palabras) y la data de Wikipedia en inglés (alrededor de 2.500M de palabras). El método utiliza toda esa data para obtener la representación útil y posteriormente usar esa representación en otras tareas adicionales como question answering, text summarization, text generation, sentiment analysis, etc.

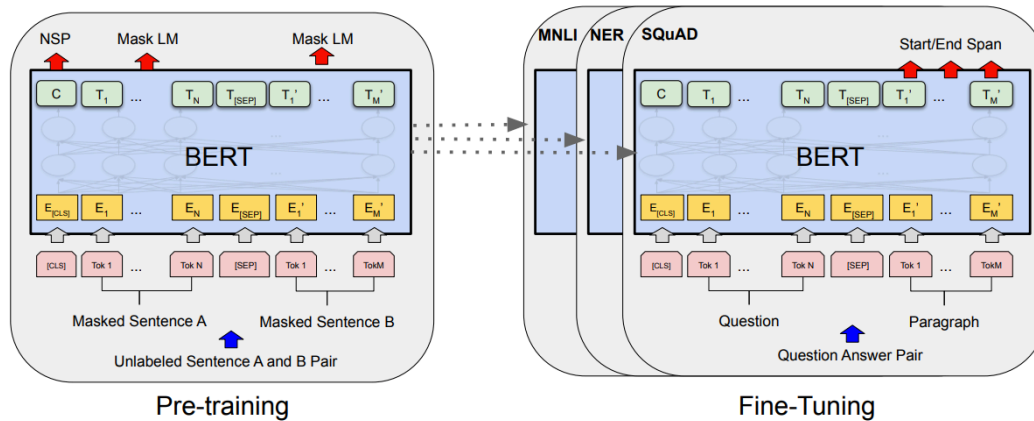


Figura 31: Pre-training y fine-tuning en BERT.

Fuente: [Devlin et al., 2018].

La clave de su éxito fue el concepto de SSL, ya que introducen dos tipos de tareas en la fase de pre-training que se basan en ese tipo de entrenamiento:

- **MLM (Masked Language Model):** Al momento de entrenar el modelo, el generador de la data elige el 15 % de los token de la sentencia para ser reemplazados de la siguiente manera:
 - El 80 % de las veces se reemplazan los token por [MASK] token.
 - El 10 % de las veces se reemplazan los token por cualquiera de forma aleatoria.
 - El 10 % de las veces no se reemplazan los token.
- **NSP (Next Sequence Prediction):** Consiste en escoger dos sentencias A y B. El 50 % de las veces son sentencias correlativas (Se le asigna el label *IsNext*) y el otro porcentaje de las veces se elige una sentencia al azar desde el corpus (Se le asigna el label *NotNext*).

Luego, para realizar fine-tuning, basta con agregar una capa adicional a la arquitectura y pasar los inputs y outputs relacionados con la downstream task. Con esta innovación, BERT logró posicionarse como el estado del arte en NLP y actualmente es un strong baseline dentro del área.

Por otra parte, una de las primeras apariciones del SSL en imágenes se le podría atribuir al modelo de autoencoder, ya que trata de reconstruir una imagen píxel por píxel, obteniendo una representación latente que sea útil para alguna tarea específica.

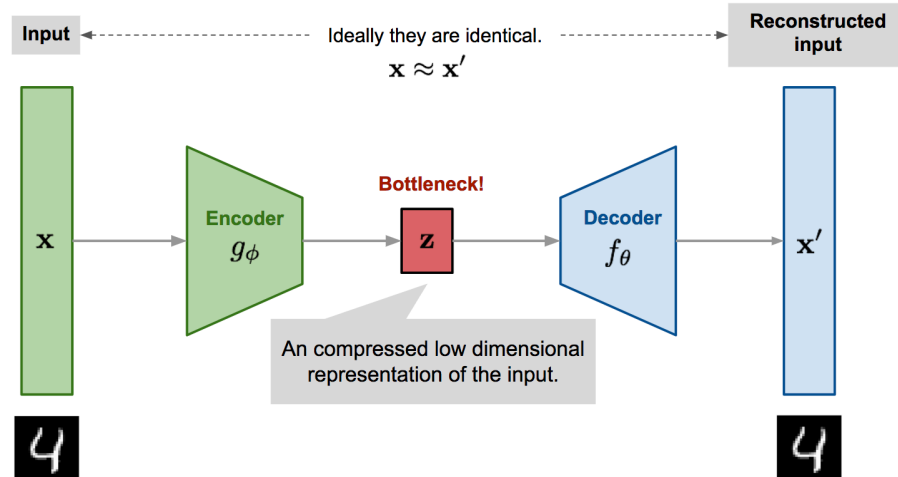


Figura 32: Arquitectura del autoencoder.
Fuente: Lilian Weng.

El autoencoder consiste en una arquitectura de tipo encoder-decoder, donde en el medio se encuentra la representación útil o en este caso el cuello de botella, que es una representación compacta de menor dimensionalidad que representa a la imagen de entrada.

Actualmente, existe una estructura que ha sido tendencia en los últimos años para el uso de SSL en imágenes. Esta corresponde a “SimCLR” [Chen *et al.*, 2020] la cual será explicada con mayor detalle en capítulos posteriores. Consiste en entrenar un par de redes neuronales a las cuales se les pasa una versión distorsionada distinta de la misma imagen de entrada. Estas versiones distorsionadas se obtienen mediante la técnica de data augmentation.

Este patrón de arquitectura se podría dividir en tres etapas generales:

1. Aumento de datos o *Augmentation*: Consiste en generar dos versiones distintas distorsionadas de la imagen de entrada mediante *data augmentation* de manera estocástica, es decir, aplicar distintas transformaciones con una cierta probabilidad. Las transformaciones más típicas que se utilizan para ello son:
 - Color Jitter: Cambia de forma aleatoria el brillo, la saturación y otras propiedades de la imagen.

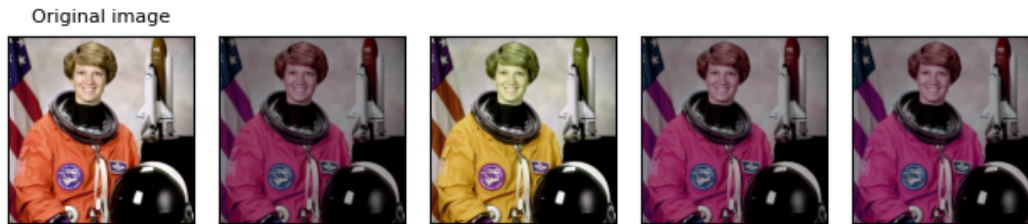


Figura 33: Color Jitter.
Fuente: PyTorch.

- Random Grayscale: De forma aleatoria cambia la imagen a escala de grises, en base a una probabilidad.

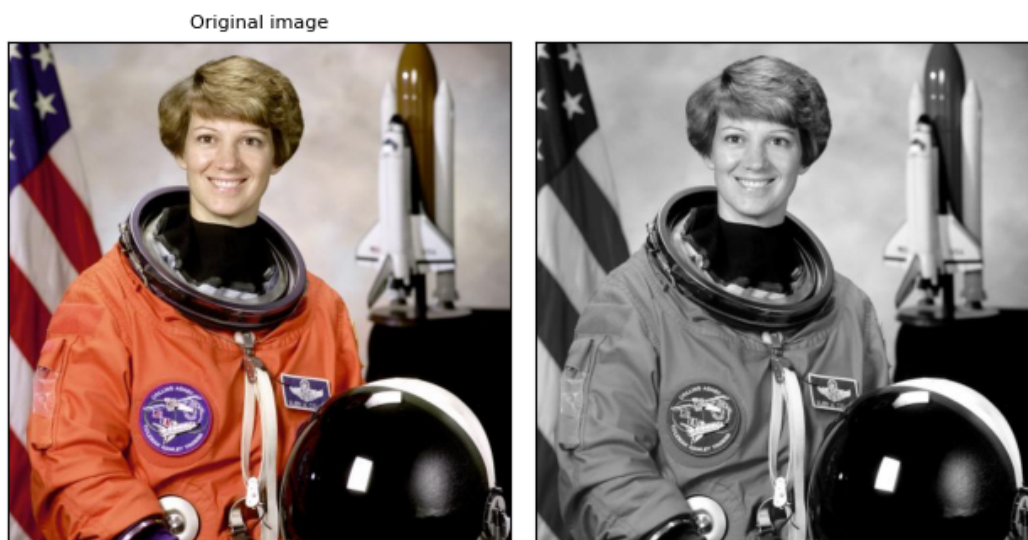


Figura 34: Random Grayscale.
Fuente: PyTorch.

- Random Horizontal Flip: Realiza de manera aleatoria un flip horizontal a la imagen, en base a una probabilidad.

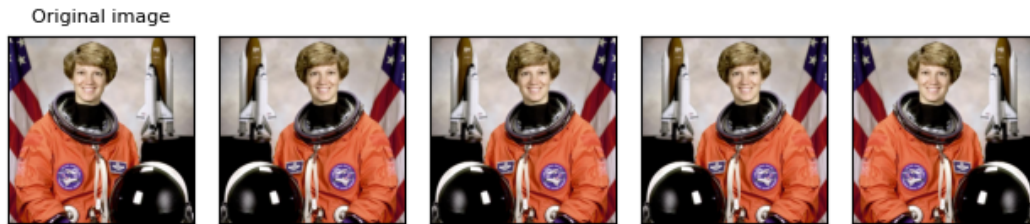


Figura 35: Random Horizontal Flip.
Fuente: PyTorch.

- Gussian Blur: Aplica Gaussian Blur a la imagen (Desenfoque gaussiano).

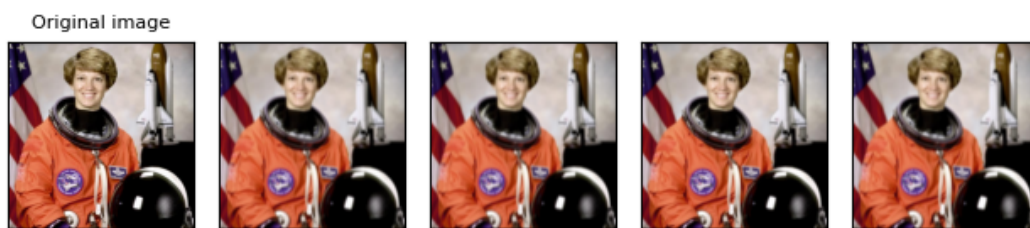


Figura 36: Gaussian Blur.
Fuente: PyTorch.

- Random Resized Crop: Recorta una imagen en un lugar aleatorio y, a continuación, redimensiona el corte a un tamaño determinado, que suele ser el mismo de la imagen de entrada.

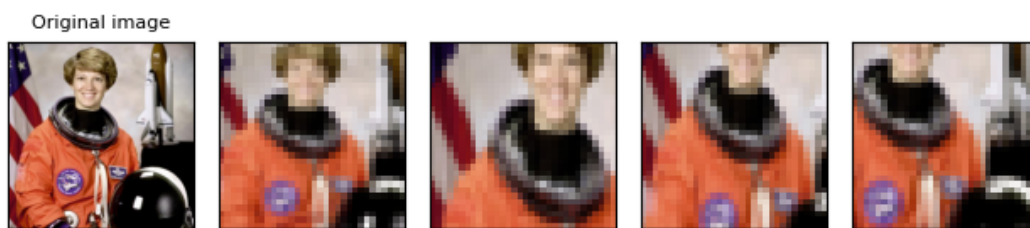


Figura 37: Random Resized Crop.
Fuente: PyTorch.

De esta forma se genera un par positivo, es decir, ambas imágenes son iguales debido a que son transformaciones de una misma imagen. La idea es que posteriormente el modelo aprenda esa relación: que las dos vistas pertenecen a lo mismo.

2. Encoder: Cada una de la imágenes obtenidas por medio de *data augmentation* se pasa por una red neuronal encargada de obtener una representación vectorial de la imagen, es por ello por lo que se denomina encoder.

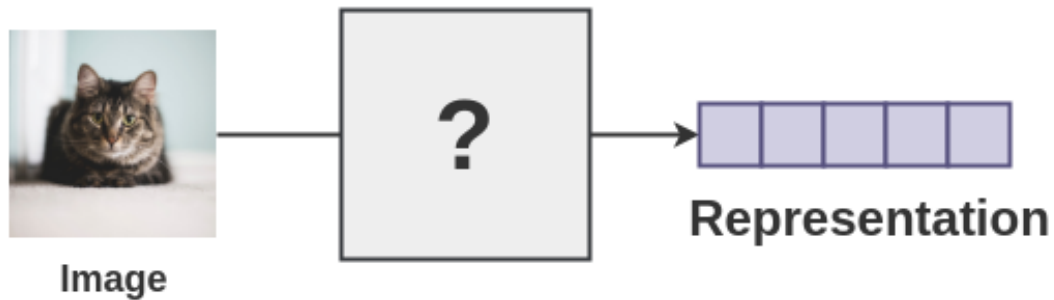


Figura 38: Encoder en SSL.
Fuente: Amit Chaudhary.

3. Minimizar la loss de representación: Finalmente al tener las dos representaciones vectorial, donde ambas son transformaciones distintas de la misma imagen, la idea es utilizar alguna función de pérdida que tome en cuenta la similitud entre los vectores, y se maximice esa métrica. Un ejemplo de esto sería la similitud coseno.

El proceso completo de SSL en imágenes se vería de la siguiente manera:

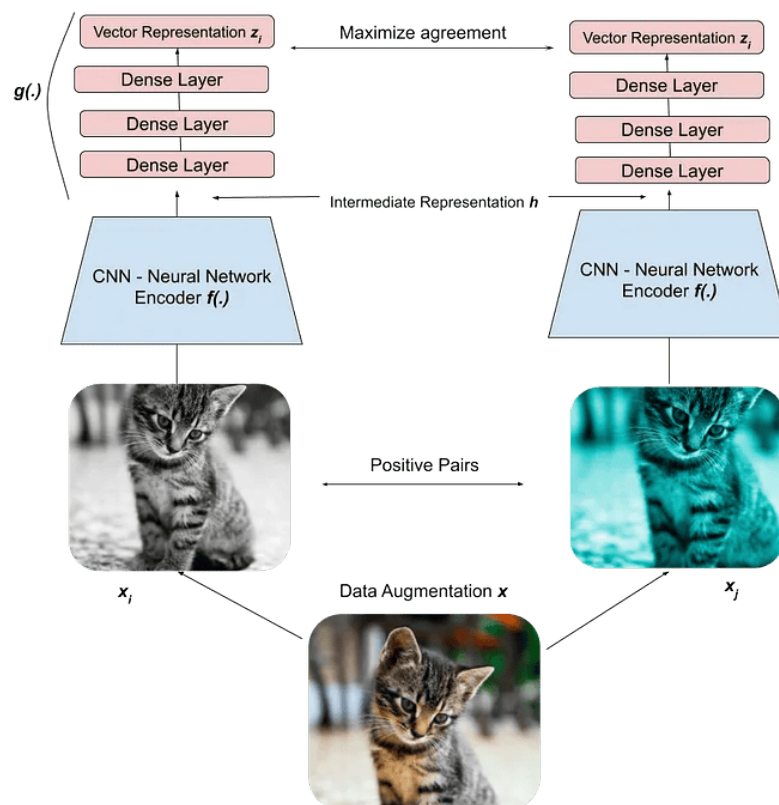


Figura 39: Proceso completo de SSL en imágenes.
Fuente: Ekin Tiü.

Para el caso de CBIR, al igual que en algún otro modelo, se puede utilizar cualquier capa como representación para la obtención de embeddings. En esta memoria se investiga el uso de h y z específicamente, que corresponden a la representación del encoder y la última capa del modelo para obtener los embeddings.

Como bien se mencionó anteriormente, el aprendizaje supervisado presenta un par de inconvenientes, es por ello por lo que *Self-supervised* ha sido muy popular últimamente. Al no requerir etiquetas se vuelve muy útil en el ámbito médico, ya que entrenar una red neuronal siempre ha tenido dificultades en base a la limitada data etiquetada que se encuentra disponible y al tiempo y/o costo de obtener buenas anotaciones para la data.

Aunque hay que considerar que aún presenta un par de inconvenientes:

- **Precisión:** Si bien en algunos casos obtiene una buena precisión, aún el aprendizaje supervisado obtiene mejores resultados en la mayoría de los casos.
- **Eficiencia computacional:** Entrenar un modelo usando una pretext task y luego hacer fine-tuning es más costoso ya que requiere de dos entrenamientos separados.
- **Cantidad de datos:** Si bien no se requieren etiquetas, todavía se requiere de una gran cantidad de data no etiquetada para obtener un buen resultado, lo que no siempre es posible. Si consideramos el caso de imágenes médicas siempre existe una regulación y es muy difícil obtenerlas públicamente, incluso data sin etiquetas.

CAPÍTULO 3

PROPUESTA DE SOLUCIÓN

3.1. Descripción General

La propuesta de solución consiste en replicar los resultados de un sistema existente de recuperación de imágenes médicas denominado *Proximity* [Sotomayor *et al.*, 2021], con el objetivo de detectar puntos débiles e implementar otros modelos para mejorar el sistema. Para lograr este objetivo, primero se recreó y evaluó *Proximity* con la misma metodología empleada en el paper. Para evaluación, se utilizó una de las métricas clásicas conocida como “Mean Average Precision” (se detalla en secciones posteriores). Además se evaluaron dos aspectos importantes en recuperación de información: el tiempo de respuesta y las dimensiones de embeddings obtenidos por el modelo. Finalmente, se implementaron distintos modelos de tipo *Self-supervised* y se contrastaron los resultados obtenidos.

3.2. Proximity

Proximity fue un proyecto de FONDEF en el año 2021, donde la idea principal era crear un sistema de recuperación de imágenes médicas para apoyar en la toma de decisiones a los médicos respecto a la presencia de la enfermedad COVID-19 en pacientes. *Proximity* consistía de cuatro baselines y dos propuestas. De los baselines solo se replican dos de ellos que eran los que obtenían los mejores resultados.

Los baselines y las propuestas reconstruidas de *Proximity* fueron las siguientes:

1. Baseline 1: Xception

Consiste en entrenar *Xception* solo con el dataset de clasificación (**SARS CoV2-CT scan**), tomar la última capa, hacer un flatten y dejar el resultado como embedding.

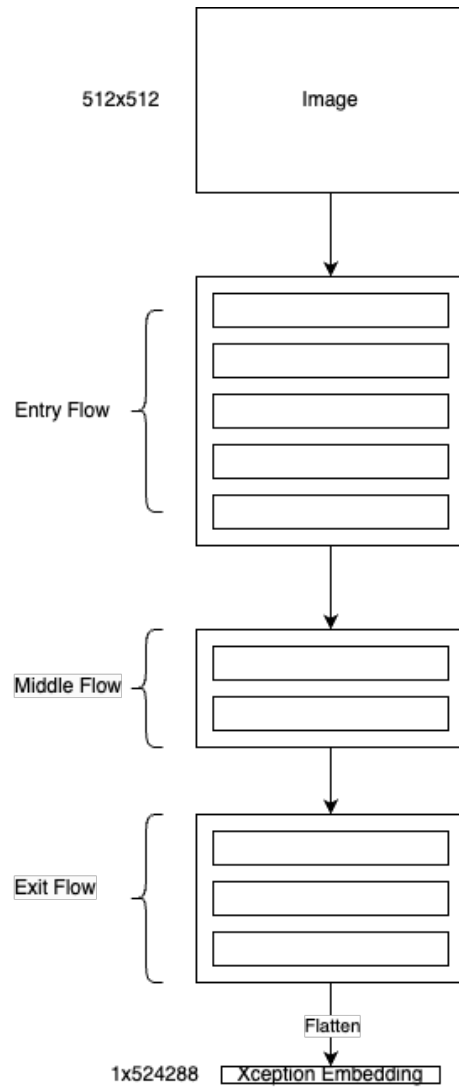


Figura 40: Xception en Proximity.
Fuente: Elaboración propia.

Considerando que la imagen de entrada es de 512x512x3, el embedding resultante es de **524.288** dimensiones.

2. Baseline 2: CE-Net

Consiste en entrenar *CE-Net* [Gu et al., 2019] solo con el dataset de segmentación (**COVID-19 CT**), tomar el output de *RMP block* y hacer un flatten al tensor multidimensional, finalmente el resultado obtenido de esa operación es usado como embedding.

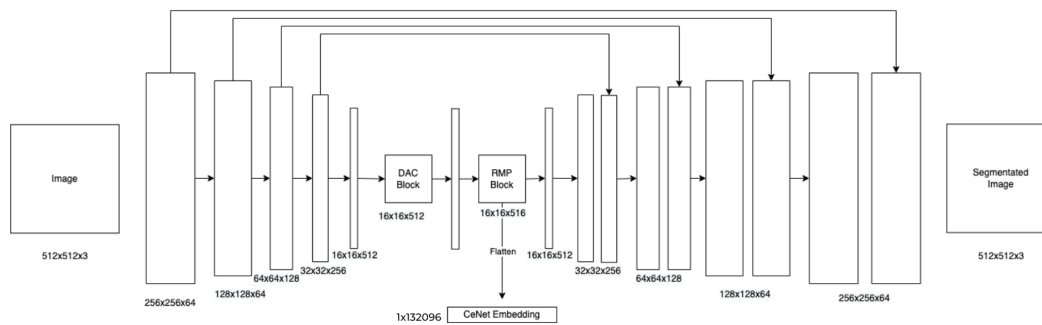


Figura 41: CE-Net en Proximity.

Fuente: Elaboración propia.

Considerando la imagen de entrada que es de 512x512x3, el embedding resultante es de **132.096** dimensiones.

3. Baseline 3: With Concatenation

Consiste en usar el modelo *CE-Net* ya entrenado (baseline 2), tomar el output de la misma capa RMP block para luego realizar un flatten al output obteniendo un vector de tamaño 1x132096, luego se realiza un reshape para dejarlo de la siguiente dimensión 384x344x1, repetir ese vector tres veces para obtener un vector de tres dimensiones, es decir 384x344x3 y finalmente utilizarlo como entrada para la red *Xception*. Finalmente se concatena el embedding del baseline 2 con el resultado de la red *Xception*. En este entrenamiento solo se utiliza el dataset de clasificación (**SARS CoV2-CT scan**).

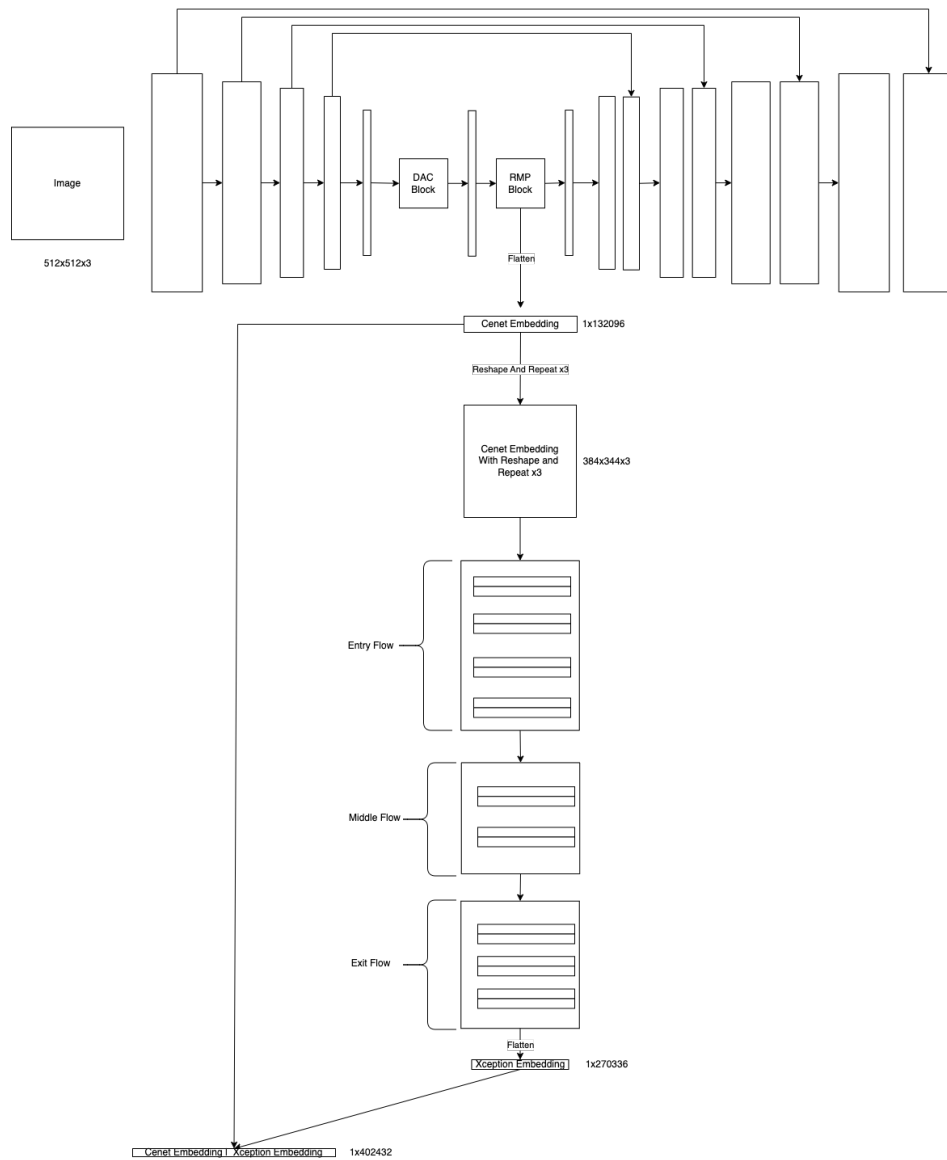


Figura 42: With concatenation en Proximity.

Fuente: Elaboración propia.

Considerando la imagen de entrada que es de 512x512x3, el embedding resultante es de **402.432** dimensiones.

4. Baseline 4: Without Concatenation

Consiste en lo mismo que la anterior, pero la diferencia es que no concatena los embedding, es decir no utiliza el embedding del baseline 2, solo se utiliza el embedding resultante de la red *Xception*. En este entrenamiento solo se utiliza el dataset de clasificación (**SARS CoV2-CT scan**).

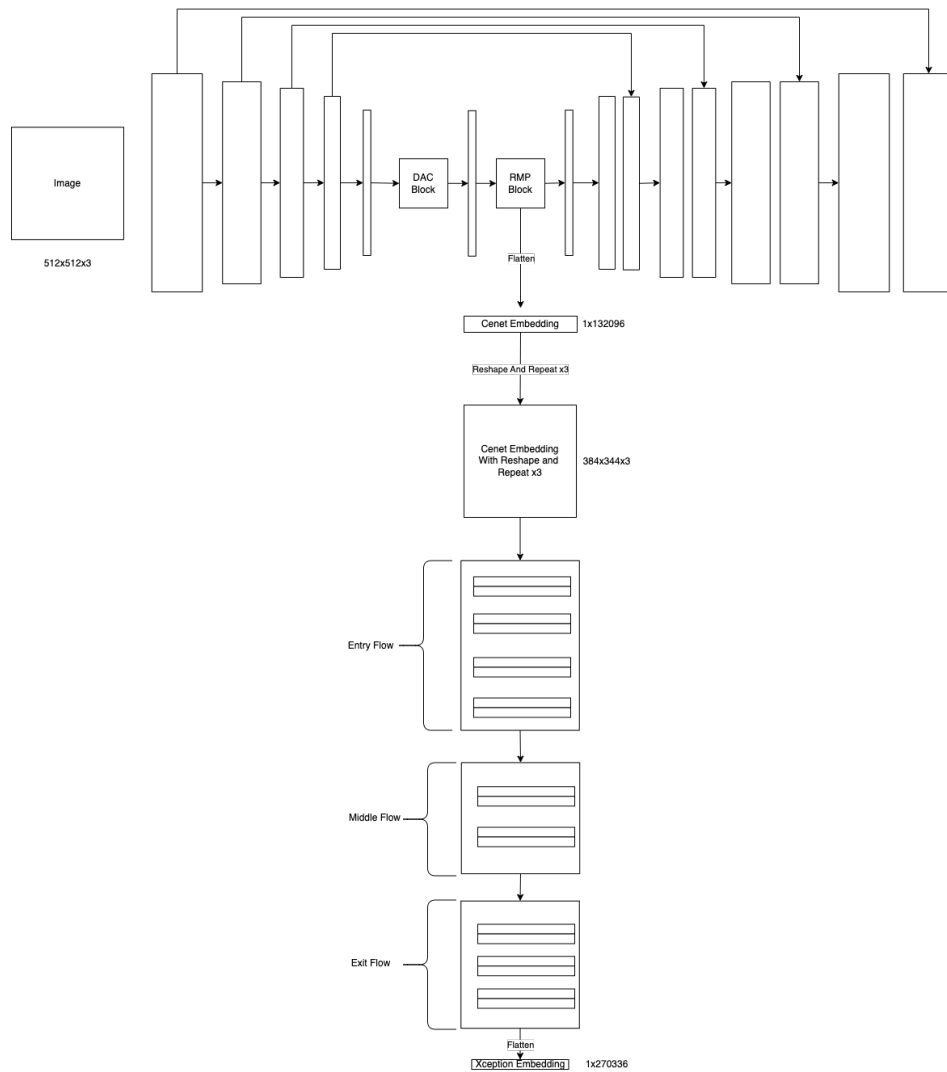


Figura 43: Without concatenation en Proximity.
Fuente: Elaboración propia.

Considerando la imagen de entrada que es de 512x512x3, el embedding resultante es de **270.336** dimensiones.

Para términos de recuperar las imágenes en base a una query (**Query Dataset**) se pasan todas las imágenes de la base de datos (**COVID-CT-Dataset**) por los distintos modelos obteniendo su correspondiente embedding, luego también se pasa la query por el modelo obteniendo el embedding de la query. Finalmente se utiliza la librería MRPT¹, que es la que se encarga de tomar todos los embeddings de la base de datos y recuperar los que son mas cercanos al embedding de la query mediante búsqueda exacta a través de la distancia euclidiana. Al tener las imágenes con su nombre y correspondiente embedding se vuelve sencillo responder directamente con la propia imagen en el resultado final del retrieval.

¹<https://github.com/vioshyvo/mrpt>

3.3. Métodos Self-Supervised

3.3.1. SimCLR

A simple framework for contrastive learning of visual representations (*SimCLR*) [Chen *et al.*, 2020], es un framework que aprende características de imágenes que son discriminativas sin requerir de etiquetas. Estas representaciones posteriormente pueden ser utilizadas en una amplia gama de problemas relacionados con imágenes, como lo es segmentación, clasificación, entre otros. Estableció uno de los patrones más utilizados actualmente en SSL, el cual consiste en maximizar la similaridad entre dos representaciones de vistas distintas de una misma imagen utilizando una función de pérdida de contraste en el espacio latente.

El framework consta de cuatro componentes principales:

1. Una función de aumento de datos estocástica $\mathcal{T}$ que transforma cualquier imagen de entrada de forma aleatoria obteniendo dos vistas correlacionadas entre sí, denominadas $\tilde{x}_i$ y $\tilde{x}_j$ respectivamente. Al ser dos vistas distintas de la misma imagen, eso se considera un par positivo.

Preparing similar pairs in a batch

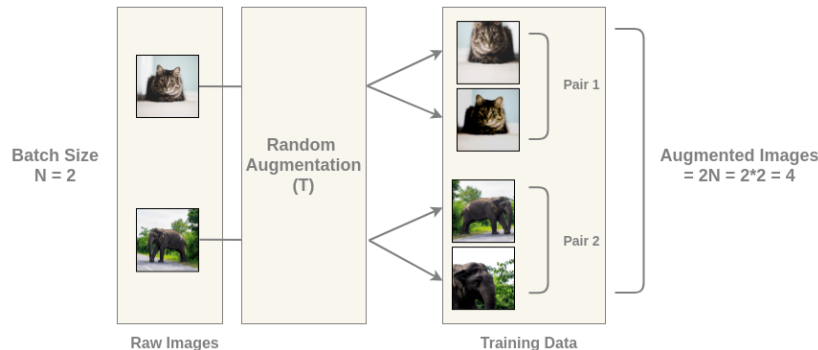


Figura 44: Obtención de pares positivos en SimCLR.

Fuente: Amit Chaudhary.

2. Una red neuronal $f(\cdot)$ que sirva como un encoder base para extraer una representación vectorial de las dos vistas obtenidas con data augmentation. Si bien la red puede ser cualquiera, utilizan *ResNet* por simplicidad.

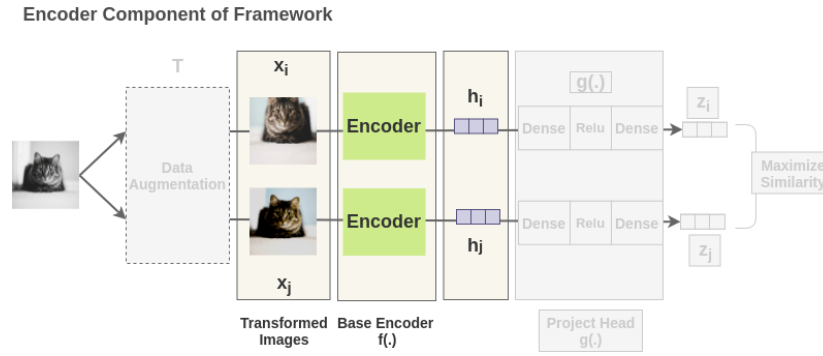


Figura 45: Base encoder en SimCLR.

Fuente: Amit Chaudhary.

- Una pequeña red neuronal llamada projection head $g(\cdot)$, que mapea la representación obtenida por $f(\cdot)$ al espacio donde se aplica la función de pérdida de contraste. Para ello utilizan un bloque MLP con solo una capa oculta (Dense $\rightarrow$ Batch Norm $\rightarrow$ ReLU $\rightarrow$ Dense $\rightarrow$ Batch Norm).

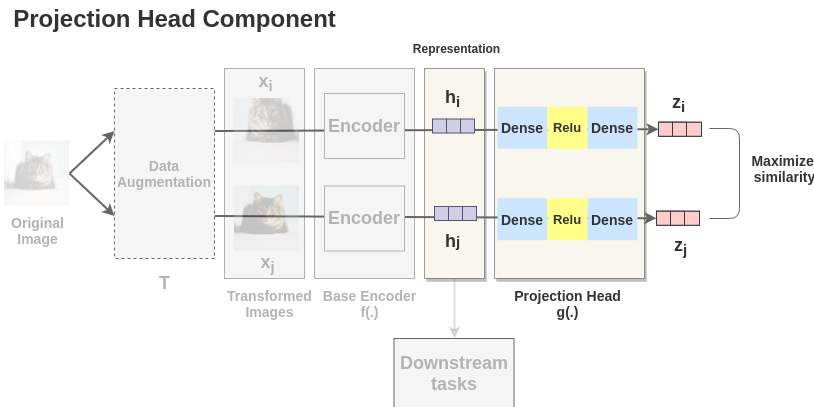


Figura 46: Projection head en SimCLR.

Fuente: Amit Chaudhary.

- Una función de pérdida definida para una tarea de predicción de contraste. Dado un conjunto de imágenes $\{x_k\}$ de tamaño N . Estas imágenes se someten a un proceso de aumento de datos, generando un conjunto $\{\tilde{x}_k\}$ de tamaño $2N$. Dentro de ese conjunto se denomina como $\tilde{x}_i$ y $\tilde{x}_j$ a un par positivo. El objetivo es encontrar $\tilde{x}_j$ dentro de $\{\tilde{x}_k\}$ excluyendo $\tilde{x}_i$.

SimCLR Framework

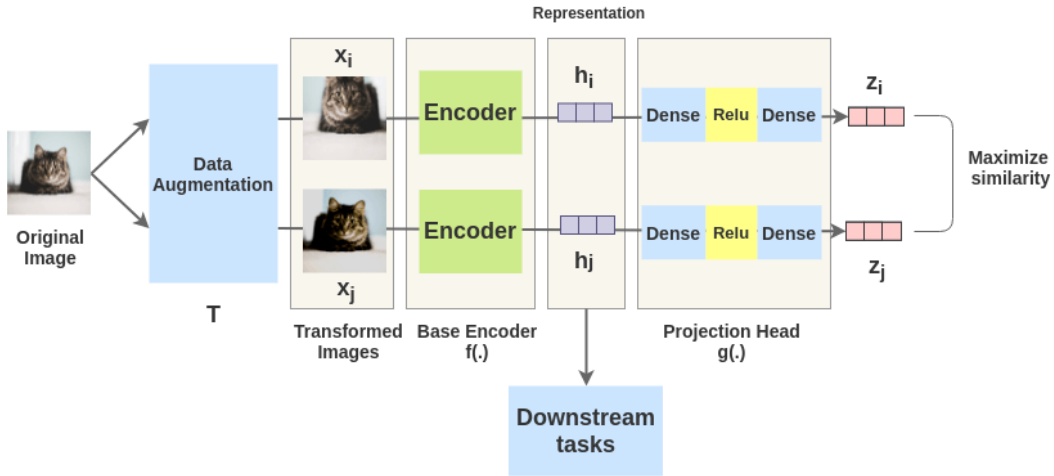


Figura 47: Arquitectura de SimCLR.

Fuente: Amit Chaudhary.

Para definir la tarea de predicción de contraste en pares de imágenes obtenidas vía data augmentation, no utilizan ejemplos negativos de forma explícita, lo que hacen es dado un par positivo, se consideran todas las otras imágenes obtenidas vía data augmentation como un ejemplo negativo.

Entonces, para computar la función de pérdida, lo primero que hacen es calcular lo que denominan *pairwise similarity*. Dada dos imágenes obtenidas vía data augmentation x_i y x_j , calculan la similitud coseno entre las proyecciones obtenidas por $g(\cdot)$, z_i y z_j respectivamente:

$$s_{i,j} = \frac{z_i^T z_j}{\|z_i\| \|z_j\|} \quad (1)$$

Luego se calcula la función de pérdida para un par positivo (i,j) de la siguiente manera:

$$\ell(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)} \quad (2)$$

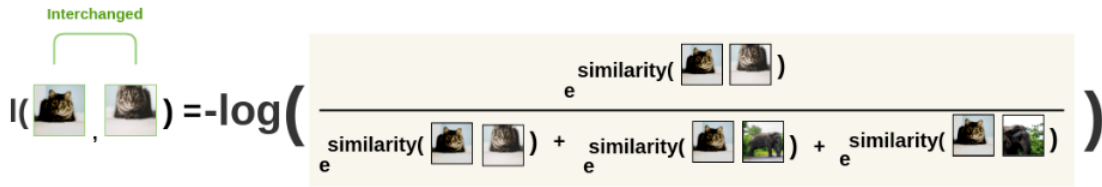
donde:

- $\mathbb{1}_{[k \neq i]} \in \{0, 1\}$ es un indicador que vale uno 1 si $k \neq i$, en otro caso vale cero.
- τ es un parámetro de temperatura.

Posteriormente, también se intercambia la posición en el mismo par positivo y se suma el resultado de $\ell(j, i)$. Finalmente se calcula la función de pérdida a través de todos los pares positivos de la siguiente manera:

$$\mathcal{L} = \frac{1}{2N} \sum_{k=1}^N [\ell(2k-1, 2k) + \ell(2k, 2k-1)] \quad (3)$$

Gráficamente, para el cálculo de la loss $\mathcal{L}$, el intercambio de la posición de las imágenes de un par positivo se vería de la siguiente manera:



$$l(I_1, I_2) = -\log \left(\frac{e^{\text{similarity}(I_1, I_2)}}{e^{\text{similarity}(I_1, I_2)} + e^{\text{similarity}(I_1, I_3)} + e^{\text{similarity}(I_1, I_4)}} \right)$$

Figura 48: Obtención de la loss en SimCLR.

Fuente: Amit Chaudhary.

Basados en esta función de pérdida, la arquitectura mejora a través del tiempo las representaciones obtenidas, dejando imágenes similares cercanas en cierto espacio latente.

Luego de entrenar a la red, se descarta todo menos el encoder $f(\cdot)$, que es de donde se obtiene una representación útil que posteriormente se puede utilizar para alguna downstream task o como fine-tuning.

3.3.2. BYOL

Bootstrap Your Own Latent (BYOL), fue un nuevo enfoque para *Self-supervised* el cual publicaron en el año 2020 [Grill et al., 2020]. La arquitectura sigue el patrón clásico que fue mencionado anteriormente. Utiliza dos redes neuronales de las cuales una se denomina *online* y la otra *target*, las cuales interactúan y aprenden entre ellas, la red denominada *target* actualiza sus pesos utilizando *exponential moving average* de los pesos de la red *online*. Desde una vista aumentada de la imagen de entrada obtenida por medio de *data augmentation*, se entrena a la red que se denomina *online* para predecir la representación de la misma imagen de entrada de la red *online* bajo otra vista aumentada de la imagen de entrada. Una de las características con la que se diferencia del método anterior, es que este método no requiere de *negative sampling*.

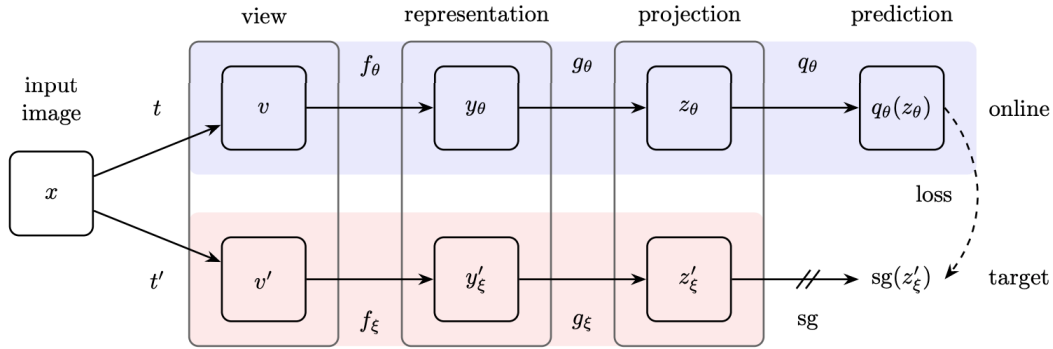


Figura 49: Arquitectura de BYOL.

Fuente: [Grill et al., 2020].

Estructura de ambas redes:

- **Online**: La red *online* es definida por un conjunto de pesos θ y consta de tres etapas: un encoder f_θ , un proyector g_θ y un predictor q_θ .
- **Target**: La red *target* tiene la misma arquitectura que la red *online* con la diferencia que cuenta con pesos ξ . Estos pesos ξ de la red *target* son un EMA de los pesos θ de la red *online* y se calculan de la siguiente manera:

$$\xi \leftarrow \tau \xi + (1 - \tau) \theta \quad (4)$$

donde $\tau \in [0, 1]$ y corresponde a un *target decay rate*.

De esta forma, los pesos de la red *target* no se aprenden, por eso en esta red aparece *sg* (*stop-gradient*). Esto implica que los pesos de la red *target* no son considerados en el cálculo del gradiente, como así lo es con los pesos de la red *online*.

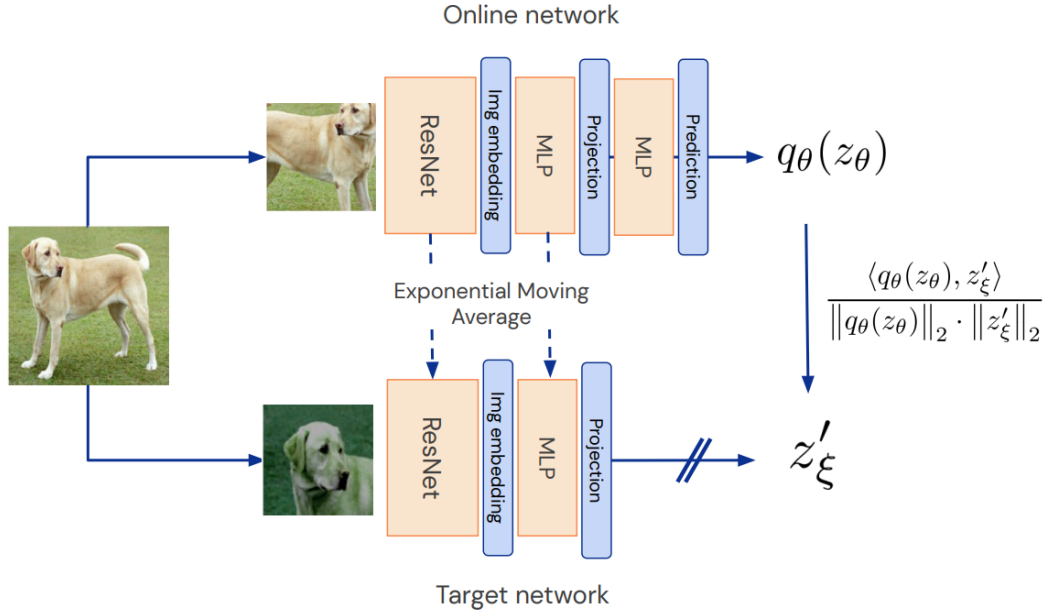


Figura 50: Arquitectura de BYOL con énfasis en la estructura.

Fuente: [Grill *et al.*, 2020].

La arquitectura del encoder f es el mismo para ambas redes al igual que la proyección g , la diferencia entre la red *online* y *target*, que corresponde al módulo de predicción q , junto con el hecho de que la red *target* no se entrena (utiliza sg), es la característica que los autores mencionan que ayuda a que la red no presente el problema llamado *collapse*, que se presenta cuando la red crea exactamente el mismo embeddings para todas las imágenes sin importar si corresponden a categorías diferentes o no.

El encoder f utilizado en este caso corresponde a *ResNet* mientras que g y q corresponden a un bloque MLP. El bloque MLP consta de una capa lineal con output de 4096, seguida de *batch normalization*, ReLU y una capa lineal final con output de 256 dimensiones. La arquitectura completa de *BYOL* con más detalle se vería de la siguiente manera:

La función de pérdida utilizando el modulo de *prediction* que es un bloque MLP, toma la representación de una versión de la imagen y simplemente trata de predecir la representación de la misma imagen bajo el otro punto de vista, ya que ambas deberían ser iguales al ser dos representaciones distintas de la misma imagen, es decir:

$$q_{\theta}(z_{\theta}) = z'_{\xi} \quad (5)$$

Expandiendo un poco la matemática:

$$q_{\theta}(z_{\theta}) = z'_{\xi}$$

$$q_\theta(f_\theta(z)) = f_\xi(z'_\xi)$$

$$q_\theta(f_\theta(A(x))) = f_\xi(A'(x))$$

Donde A representa una augmentation y x representa la imagen de entrada. Entonces lo que tiene que hacer q es predecir el valor esperado de la representación de la imagen de entrada bajo todas las diferentes posibilidades de augmentation y es por esto que la red aprende a ignorar los diferentes tipos de augmentation. Lo que se intenta conseguir con esto es que la red destruya toda la información no semántica o bien que no aporte valor a la imagen, al obtener una representación que sea invariante a los diferentes tipos de augmentation.

Formalmente la función de pérdida de BYOL corresponde a:

$$\mathcal{L}_{\theta,\xi} \triangleq \|\overline{q_\theta}(z_\theta) - \overline{z'_\xi}\|_2^2 = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{\|q_\theta(z_\theta)\|_2 \cdot \|z'_\xi\|_2} \quad (6)$$

Donde $\overline{x}$ significa normalizar el vector x utilizando la norma ℓ_2 , es decir:

$$\overline{q_\theta}(z_\theta) \triangleq \frac{q_\theta(z_\theta)}{\|q_\theta(z_\theta)\|_2}$$

$$\overline{z'_\xi} \triangleq \frac{z'_\xi}{\|z'_\xi\|_2}$$

Luego de entrenar a la red, se descarta todo menos el encoder f_θ , que es de donde se obtiene esta representación útil que posteriormente se puede utilizar para alguna *downstream task* o como *fine-tuning*.

3.3.3. SiamSiam

Exploring Simple Siamese Representation Learning (*SiamSiam*) [Chen y He, 2021] es un método que si bien sigue el mismo patrón de maximizar la similaridad entre dos vistas obtenidas de una misma imagen por medio de augmentation, se caracteriza por no utilizar lo siguiente:

- Pares de ejemplos negativos (*SimCLR*).
- Batch grandes (*SimCLR*).

- Momentum encoder (BYOL).

Para no requerir de lo anterior una clave de ello es su arquitectura:

- Toma como input x_1 y x_2 que son dos vistas extraídas de una misma imagen x por medio de aumento de datos estocástico.
- Las dos vistas son procesadas por una red de tipo encoder f que cuenta de un backbone como lo es *ResNet* más una projection MLP head.
- El encoder f comparte pesos entre las dos vistas x_1 y x_2 .
- Un prediction MLP head h que transforma la salida de una vista y la iguala a la otra vista.

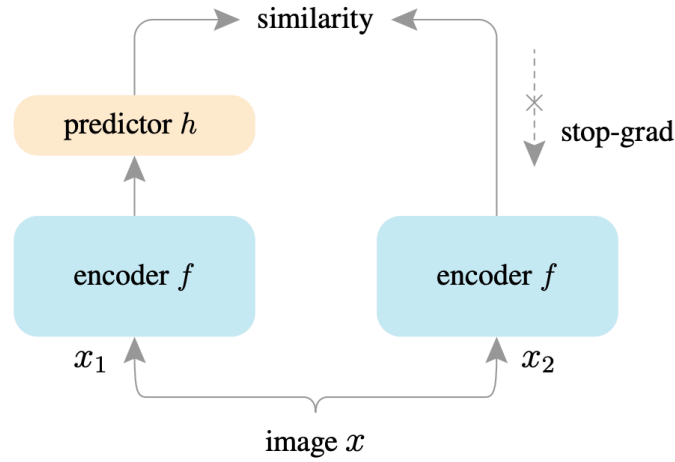


Figura 51: Arquitectura de SiamSiam.

Fuente: [Chen y He, 2021].

Considerando los dos vectores de salida $p_1 \triangleq h(f(x_1))$ y $z_2 \triangleq f(x_2)$, se minimiza el negativo de la similitud coseno:

$$\mathcal{D}(p_1, z_2) = -\frac{p_1}{\|p_1\|_2} \cdot \frac{z_2}{\|z_2\|_2} \quad (7)$$

Finalmente al igual que *SimCLR* dejan simétrica la función de pérdida:

$$\mathcal{L} = \frac{1}{2}\mathcal{D}(p_1, z_2) + \frac{1}{2}\mathcal{D}(p_2, z_1) \quad (8)$$

Un componente importante para que esto funcione es el operador *stop-gradient* (stopgrad) que se aplica a z en la función de pérdida:

$$\mathcal{L} = \frac{1}{2}\mathcal{D}(p_1, \text{stopgrad}(z_2)) + \frac{1}{2}\mathcal{D}(p_2, \text{stopgrad}(z_1)) \quad (9)$$

Esto significa que el encoder en x_2 no recibe gradientes de z_2 en el primer término de la ecuación, pero si recibe gradientes de p_2 en el segundo término y viceversa para el caso de x_1 .

También demuestran experimentalmente la importancia del operador *stop-gradient* en el aprendizaje de la red.

Luego de entrenar a la red, se descarta todo menos el backbone de f , que es de donde se obtiene esta representación útil que posteriormente se puede utilizar para alguna downstream task o como fine-tuning.

3.3.4. Barlow Twins

Self-Supervised Learning via Redundancy Reduction (*Barlow Twins*) [Zbontar *et al.*, 2021], es otro enfoque para SSL un poco diferente al anterior al contar con una arquitectura que es simétrica. Se propone una función objetivo que naturalmente evita el problema del *collapse* computando la matriz de cross-correlación entre los embeddings de dos redes neuronales idénticas alimentadas por diferentes distorsiones de la imagen de entrada, haciendo que esta sea lo mas cercano posible a la matriz identidad. Esto causa que los embeddings de diferentes distorsiones de una imagen sean similares entre si, mientras se minimiza la redundancia entre los componentes de esos vectores.

Barlow Twins obtuvo su nombre del neurocientífico H. Barlow [1961] quien en su trabajo emplea el término de *redundancy-reduction*, el cual inspira a este método al ser aplicado este principio en un par idéntico de redes neuronales. Este tipo de arquitectura se caracteriza por no requerir un tamaño de batch grande o presentar asimetría entre las dos redes, tampoco requiere de *stop-gradient* ni de EMA.

Este método es intuitivo y simple de implementar, es similar a otros métodos de SSL en el hecho de contar con el objetivo de aprender representaciones o bien embeddings que sean invariantes a las distintas distorsiones que uno le pueda aplicar a la imagen de entrada. Para cumplir con dicho objetivo en *Barlow Twins* se realizan tres etapas:

1. Se aplica *data augmentation* a un batch de imágenes de entrada X obteniendo batches de dos versiones distorsionadas distintas de cada imagen Y^A e Y^B .

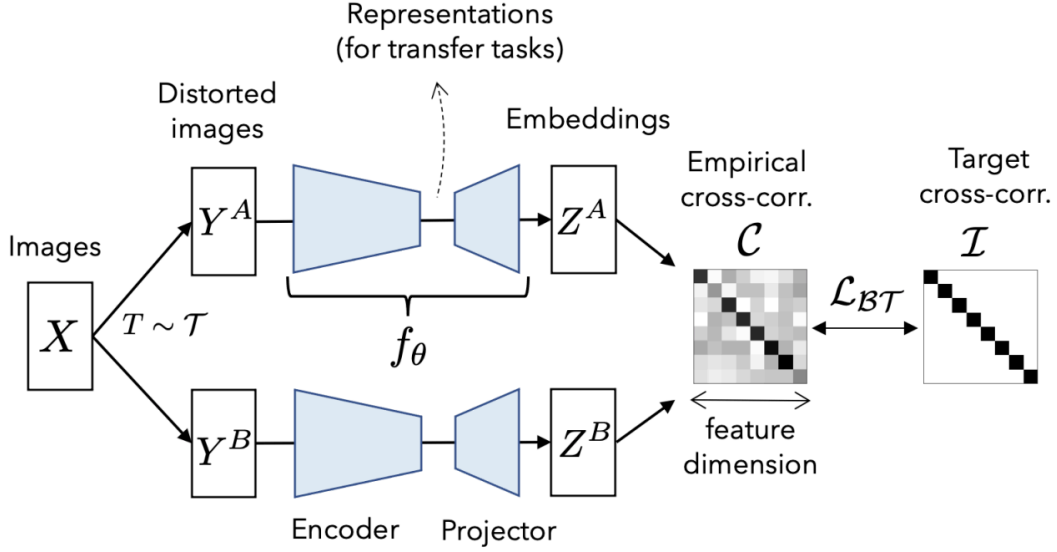


Figura 52: Arquitectura de Barlow Twins
Fuente: [Zbontar et al., 2021].

2. Los dos batches de las dos versiones distorsionadas de cada imagen de entrada (Y^A e Y^B) se pasan a través de un mismo encoder f_θ , generando dos batches de embedding Z^A y Z^B respectivamente. A ambos batches se les sustrae la media en base a la dimensión del batch.
3. Se calcula la matriz de correlación cruzada $\mathcal{C}$ entre ambos batches de embeddings y se aplica una loss que computa que tan lejos se encuentra la matriz $\mathcal{C}$ de la matriz de identidad $\mathcal{I}$.

Formalmente la función de pérdida de *Barlow Twins* corresponde a:

$$\mathcal{L}_{BT} \triangleq \sum_i (1 - C_{ii})^2 + \lambda \sum_i \sum_{j \neq i} C_{ij}^2 \quad (10)$$

Donde:

$$C_{ij} \triangleq \frac{\sum_b z_{b,i}^A z_{b,j}^B}{\sqrt{\sum_b (z_{b,i}^A)^2} \sqrt{\sum_b (z_{b,j}^B)^2}}$$

El primer término de la función de pérdida se conoce como *invariance term* y el segundo se conoce como *redundance reduction term*, λ es una constante positiva que se multiplica al término de *redundance reduction term* y funciona como trade off de la importancia del primer

y segundo término en la función de pérdida (En el paper mencionan que un valor de 0.005 provee buenos resultados). $\mathcal{C}$ es la matriz *cross-correlation* que se calcula considerando la salida de las dos redes neuronales idénticas a través de la dimensión del batch, b indexa las imágenes del batch e i, j indexan las dimensiones del vector de salida de las dos redes que son idénticas.

El *invariance term* se relaciona con la diagonal de la matriz $\mathcal{C}$. Como los elementos de la diagonal miden la correlación entre embeddings de la misma imagen, lo que queremos es que tengan una correlación perfecta igual a 1. Esta correlación haría que los embeddings sean invariante a distorsiones en la imagen, lo que significa que embeddings de la misma imagen aunque sean un poco diferente o tengan alguna modificación se encuentren en el mismo espacio.

Por otra parte *redundance reduction term* se relaciona con los elementos que no están en la diagonal $\mathcal{C}_{ij}$. Estos elementos miden la correlación entre diferentes embeddings. Si minimizamos esos elementos a cero, la salida de la red haría que esos elementos no tengan correlación. Mediante esto ultimo se consigue que la salida de la red no contenga información que sea redundante en las imágenes.

Luego de entrenar a la red, se descarta todo menos el backbone de f_θ , que es de donde se obtiene esta representación útil que posteriormente se puede utilizar para alguna downstream task o como fine-tuning.

CAPÍTULO 4

EXPERIMENTOS

4.1. DATASETS

- **SARS CoV2-CT scan:** dataset de COVID-19 de libre acceso². El dataset se compone de un total de 2482 imágenes CT, de las cuales 1252 corresponden a pacientes infectados por SARS-CoV-2 (COVID-19) y 1230 corresponden a pacientes que no presentan este virus, pero presentaban otras enfermedades pulmonares. La data fue obtenida desde hospitales de Sao Paulo, Brazil [Soares *et al.*, 2020].

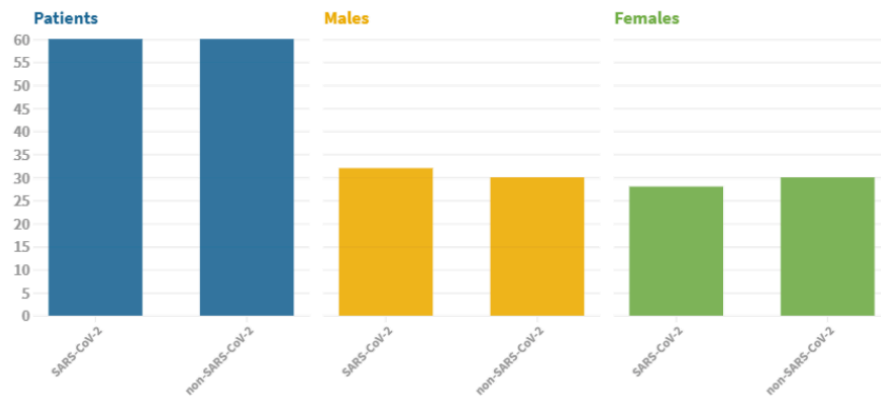


Figura 53: Proporción de géneros y cantidad de pacientes (SARS CoV2-CT scan).
Fuente: [Soares *et al.*, 2020].

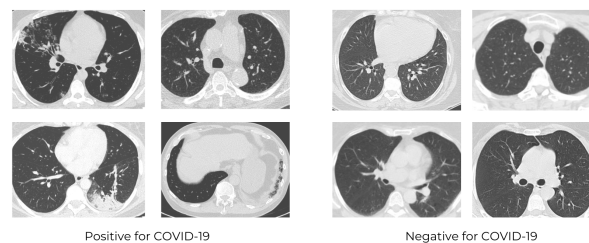


Figura 54: Ejemplos de imágenes con y sin COVID-19 (SARS CoV2-CT scan).
Fuente: Elaboración propia.

²<https://www.kaggle.com/datasets/plameneduardo/sarscov2-ctscan-dataset>

- **COVID-19 CT:** el dataset contiene imágenes de tipo CT con su correspondiente máscara de segmentación. En el paper Content-Based Medical Image Retrieval [Sotomayor *et al.*, 2021], el autor menciona que el dataset se encuentra disponible públicamente³, pero se encuentra en formato raw. Es por ello que el dataset ya procesado, es proporcionado finalmente por el mismo autor del paper. El dataset consta de un total de 2316 imágenes de tipo CT, cada una de ellas con su correspondiente máscara. El dataset listo para ser utilizado en una red neuronal de segmentación puede encontrarse directamente en el siguiente enlace⁴.

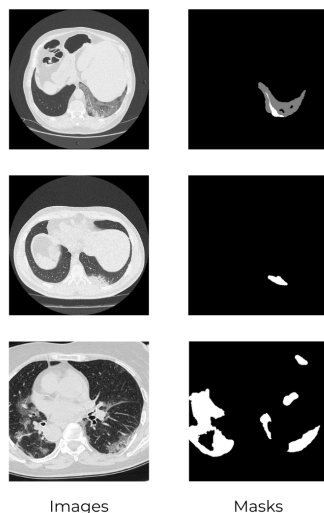


Figura 55: Ejemplos de imágenes con y sin COVID-19 (COVID-19 CT).

Fuente: Elaboración propia.

- **COVID-CT-Dataset:** también es un dataset de COVID-19 de libre acceso⁵. Este dataset consiste en un conjunto de 349 imágenes de cortes axiales pulmonares con COVID-19, las cuales son obtenidas de 216 pacientes y también cuenta con un conjunto de 463 imágenes de cortes axiales pulmonares sin COVID-19. La mayoría de las imágenes del conjunto de entrenamiento son obtenidas de papers relacionados con COVID-19 desde medRxiv⁶ y bioRxiv⁷. Las imágenes asociadas a cada paper se obtienen usando métodos de scrapping y luego se etiquetan manualmente revisando la descripción de la imagen. Además, para cada caso se guarda la metadata asociada al paciente, como lo es la edad, género, fecha, ubicación e historial médico. Este procedimiento genera un dataset bastante heterogéneo, donde el mínimo, promedio y máximo de altura para las imágenes corresponde a 153, 491 y 1853 respectivamente, y el mínimo, promedio y máximo para el ancho corresponde a 124, 383 y 1485 respectivamente. Para el

³<http://medicalsegmentation.com/covid19/>

⁴https://drive.google.com/drive/folders/1U9KL0dBzlNITgUqHmGh1FLz0sqHLWtWr?usp=share_link

⁵<https://github.com/UCSD-AI4H/COVID-CT>

⁶<https://www.medrxiv.org/>

⁷<https://www.biorxiv.org/>

conjunto de validación y pruebas, se utilizan únicamente imágenes originales donadas por hospitales [Zhao *et al.*, 2020].

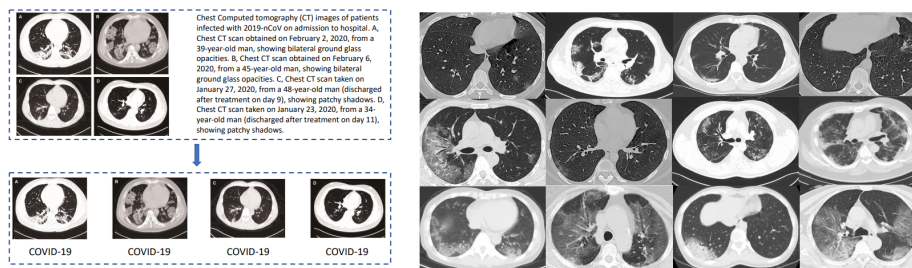


Figura 56: Proceso de obtención de imágenes CT (COVID-CT Dataset).
Fuente: [Zhao *et al.*, 2020].

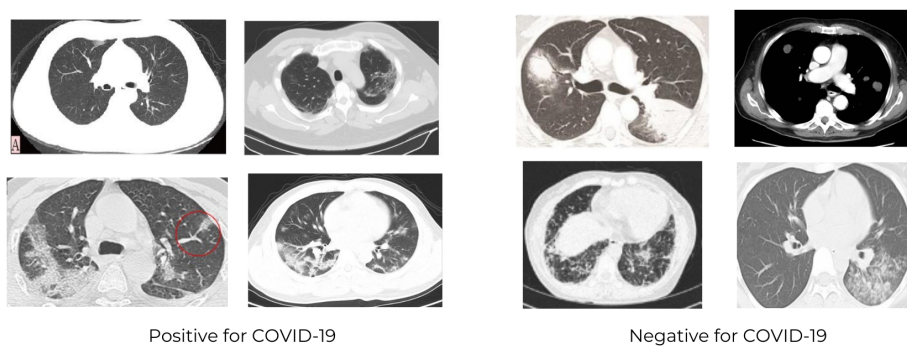


Figura 57: Ejemplos de imágenes con y sin COVID-19 (COVID-CT Dataset).
Fuente: Elaboración propia.

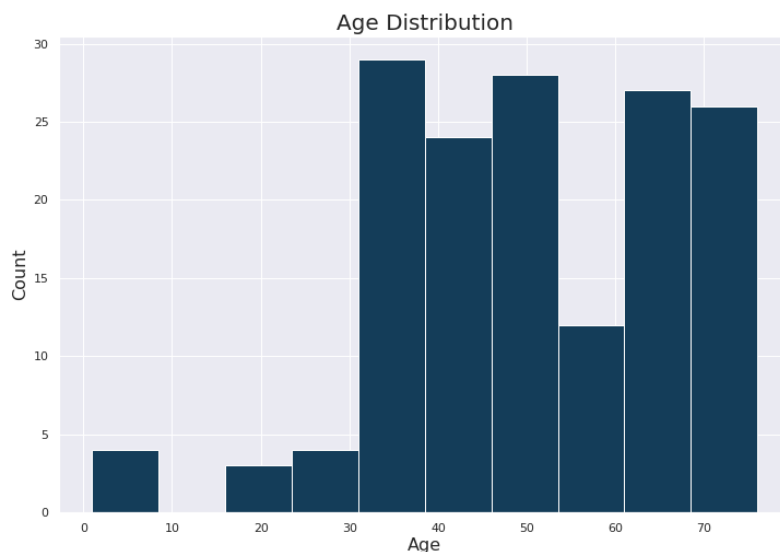


Figura 58: Distribución de la edad para pacientes con COVID-19 (COVID-CT Dataset).
Fuente: Elaboración propia.

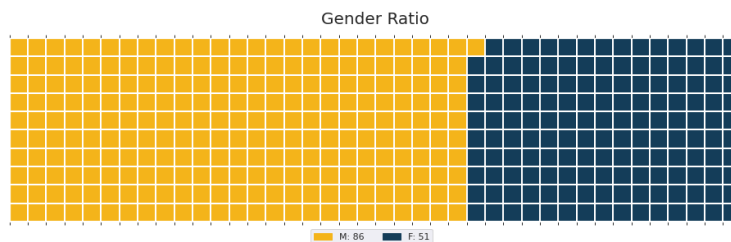


Figura 59: Proporción de géneros para pacientes con COVID-19 (COVID-CT Dataset).
Fuente: Elaboración propia.

- Query Dataset: dataset con un total de 50 imágenes relacionadas con la presencia de COVID-19, 25 de la imágenes son pacientes que sí presentan COVID-19 y el otro 25 son pacientes que no presentan COVID-19. También es un dataset privado proporcionado por el equipo de médicos.

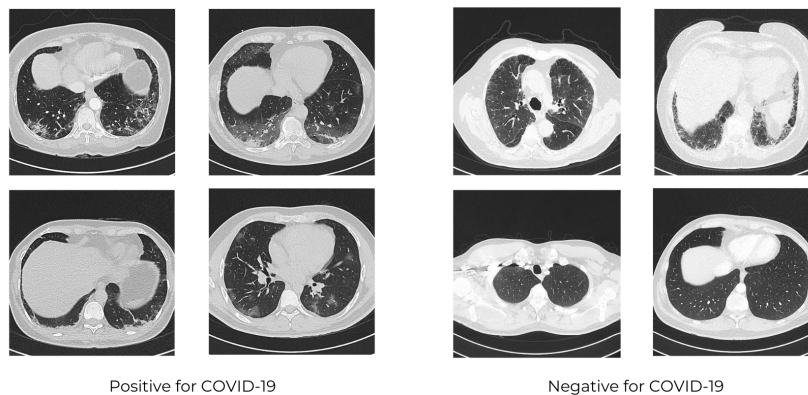


Figura 60: Ejemplos de imágenes con y sin COVID-19 (Query Dataset).
Fuente: Elaboración propia.

Resumen de los dataset utilizados:

Dataset	Imágenes	Tipo	Uso	Peso
SARS CoV2-CT scan	- Covid: 1,252 - No Covid: 1,229	Clasificación	Training	231 MB
COVID-19 CT	- Images: 2,316 - Masks: 2,316	Segmentación	Training	10.3 GB
Covid-CT-Dataset	- Covid: 349 - No Covid: 397	Clasificación	Testing	92.6 MB
Query Dataset	- Covid: 25 - No Covid: 25	Clasificación	Query	12.7 MB

Figura 61: Resumen de los dataset que contienen imágenes médicas.
Fuente: Elaboración propia.

4.2. MÉTRICAS

Como se definió anteriormente, típicamente un problema de recuperación de información se representa mediante una “query” (q), que simboliza lo que el usuario está buscando dentro de la base de datos, por lo que una medida clásica para evaluar un sistema de tipo CBIR es

la precisión que se define como:

$$P@k(q) = \frac{\text{No. relevant images retrieved to } k}{\text{Total No. images retrieved to } k} \quad (11)$$



Figura 62: Ejemplo para calculo de la precisión en un sistema CBIR.

Fuente: Elaboración propia.

Al utilizar la fórmula descrita anteriormente sobre el ejemplo de la figura 62, en el cual solo dos imágenes son relevantes, obtenemos:

- $P@1 = 1/1 = 1$
- $P@2 = 1/2 = 0.50$
- $P@3 = 2/3 = 0.66$
- $P@n = 2/n$

Otra medida que deriva de la precisión es la AP (Average Precision) que se define como:

$$AP(q) = \frac{\sum_{k=1}^n P@k(q) \cdot l_k}{\text{Total No. relevant images}} \quad (12)$$

donde l_k es una variable binaria que vale 1 si la imagen es relevante en la posición k o cero en caso contrario.

También existe la métrica mAP (Mean Average Precision) que a diferencia del anterior calcula un promedio de AP considerando todas las N query que realiza un usuario y se define como:

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (13)$$

4.3. RESULTADOS

Los siguientes resultados son los obtenidos por Proximity considerando los dataset con imágenes médicas. Para evaluar los modelos se evalúan principalmente 3 puntos clave en la recuperación de imágenes:

- Mean Average Precision (mAP@k).
- Tiempo de recuperación de imágenes.
- Tamaño de los embeddings.

4.3.1. Proximity

Para el cálculo de la métrica $mAP@k$ *CeNet* se entrena con el dataset *COVID-19 CT* y los otros tres modelos utilizan el dataset *SARS CoV2-CT scan*. Para testing todos los modelos utilizan el *Query Dataset* como input (query) y utilizan el *Covid-CT-Dataset* como base de datos para obtener los resultados.

Resultados considerando todas las imágenes query:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity							
Xception	0,456	0,611	0,585	0,554	0,543	0,535	0,530
CeNet	0,420	0,587	0,563	0,530	0,524	0,524	0,523
With Concat	0,420	0,586	0,563	0,530	0,524	0,524	0,523
Without Concat	0,472	0,602	0,563	0,539	0,530	0,525	0,520

Tabla 1: Proximity: $mAP@k$ all images (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity							
Xception	0,055	0,062	0,042	0,033	0,030	0,029	0,027
CeNet	0,057	0,069	0,056	0,061	0,050	0,042	0,039
With Concat	0,057	0,069	0,056	0,061	0,050	0,042	0,039
Without Concat	0,100	0,097	0,081	0,057	0,044	0,036	0,030

Tabla 2: Proximity: $mAP@k$ all images (stdev between five different seeds).

Fuente: Elaboración Propia.

Aunque *Without Concat* muestra un rendimiento destacado al considerar únicamente una imagen como objetivo de retrieval, a media que aumenta la cantidad de imágenes requeridas, *Xception* logra posicionarse como el mejor modelo. Es destacable que, siendo un modelo clásico y menos complejo que *With Concat* o *Without Concat*, debido a que no depende del segmentador *CeNet*, sea un modelo que logra obtener resultados superiores, incluso contando con una desviación estándar menor que los otros tres modelos.

Resultados considerando solo imágenes query que contienen COVID-19:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity							
Xception	0,408	0,563	0,529	0,485	0,463	0,447	0,436
CeNet	0,416	0,611	0,578	0,538	0,517	0,506	0,496
With Concat	0,416	0,609	0,577	0,537	0,517	0,506	0,496
Without Concat	0,480	0,612	0,562	0,515	0,496	0,485	0,477

Tabla 3: Proximity: mAP@k images with covid (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity							
Xception	0,131	0,127	0,106	0,107	0,108	0,110	0,109
CeNet	0,178	0,121	0,112	0,101	0,091	0,077	0,066
With Concat	0,178	0,123	0,113	0,102	0,091	0,078	0,066
Without Concat	0,155	0,084	0,042	0,038	0,037	0,036	0,038

Tabla 4: Proximity: mAP@k images with covid (stdev between five different seeds).

Fuente: Elaboración Propia.

Contrario al caso anterior, los mejores resultados son obtenidos por parte de *CeNet*. En algunos casos, es superado o igualado por *With Concat* y *Without Concat*, lo cual tiene sentido debido a que estos dos últimos utilizan el embedding obtenido por *CeNet* para la recuperación de las imágenes. La intuición respecto al buen desempeño por parte de *CeNet* se debe a que es una red especializada en segmentación de imágenes y, por ende, en detección de anomalías en la imagen. Esta información puede que sea de gran ayuda al momento de discernir si una imagen presenta la enfermedad de COVID-19 o no.

Resultados considerando solo imágenes query que no contienen COVID-19:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity							
Xception	0,504	0,659	0,641	0,622	0,623	0,623	0,624
CeNet	0,424	0,563	0,548	0,522	0,531	0,542	0,551
With Concat	0,424	0,563	0,548	0,522	0,531	0,542	0,551
Without Concat	0,464	0,593	0,565	0,563	0,563	0,564	0,564

Tabla 5: Proximity: mAP@k images without covid (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity							
Xception	0,112	0,089	0,080	0,076	0,072	0,071	0,071
CeNet	0,244	0,198	0,172	0,180	0,147	0,119	0,096
With Concat	0,244	0,199	0,172	0,181	0,148	0,119	0,097
Without Concat	0,193	0,211	0,184	0,140	0,107	0,089	0,077

Tabla 6: Proximity: mAP@k images without covid (stdev between five different seeds).

Fuente: Elaboración Propia.

En este último caso, *Xception* logra estar a la delantera en todos los casos, superando completamente a *CeNet*, con una menor desviación estándar que los otros tres modelos. La intuición respecto al bajo desempeño de *CeNet* para este caso es similar al anterior. Debido a que es una red especializada en segmentación, las imágenes que no presentan la enfermedad de COVID-19 pueden obtener un embedding que no sigue un patrón entre imágenes del mismo tipo (sin COVID-19) y tampoco están relacionadas entre sí en cierto espacio, dificultando así la tarea del retrieval.

Para el cálculo de los tiempos de recuperación de imágenes, estos se obtienen considerando todo el Query dataset, es decir, se procesan cada una de las 50 imágenes query y se calcula el tiempo en segundos que le toma al modelo recuperar las imágenes, finalmente, se extrae la media de estos tiempos.

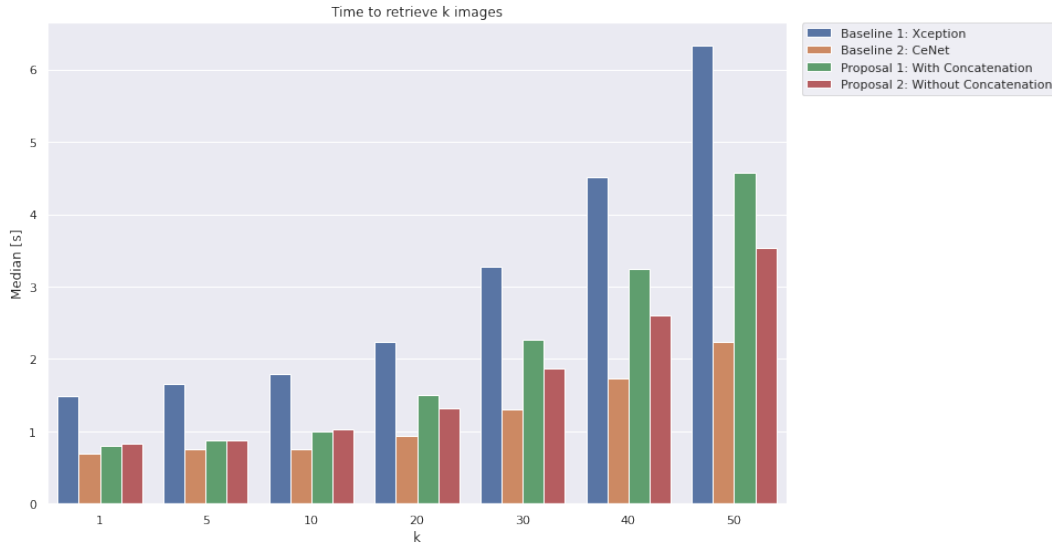


Figura 63: Proximity: Median in seconds to retrieve k images across the whole query dataset.

Fuente: Elaboración propia.

	k=1	k=5	k=10	k=20	k=30	k=40	k=50
Proximity							
Xception	1.489	1.659	1.788	2.242	3.275	4.506	6.329
CeNet	0.696	0.753	0.752	0.931	1.304	1.730	2.237
With Concat	0.805	0.868	1.004	1.501	2.263	3.241	4.575
Without Concat	0.834	0.876	1.023	1.317	1.862	2.603	3.542

Tabla 7: Proximity: Median in seconds to retrieve k images across the whole query dataset.

Fuente: Elaboración Propia.

Si bien, al principio se obtienen resultados considerables, es decir, tiempos menores a un segundo. Cuando se intenta hacer una recuperación con mayor cantidad de imágenes, se aprecia un comportamiento exponencial, que llega a seis segundos aproximadamente para el mejor modelo con mAP@k cuando se requiere recuperar cincuenta imágenes. Considerando que la base de datos solo cuenta con alrededor de 700 imágenes, es probable que el modelo en un ambiente clínico real con un mayor número de imágenes no sea escalable.

Para el cálculo del tamaño de los embeddings, estos se obtienen directamente del modelo utilizado, y además se calcula el tamaño en bytes de la carpeta resultante al transformar toda la base de datos a embeddings mediante los modelos.

	Dimensions	Size
Proximity		
Xception	524,288	1.5G
CeNet	132,096	376.0M
With Concat	402,432	1.1G
Without Concat	270,336	769.4M

Tabla 8: Proximity: Embeddings dimension and size of every method.

Fuente: Elaboración Propia.

Probablemente, este sea uno de los puntos débiles del modelo, al no controlar o tener en consideración los inputs pasados por la red o los embeddings utilizados, se terminan obteniendo embeddings muy grandes que impactan directamente el tiempo de recuperación de las imágenes y al hardware necesario para implementar el modelo. Si consideramos que solo había 700 imágenes en la base de datos, el modelo utiliza 769.4 MB, y si tuviéramos 300.000 imágenes se necesitarían aproximadamente 330 GB de espacio en el disco duro.

También se realiza un pequeño experimento para evaluar que pasa con la métrica mAP@k a medida que vamos utilizando menos etiquetas del dataset.

Resultados considerando todas las imágenes query:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Xception	0,456	0,611	0,585	0,554	0,543	0,535	0,530
CeNet	0,420	0,587	0,563	0,530	0,524	0,524	0,523
With Concat	0,420	0,586	0,563	0,530	0,524	0,524	0,523
Without Concat	0,472	0,602	0,563	0,539	0,530	0,525	0,520
Proximity 70 %							
Xception	0,488	0,622	0,577	0,544	0,532	0,523	0,517
CeNet	0,460	0,590	0,564	0,519	0,512	0,512	0,510
With Concat	0,456	0,590	0,563	0,519	0,512	0,512	0,510
Without Concat	0,500	0,645	0,603	0,562	0,547	0,539	0,534
Proximity 50 %							
Xception	0,516	0,653	0,625	0,589	0,570	0,559	0,551
CeNet	0,520	0,630	0,604	0,575	0,560	0,552	0,547
With Concat	0,520	0,630	0,604	0,575	0,560	0,552	0,547
Without Concat	0,480	0,603	0,576	0,548	0,536	0,528	0,525
Proximity 30 %							
Xception	0,524	0,654	0,624	0,585	0,563	0,552	0,545
CeNet	0,500	0,624	0,584	0,551	0,541	0,535	0,532
With Concat	0,502	0,624	0,584	0,551	0,541	0,535	0,532
Without Concat	0,432	0,587	0,581	0,557	0,541	0,533	0,529
Proximity 10 %							
Xception	0,504	0,659	0,616	0,579	0,562	0,552	0,544
CeNet	0,396	0,563	0,540	0,527	0,519	0,513	0,510
With Concat	0,392	0,563	0,540	0,527	0,518	0,514	0,510
Without Concat	0,456	0,628	0,583	0,548	0,537	0,530	0,525

Tabla 9: Proximity Experiment: mAP@k all images (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Xception	0,055	0,062	0,042	0,033	0,030	0,029	0,027
CeNet	0,057	0,069	0,056	0,061	0,050	0,042	0,039
With Concat	0,057	0,069	0,056	0,061	0,050	0,042	0,039
Without Concat	0,100	0,097	0,081	0,057	0,044	0,036	0,030
Proximity 70 %							
Xception	0,100	0,058	0,035	0,020	0,016	0,016	0,015
CeNet	0,037	0,049	0,026	0,025	0,022	0,016	0,015
With Concat	0,043	0,049	0,026	0,025	0,022	0,016	0,015
Without Concat	0,058	0,035	0,032	0,020	0,016	0,014	0,011
Proximity 50 %							
Xception	0,102	0,058	0,043	0,038	0,037	0,036	0,034
CeNet	0,087	0,130	0,105	0,079	0,066	0,053	0,046
With Concat	0,087	0,130	0,106	0,079	0,066	0,053	0,046
Without Concat	0,069	0,096	0,084	0,076	0,068	0,061	0,051
Proximity 30 %							
Xception	0,079	0,051	0,049	0,045	0,040	0,037	0,035
CeNet	0,055	0,055	0,042	0,041	0,041	0,039	0,037
With Concat	0,051	0,055	0,043	0,041	0,041	0,039	0,037
Without Concat	0,018	0,020	0,023	0,025	0,026	0,026	0,026
Proximity 10 %							
Xception	0,055	0,022	0,023	0,019	0,016	0,015	0,013
CeNet	0,121	0,061	0,049	0,032	0,025	0,020	0,017
With Concat	0,118	0,061	0,049	0,032	0,025	0,020	0,017
Without Concat	0,097	0,058	0,049	0,038	0,034	0,030	0,027

Tabla 10: Proximity Experiment: mAP@k all images (stdev between five different seeds).

Fuente: Elaboración Propia.

Contrario a lo esperado, se aprecia como *CeNet* obtiene mejores resultados a medida que tiene menos acceso a data etiquetada, y el modelo *With Concat* se aprovecha mayormente del embedding obtenido por *CeNet* para mejorar el resultado en la recuperación de imágenes. En cambio, a medida que se utiliza más data etiquetada *Xception* supera a *CeNet* en los casos extremos donde se utiliza toda la data, y en ese momento, el modelo *Without Concat* logra superar por muy poca diferencia a *Xception*.

Resultados considerando solo imágenes query que contienen COVID-19:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Xception	0,408	0,563	0,529	0,485	0,463	0,447	0,436
CeNet	0,416	0,611	0,578	0,538	0,517	0,506	0,496
With Concat	0,416	0,609	0,577	0,537	0,517	0,506	0,496
Without Concat	0,480	0,612	0,562	0,515	0,496	0,485	0,477
Proximity 70 %							
Xception	0,376	0,536	0,497	0,455	0,438	0,427	0,418
CeNet	0,520	0,662	0,623	0,566	0,541	0,525	0,510
With Concat	0,512	0,660	0,621	0,565	0,541	0,524	0,510
Without Concat	0,456	0,628	0,580	0,533	0,512	0,499	0,487
Proximity 50 %							
Xception	0,408	0,570	0,554	0,519	0,500	0,489	0,482
CeNet	0,504	0,650	0,621	0,581	0,562	0,546	0,536
With Concat	0,504	0,649	0,621	0,582	0,562	0,546	0,536
Without Concat	0,528	0,644	0,595	0,557	0,539	0,527	0,518
Proximity 30 %							
Xception	0,432	0,555	0,530	0,482	0,455	0,444	0,438
CeNet	0,464	0,586	0,555	0,502	0,483	0,471	0,464
With Concat	0,464	0,586	0,555	0,502	0,483	0,471	0,464
Without Concat	0,344	0,501	0,503	0,480	0,467	0,462	0,461
Proximity 10 %							
Xception	0,392	0,580	0,532	0,480	0,455	0,444	0,434
CeNet	0,304	0,504	0,481	0,453	0,432	0,419	0,411
With Concat	0,304	0,504	0,482	0,453	0,432	0,419	0,412
Without Concat	0,304	0,535	0,489	0,455	0,447	0,442	0,437

Tabla 11: Proximity Experiment: mAP@k images with covid (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Xception	0,131	0,127	0,106	0,107	0,108	0,110	0,109
CeNet	0,178	0,121	0,112	0,101	0,091	0,077	0,066
With Concat	0,178	0,123	0,113	0,102	0,091	0,078	0,066
Without Concat	0,155	0,084	0,042	0,038	0,037	0,036	0,038
Proximity 70 %							
Xception	0,073	0,066	0,045	0,055	0,055	0,057	0,056
CeNet	0,133	0,089	0,088	0,097	0,091	0,086	0,080
With Concat	0,137	0,092	0,091	0,098	0,092	0,087	0,080
Without Concat	0,197	0,113	0,101	0,098	0,094	0,091	0,088
Proximity 50 %							
Xception	0,095	0,072	0,053	0,055	0,055	0,056	0,056
CeNet	0,149	0,112	0,116	0,110	0,103	0,096	0,089
With Concat	0,149	0,115	0,116	0,110	0,103	0,096	0,089
Without Concat	0,121	0,078	0,078	0,079	0,081	0,081	0,080
Proximity 30 %							
Xception	0,121	0,084	0,080	0,072	0,064	0,059	0,055
CeNet	0,189	0,129	0,117	0,109	0,100	0,093	0,086
With Concat	0,189	0,129	0,117	0,109	0,100	0,093	0,086
Without Concat	0,166	0,120	0,090	0,075	0,072	0,071	0,070
Proximity 10 %							
Xception	0,091	0,039	0,054	0,044	0,040	0,036	0,032
CeNet	0,134	0,116	0,101	0,098	0,097	0,094	0,089
With Concat	0,134	0,116	0,100	0,098	0,097	0,094	0,089
Without Concat	0,092	0,058	0,064	0,070	0,072	0,069	0,066

Tabla 12: Proximity Experiment: mAP@k images with covid (stdev between five different seeds).

Fuente: Elaboración Propia.

Similar al caso anterior, los mejores resultados se concentran al utilizar solo la mitad del dataset. Para el caso particular en que se hace recuperación de imágenes utilizando solo imágenes con COVID-19 como query, *CeNet* en la mayoría de los casos es el modelo que obtiene los mejores resultados, seguido por el modelo *With Concat*, lo que tiene sentido, ya que este modelo se apoya en el embedding de *CeNet*.

Resultados considerando solo imágenes query que no contienen COVID-19:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Xception	0,504	0,659	0,641	0,622	0,623	0,623	0,624
CeNet	0,424	0,563	0,548	0,522	0,531	0,542	0,551
With Concat	0,424	0,563	0,548	0,522	0,531	0,542	0,551
Without Concat	0,464	0,593	0,565	0,563	0,563	0,564	0,564
Proximity 70 %							
Xception	0,600	0,708	0,656	0,634	0,625	0,620	0,616
CeNet	0,400	0,519	0,506	0,473	0,482	0,500	0,511
With Concat	0,400	0,520	0,506	0,473	0,483	0,500	0,511
Without Concat	0,544	0,663	0,627	0,590	0,582	0,579	0,580
Proximity 50 %							
Xception	0,624	0,735	0,696	0,659	0,640	0,628	0,620
CeNet	0,536	0,610	0,587	0,568	0,559	0,559	0,557
With Concat	0,536	0,610	0,587	0,567	0,559	0,559	0,557
Without Concat	0,432	0,563	0,557	0,540	0,533	0,529	0,531
Proximity 30 %							
Xception	0,616	0,754	0,718	0,687	0,671	0,661	0,654
CeNet	0,536	0,662	0,612	0,600	0,598	0,599	0,600
With Concat	0,536	0,662	0,613	0,600	0,599	0,599	0,600
Without Concat	0,512	0,672	0,658	0,633	0,615	0,605	0,598
Proximity 10 %							
Xception	0,616	0,737	0,701	0,679	0,668	0,660	0,654
CeNet	0,488	0,624	0,598	0,601	0,605	0,608	0,609
With Concat	0,480	0,622	0,598	0,601	0,605	0,608	0,609
Without Concat	0,608	0,721	0,677	0,640	0,627	0,619	0,613

Tabla 13: Proximity Experiment: mAP@k images without covid (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Xception	0,112	0,089	0,080	0,076	0,072	0,071	0,071
CeNet	0,244	0,198	0,172	0,180	0,147	0,119	0,096
With Concat	0,244	0,199	0,172	0,181	0,148	0,119	0,097
Without Concat	0,193	0,211	0,184	0,140	0,107	0,089	0,077
Proximity 70 %							
Xception	0,183	0,097	0,085	0,077	0,072	0,068	0,065
CeNet	0,113	0,107	0,096	0,114	0,112	0,102	0,099
With Concat	0,113	0,106	0,095	0,115	0,112	0,102	0,100
Without Concat	0,173	0,118	0,117	0,123	0,121	0,114	0,106
Proximity 50 %							
Xception	0,213	0,109	0,084	0,065	0,063	0,060	0,056
CeNet	0,286	0,325	0,298	0,247	0,214	0,184	0,163
With Concat	0,286	0,325	0,298	0,247	0,214	0,184	0,163
Without Concat	0,250	0,250	0,236	0,224	0,211	0,198	0,177
Proximity 30 %							
Xception	0,073	0,027	0,032	0,032	0,032	0,034	0,034
CeNet	0,199	0,162	0,142	0,140	0,135	0,125	0,117
With Concat	0,199	0,162	0,143	0,140	0,135	0,125	0,117
Without Concat	0,195	0,133	0,106	0,085	0,071	0,058	0,048
Proximity 10 %							
Xception	0,171	0,079	0,082	0,073	0,066	0,060	0,056
CeNet	0,336	0,203	0,181	0,150	0,137	0,128	0,120
With Concat	0,333	0,203	0,181	0,150	0,137	0,128	0,120
Without Concat	0,173	0,124	0,128	0,126	0,117	0,109	0,101

Tabla 14: Proximity Experiment: mAP@k images without covid (stdev between five different seeds).

Fuente: Elaboración Propia.

En este caso *Xception* es el modelo que obtiene los mejores resultados en la mayoría de los casos, superando ampliamente a *CeNet* a medida que se utiliza mayor porcentaje del dataset.

Al igual que el punto anterior, se evalúan los mismos apartados para el caso de los métodos SSL:

- Mean Average Precision (mAP@k).
- Tiempo de recuperación de imágenes.
- Tamaño de los embeddings.

4.3.2. Métodos Self-Supervised

Todos los métodos *Self-supervised* son entrenados usando SARS-CoV-2 CT scan. En el caso del apartado *SSL Methods h* , esto indica que se realizó extracción de embeddings de la salida del encoder, mientras que *SSL Methods z* indica que se realiza la extracción de embeddings de la última capa más cercana a la función de pérdida. Para testing, todos los modelos utilizan el *Query Dataset* como input (query) y utilizan el *Covid-CT-Dataset* como base de datos para obtener los resultados y calcular las distintas métricas.

Resultados considerando todas las imágenes:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,452	0,639	0,606	0,568	0,551	0,555	0,541
Proximity							
Xception	0,456	0,611	0,585	0,554	0,543	0,535	0,530
CeNet	0,420	0,587	0,563	0,530	0,524	0,524	0,523
With Concat	0,420	0,586	0,563	0,530	0,524	0,524	0,523
Without Concat	0,472	0,602	0,563	0,539	0,530	0,525	0,520
SSL Methods h							
Barlow Twins	0,392	0,559	0,530	0,515	0,507	0,502	0,499
BYOL	0,448	0,592	0,565	0,534	0,523	0,518	0,514
SiamSiam	0,464	0,609	0,583	0,551	0,537	0,529	0,524
SimCLR	0,320	0,515	0,506	0,477	0,472	0,471	0,471
SSL Methods z							
Barlow Twins	0,448	0,580	0,552	0,534	0,526	0,521	0,517
BYOL	0,496	0,641	0,604	0,562	0,547	0,538	0,532
SiamSiam	0,488	0,638	0,604	0,563	0,547	0,539	0,534
SimCLR	0,368	0,526	0,507	0,479	0,468	0,463	0,461

Tabla 15: All methods: mAP@k all images (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,050	0,053	0,019	0,014	0,012	0,014	0,011
Proximity							
Xception	0,055	0,062	0,042	0,033	0,030	0,029	0,027
CeNet	0,057	0,069	0,056	0,061	0,050	0,042	0,039
With Concat	0,057	0,069	0,056	0,061	0,050	0,042	0,039
Without Concat	0,100	0,097	0,081	0,057	0,044	0,036	0,030
SSL Methods h							
Barlow Twins	0,129	0,099	0,078	0,055	0,049	0,043	0,039
BYOL	0,050	0,035	0,027	0,033	0,035	0,038	0,038
SiamSiam	0,107	0,044	0,042	0,033	0,032	0,030	0,028
SimCLR	0,135	0,111	0,085	0,062	0,049	0,040	0,033
SSL Methods z							
Barlow Twins	0,160	0,117	0,083	0,059	0,052	0,047	0,041
BYOL	0,061	0,017	0,017	0,020	0,017	0,016	0,015
SiamSiam	0,101	0,065	0,046	0,039	0,033	0,029	0,026
SimCLR	0,067	0,068	0,061	0,053	0,044	0,041	0,039

Tabla 16: All methods: mAP@k all images (stdev between five different seeds).

Fuente: Elaboración Propia.

Los modelos que utilizan SSL obtienen mejores resultados sin utilizar ninguna etiqueta y sin contar con la ayuda de un modelo de segmentación para generar embeddings de calidad. Específicamente, *BYOL* y *SiamSiam* obtienen los mejores resultados, con una desviación estándar menor que los métodos de Proximity. También, se aprecia cómo el embedding h obtenido por el encoder en los métodos *Self-supervised* obtiene una buena representación, la cual puede ser utilizada posteriormente como inicialización para alguna downstream task.

Resultados considerando solo imágenes query que contienen COVID-19:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,432	0,620	0,594	0,551	0,532	0,528	0,511
Proximity							
Xception	0,408	0,563	0,529	0,485	0,463	0,447	0,436
CeNet	0,416	0,611	0,578	0,538	0,517	0,506	0,496
With Concat	0,416	0,609	0,577	0,537	0,517	0,506	0,496
Without Concat	0,480	0,612	0,562	0,515	0,496	0,485	0,477
SSL Methods h							
Barlow Twins	0,448	0,574	0,524	0,485	0,462	0,449	0,442
BYOL	0,504	0,597	0,557	0,498	0,474	0,460	0,452
SiamSiam	0,496	0,582	0,546	0,499	0,481	0,472	0,467
SimCLR	0,272	0,466	0,441	0,384	0,366	0,357	0,352
SSL Methods z							
Barlow Twins	0,448	0,578	0,547	0,508	0,491	0,482	0,476
BYOL	0,472	0,634	0,585	0,542	0,526	0,517	0,512
SiamSiam	0,448	0,590	0,558	0,515	0,500	0,493	0,488
SimCLR	0,272	0,436	0,420	0,383	0,367	0,361	0,360

Tabla 17: All methods: mAP@k images with covid (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,077	0,076	0,027	0,027	0,009	0,018	0,010
Proximity							
Xception	0,131	0,127	0,106	0,107	0,108	0,110	0,109
CeNet	0,178	0,121	0,112	0,101	0,091	0,077	0,066
With Concat	0,178	0,123	0,113	0,102	0,091	0,078	0,066
Without Concat	0,155	0,084	0,042	0,038	0,037	0,036	0,038
SSL Methods h							
Barlow Twins	0,270	0,198	0,176	0,150	0,141	0,132	0,121
BYOL	0,151	0,102	0,110	0,091	0,082	0,076	0,070
SiamSiam	0,166	0,119	0,108	0,100	0,096	0,086	0,078
SimCLR	0,203	0,143	0,102	0,039	0,024	0,022	0,024
SSL Methods z							
Barlow Twins	0,222	0,201	0,179	0,160	0,153	0,147	0,137
BYOL	0,111	0,079	0,045	0,047	0,041	0,042	0,042
SiamSiam	0,134	0,096	0,061	0,050	0,045	0,044	0,044
SimCLR	0,118	0,099	0,079	0,058	0,053	0,049	0,047

Tabla 18: All methods: mAP@k images with covid (stdev between five different seeds).

Fuente: Elaboración Propia.

Al contar con solo imágenes de tipo COVID-19 como query, BYOL logra liderar la posición en

la mayoría de los casos y contrario al caso anterior, *SiamSiam* no logra superar a *BYOL* en ningún caso.

Resultados considerando solo imágenes query que no contienen COVID-19:

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,472	0,658	0,617	0,586	0,569	0,581	0,571
Proximity							
Xception	0,504	0,659	0,641	0,622	0,623	0,623	0,624
CeNet	0,424	0,563	0,548	0,522	0,531	0,542	0,551
With Concat	0,424	0,563	0,548	0,522	0,531	0,542	0,551
Without Concat	0,464	0,593	0,565	0,563	0,563	0,564	0,564
SSL Methods h							
Barlow Twins	0,336	0,544	0,537	0,546	0,552	0,556	0,556
BYOL	0,392	0,586	0,573	0,569	0,572	0,575	0,577
SiamSiam	0,432	0,636	0,620	0,603	0,593	0,586	0,581
SimCLR	0,368	0,564	0,571	0,570	0,579	0,585	0,590
SSL Methods z							
Barlow Twins	0,448	0,582	0,557	0,560	0,560	0,560	0,558
BYOL	0,520	0,649	0,622	0,582	0,569	0,559	0,553
SiamSiam	0,528	0,687	0,650	0,611	0,594	0,585	0,580
SimCLR	0,464	0,617	0,594	0,575	0,569	0,565	0,562

Tabla 19: All models: mAP@k images without covid (mean between five different seeds).

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,052	0,066	0,022	0,018	0,023	0,026	0,015
Proximity							
Xception	0,112	0,089	0,080	0,076	0,072	0,071	0,071
CeNet	0,244	0,198	0,172	0,180	0,147	0,119	0,096
With Concat	0,244	0,199	0,172	0,181	0,148	0,119	0,097
Without Concat	0,193	0,211	0,184	0,140	0,107	0,089	0,077
SSL Methods h							
Barlow Twins	0,164	0,090	0,080	0,071	0,067	0,067	0,071
BYOL	0,175	0,126	0,104	0,089	0,080	0,074	0,069
SiamSiam	0,186	0,102	0,091	0,073	0,062	0,053	0,047
SimCLR	0,193	0,190	0,139	0,117	0,100	0,085	0,075
SSL Methods z							
Barlow Twins	0,125	0,077	0,058	0,052	0,052	0,058	0,063
BYOL	0,102	0,051	0,037	0,030	0,028	0,024	0,023
SiamSiam	0,072	0,042	0,040	0,049	0,049	0,047	0,046
SimCLR	0,173	0,132	0,116	0,115	0,107	0,103	0,099

Tabla 20: All models: mAP@k images without covid (stdev between five different seeds).

Fuente: Elaboración Propia.

Contrario al caso anterior, al contar solo con imágenes sin COVID-19 como query, *Xception* sí logra superar a los modelos SSL en algunos casos particulares. Específicamente, *SiamSiam* obtiene mejores resultados solo en la parte inicial del retrieval, que es hasta la recuperación de 20 imágenes por query.

Para el cálculo de tiempos en recuperación de imágenes, se utiliza la misma metodología anterior, es decir, se procesan cada una de las 50 imágenes tipo query y se calcula el tiempo en segundos que le toma al modelo recuperar las imágenes a medida que se varía el tamaño de recuperación entre 1 hasta 50 imágenes. Finalmente, se extrae la media de estos tiempos, a través de las 50 queries.

	k=1	k=5	k=10	k=20	k=30	k=40	k=50
Proximity							
Xception	1,489	1,659	1,788	2,242	3,275	4,506	6,329
CeNet	0,696	0,753	0,752	0,931	1,304	1,730	2,237
With Concat	0,805	0,868	1,004	1,501	2,263	3,241	4,575
Without Concat	0,834	0,876	1,023	1,317	1,862	2,603	3,542
SSL Methods h							
Barlow Twins	0,068	0,036	0,051	0,105	0,222	0,316	0,464
BYOL	0,032	0,038	0,059	0,114	0,234	0,317	0,478
SiamSiam	0,029	0,027	0,052	0,108	0,194	0,311	0,465
SimCLR	0,059	0,025	0,050	0,112	0,194	0,342	0,479
SSL Methods z							
Barlow Twins	0,022	0,036	0,051	0,112	0,204	0,332	0,493
BYOL	0,030	0,032	0,081	0,102	0,178	0,295	0,444
SiamSiam	0,035	0,026	0,081	0,107	0,190	0,346	0,470
SimCLR	0,027	0,044	0,048	0,105	0,218	0,292	0,430

Tabla 21: All methods: Median in seconds to retrieve k images across the whole query dataset.

Fuente: Elaboración Propia.

Los resultados relacionados con el tiempo están directamente relacionados con el tamaño del embedding utilizado para la recuperación de imágenes. Como en Proximity no se controla el tamaño de la imagen de entrada ni el vector de salida de la red, se obtiene el peor resultado y se demora aproximadamente 12 veces más que los modelos SSL. Como este tiempo crece exponencialmente a medida que aumentamos la cantidad de imágenes recuperadas, es esencial tomar en cuenta el tiempo, además del mAP@k.

Finalmente, se evalúa el tamaño del embedding resultante de cada modelo:

	Dimensions	Size
Proximity		
Xception	524.288	1,5 GB
CeNet	132.096	376 MB
With Concat	402.432	1,1 GB
Without Concat	270.336	769,4 MB
SSL Methods h		
Barlow Twins	2.048	5,9 MB
BYOL	2.048	5,9 MB
SiamSiam	2.048	5,9 MB
SimCLR	2.048	5,9 MB
SSL Methods z		
Barlow Twins	8.192	23,4 MB
BYOL	256	840 KB
SiamSiam	2.048	5,9 MB
SimCLR	128	467 KB

Tabla 22: All methods: Embeddings dimension and size of every method.

Fuente: Elaboración Propia.

Al utilizar *ResNet-50*, los modelos *SSL h* solo producen embeddings de tamaño 2048, lo cual es al menos 65 veces menor que los embeddings obtenidos por *CeNet*, el modelo que produce embeddings de menor dimensionalidad para el caso de Proximity. Para el caso de *SSL z* , sí se obtiene una variación más amplia en relación a los tamaños de los embeddings, y uno de los mejores modelos es *BYOL*, que obtiene un buen rendimiento con un tamaño de embedding de tan solo 256. De esta forma, con los métodos *Self-supervised*, solo se necesitan 5.9 Megabytes como máximo en promedio para almacenar la base de datos.

CAPÍTULO 5

CONCLUSIONES

5.1. Conclusiones generales y contribuciones

En el presente trabajo se logra recrear los modelos investigados en Proximity para *Content-based image retrieval*: *Xception*, *CeNet*, *With Concat* y *Without Concat*. Además, se replican los resultados obtenidos usando la misma metodología expuesta en el paper. Se calculan las distintas métricas para tres casos distintos: todas las imágenes del dataset query como imagen de entrada, solo las imágenes que tienen COVID-19 como imagen de entrada y finalmente, solo las imágenes que no tienen COVID-19 como imagen de entrada. A base de los resultados obtenidos se detectan diferentes problemas, uno de ellos es sencillo de resolver, y hace referencia a la obtención de embeddings muy grande, lo que impacta directamente al tiempo de recuperación de las imágenes. Esto es debido a que no se realiza un *resize* a las imágenes que son pasadas por la red neuronal. Si bien una red neuronal puede recibir una imagen de cualquier tamaño, las arquitecturas convolucionales presentan un tamaño de input predefinido, que si se utiliza, se termina replicando la arquitectura idénticamente a como la definen los autores, y de esta forma, obteniendo la misma dimensión en la última capa oculta de la red. En el caso de *ResNet-50* se espera un input con dimensiones de $224 \times 224 \times 3$, para obtener una última capa de dimensión 2,048. Para el caso de *Xception*, se espera un input de $299 \times 299 \times 3$, para obtener una última capa con tamaño de 2,048 también. En Proximity se decide pasar un input de tamaño $512 \times 512 \times 3$, y esa es una de la principal razón de la obtención de tamaños de embeddings muy grandes, solo para este caso se obtiene un embedding de 524,288 dimensiones, en vez de obtener algo de un tamaño de 2,048, que es alrededor de 256 veces menor que lo obtenido.

A base de lo anterior, lo primero que se realiza es decidir un tamaño estándar para todas las imágenes utilizadas, tanto en entrenamiento como en testing o bien recuperación de las imágenes. Basándose netamente en la arquitectura a utilizar, que en este caso es *ResNet-50*. Posteriormente, se realiza la implementación de cuatro modelos distintos relacionados con el *Self-supervised learning*: *SimCLR*, *BYOL*, *SiamSiam* y *Barlow Twins*. Con ello se logra mejorar los métodos propuestos en Proximity considerando tres aspectos fundamentales en la recuperación de imágenes: precisión, tiempo y tamaño de embeddings. Se demuestra experimentalmente que los métodos de tipo *Self-supervised Learning* son capaces de generar embeddings de calidad sin depender de ninguna etiqueta, ni tampoco depender de pesos preentrenados como Proximity con el uso de pesos relacionados con *Imagenet*. Como estos métodos no requieren de una red especializada en segmentación, no se necesita realizar más de un entrenamiento, y estos modelos SSL pueden expandirse fácilmente a problemas con presencia de otros tipos de enfermedades, ya que para ese caso *CeNet* necesitaría pares de segmentación relacionados con los otros tipos de enfermedades que se requieran incorporar, y como ya contar con imágenes de tipo segmentación de por sí es difícil, se complica

aún más en el caso de imágenes médicas.

5.2. Alcance y limitaciones

Como bien sabemos para todo tipo de entrenamiento es importante la calidad de la data, al igual que lo es la cantidad de data que se tiene, para que los modelos no caigan en el problema ya conocido de *overfitting*. En este caso todos los dataset utilizados son distintos entre sí, lo cual puede influir en el resultado, incluso se puede apreciar en las imágenes de muestra en cada dataset, que existe cierta variación de intensidad en los píxeles de las imágenes. Además, algunas presentan anotaciones dentro de la imagen, lo cual puede afectar directamente a la precisión del modelo, ya que, debido a ello, la data obtiene un cambio de distribución entre los distintos datasets.

Otro de los inconvenientes, es que en total se tiene como máximo alrededor de 6000 imágenes para entrenar, validar y testear el modelo. Al utilizar modelos de deep learning, tener hartos datos es importante para no caer en problemas de *overfitting*.

5.3. Trabajo futuro

Como trabajo futuro se plantea seguir investigando el área del *Self-supervised*, pero considerando la implementación de otras arquitecturas que incluyan modelos de tipo transformers, como lo es ViT (Vision Transformers) [Dosovitskiy *et al.*, 2020] o bien otras arquitecturas que no dependan de redes siamesas como lo son los *Masked Autoencoders* [He *et al.*, 2022].

Otro aspecto importante es volver a entrenar los modelos con más datos y expandir el problema de clasificación binaria a clasificación Multi-Clase.

Por último, investigar en mayor profundidad el experimento realizado en Proximity, al disminuir el porcentaje de data etiquetada, ya que al tener menor data etiquetada lo esperado es que el modelo obtenga peores resultados, pero pasa exactamente lo contrario.

ANEXOS

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,420	0,564	0,544	0,524	0,520	0,516	0,514
Baseline 2: CeNet	0,360	0,494	0,499	0,449	0,467	0,484	0,493
Proposal 1: With Concat	0,360	0,494	0,499	0,449	0,467	0,484	0,493
Proposal 2: Without Concat	0,420	0,476	0,450	0,459	0,469	0,474	0,479
Proximity 70 %							
Baseline 1: Xception	0,580	0,683	0,601	0,562	0,543	0,537	0,530
Baseline 2: CeNet	0,460	0,600	0,581	0,538	0,532	0,529	0,529
Proposal 1: With Concat	0,460	0,600	0,579	0,538	0,532	0,529	0,529
Proposal 2: Without Concat	0,420	0,632	0,604	0,577	0,562	0,553	0,544
Proximity 50 %							
Baseline 1: Xception	0,520	0,645	0,617	0,575	0,555	0,541	0,533
Baseline 2: CeNet	0,600	0,686	0,654	0,608	0,582	0,569	0,561
Proposal 1: With Concat	0,600	0,686	0,654	0,608	0,581	0,569	0,561
Proposal 2: Without Concat	0,500	0,660	0,623	0,590	0,576	0,565	0,557
Proximity 30 %							
Baseline 1: Xception	0,600	0,688	0,649	0,598	0,569	0,557	0,548
Baseline 2: CeNet	0,560	0,676	0,600	0,558	0,545	0,537	0,530
Proposal 1: With Concat	0,560	0,676	0,602	0,558	0,545	0,537	0,531
Proposal 2: Without Concat	0,440	0,616	0,612	0,594	0,578	0,564	0,553
Proximity 10 %							
Baseline 1: Xception	0,600	0,694	0,649	0,599	0,575	0,565	0,556
Baseline 2: CeNet	0,480	0,602	0,559	0,530	0,518	0,511	0,508
Proposal 1: With Concat	0,460	0,600	0,560	0,529	0,517	0,511	0,508
Proposal 2: Without Concat	0,420	0,605	0,568	0,546	0,542	0,538	0,533

Tabla 23: Proximity Experiment: mAP@k all images seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,440	0,618	0,576	0,553	0,545	0,535	0,528
Baseline 2: CeNet	0,400	0,582	0,550	0,522	0,510	0,507	0,507
Proposal 1: With Concat	0,400	0,582	0,550	0,521	0,510	0,506	0,507
Proposal 2: Without Concat	0,340	0,527	0,505	0,497	0,497	0,500	0,498
Proximity 70 %							
Baseline 1: Xception	0,500	0,615	0,577	0,535	0,524	0,517	0,512
Baseline 2: CeNet	0,420	0,539	0,534	0,503	0,502	0,505	0,507
Proposal 1: With Concat	0,400	0,538	0,532	0,502	0,502	0,505	0,506
Proposal 2: Without Concat	0,560	0,681	0,644	0,580	0,556	0,544	0,534
Proximity 50 %							
Baseline 1: Xception	0,640	0,726	0,690	0,640	0,617	0,606	0,597
Baseline 2: CeNet	0,540	0,629	0,599	0,575	0,559	0,550	0,546
Proposal 1: With Concat	0,540	0,626	0,599	0,575	0,559	0,550	0,547
Proposal 2: Without Concat	0,500	0,615	0,585	0,567	0,555	0,547	0,541
Proximity 30 %							
Baseline 1: Xception	0,500	0,656	0,636	0,597	0,574	0,562	0,553
Baseline 2: CeNet	0,420	0,576	0,554	0,541	0,543	0,543	0,543
Proposal 1: With Concat	0,430	0,576	0,554	0,541	0,543	0,543	0,543
Proposal 2: Without Concat	0,440	0,564	0,552	0,526	0,514	0,507	0,503
Proximity 10 %							
Baseline 1: Xception	0,480	0,649	0,595	0,558	0,545	0,537	0,533
Baseline 2: CeNet	0,460	0,578	0,571	0,553	0,540	0,535	0,530
Proposal 1: With Concat	0,460	0,578	0,571	0,553	0,540	0,535	0,530
Proposal 2: Without Concat	0,520	0,667	0,617	0,562	0,540	0,531	0,525

Tabla 24: Proximity Experiment: mAP@k all images seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,540	0,684	0,625	0,565	0,547	0,532	0,522
Baseline 2: CeNet	0,480	0,662	0,640	0,613	0,599	0,593	0,589
Proposal 1: With Concat	0,480	0,661	0,640	0,613	0,599	0,592	0,589
Proposal 2: Without Concat	0,580	0,690	0,622	0,578	0,557	0,546	0,537
Proximity 70 %							
Baseline 1: Xception	0,320	0,527	0,517	0,513	0,506	0,498	0,493
Baseline 2: CeNet	0,520	0,565	0,545	0,484	0,481	0,493	0,492
Proposal 1: With Concat	0,520	0,565	0,544	0,484	0,481	0,493	0,492
Proposal 2: Without Concat	0,460	0,591	0,556	0,535	0,535	0,531	0,529
Proximity 50 %							
Baseline 1: Xception	0,500	0,651	0,622	0,608	0,593	0,581	0,572
Baseline 2: CeNet	0,500	0,665	0,647	0,594	0,571	0,556	0,546
Proposal 1: With Concat	0,500	0,665	0,648	0,593	0,571	0,556	0,545
Proposal 2: Without Concat	0,540	0,653	0,607	0,567	0,549	0,538	0,531
Proximity 30 %							
Baseline 1: Xception	0,400	0,579	0,553	0,524	0,512	0,507	0,503
Baseline 2: CeNet	0,500	0,677	0,645	0,614	0,602	0,593	0,585
Proposal 1: With Concat	0,500	0,677	0,645	0,615	0,602	0,593	0,585
Proposal 2: Without Concat	0,440	0,595	0,592	0,563	0,551	0,543	0,540
Proximity 10 %							
Baseline 1: Xception	0,480	0,664	0,631	0,596	0,575	0,564	0,554
Baseline 2: CeNet	0,500	0,629	0,581	0,554	0,542	0,531	0,526
Proposal 1: With Concat	0,500	0,630	0,581	0,554	0,542	0,531	0,526
Proposal 2: Without Concat	0,520	0,688	0,643	0,595	0,580	0,568	0,559

Tabla 25: Proximity Experiment: mAP@k all images seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,400	0,536	0,547	0,523	0,514	0,509	0,508
Baseline 2: CeNet	0,480	0,646	0,596	0,557	0,542	0,532	0,527
Proposal 1: With Concat	0,480	0,646	0,597	0,558	0,542	0,533	0,527
Proposal 2: Without Concat	0,460	0,628	0,616	0,583	0,567	0,555	0,549
Proximity 70 %							
Baseline 1: Xception	0,540	0,641	0,601	0,558	0,544	0,532	0,525
Baseline 2: CeNet	0,460	0,668	0,598	0,536	0,512	0,505	0,501
Proposal 1: With Concat	0,460	0,668	0,598	0,536	0,512	0,505	0,501
Proposal 2: Without Concat	0,520	0,669	0,616	0,571	0,557	0,549	0,544
Proximity 50 %							
Baseline 1: Xception	0,360	0,566	0,570	0,539	0,521	0,514	0,510
Baseline 2: CeNet	0,580	0,757	0,694	0,654	0,635	0,617	0,604
Proposal 1: With Concat	0,580	0,757	0,694	0,654	0,635	0,617	0,604
Proposal 2: Without Concat	0,500	0,654	0,635	0,602	0,583	0,568	0,558
Proximity 30 %							
Baseline 1: Xception	0,540	0,637	0,600	0,560	0,540	0,531	0,527
Baseline 2: CeNet	0,540	0,633	0,584	0,539	0,524	0,517	0,514
Proposal 1: With Concat	0,540	0,633	0,584	0,539	0,525	0,517	0,514
Proposal 2: Without Concat	0,440	0,579	0,579	0,546	0,517	0,505	0,499
Proximity 10 %							
Baseline 1: Xception	0,460	0,638	0,606	0,561	0,544	0,535	0,527
Baseline 2: CeNet	0,320	0,533	0,528	0,521	0,512	0,503	0,495
Proposal 1: With Concat	0,320	0,533	0,529	0,521	0,512	0,504	0,495
Proposal 2: Without Concat	0,520	0,640	0,571	0,545	0,537	0,529	0,525

Tabla 26: Proximity Experiment: mAP@k all images seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,480	0,655	0,633	0,603	0,590	0,582	0,577
Baseline 2: CeNet	0,380	0,551	0,530	0,509	0,502	0,503	0,501
Proposal 1: With Concat	0,380	0,548	0,528	0,509	0,502	0,503	0,501
Proposal 2: Without Concat	0,560	0,690	0,624	0,577	0,558	0,549	0,539
Proximity 70 %							
Baseline 1: Xception	0,500	0,645	0,588	0,554	0,541	0,532	0,525
Baseline 2: CeNet	0,440	0,579	0,563	0,536	0,533	0,529	0,523
Proposal 1: With Concat	0,440	0,579	0,563	0,536	0,533	0,529	0,523
Proposal 2: Without Concat	0,540	0,654	0,597	0,545	0,524	0,519	0,517
Proximity 50 %							
Baseline 1: Xception	0,560	0,675	0,626	0,583	0,563	0,551	0,543
Baseline 2: CeNet	0,380	0,414	0,425	0,443	0,454	0,469	0,477
Proposal 1: With Concat	0,380	0,414	0,425	0,443	0,454	0,469	0,477
Proposal 2: Without Concat	0,360	0,435	0,429	0,414	0,417	0,421	0,436
Proximity 30 %							
Baseline 1: Xception	0,580	0,712	0,682	0,644	0,620	0,605	0,596
Baseline 2: CeNet	0,480	0,559	0,536	0,503	0,489	0,486	0,486
Proposal 1: With Concat	0,480	0,559	0,536	0,503	0,489	0,486	0,486
Proposal 2: Without Concat	0,400	0,580	0,568	0,554	0,546	0,548	0,552
Proximity 10 %							
Baseline 1: Xception	0,500	0,648	0,601	0,583	0,569	0,560	0,552
Baseline 2: CeNet	0,220	0,475	0,459	0,476	0,481	0,487	0,492
Proposal 1: With Concat	0,220	0,475	0,459	0,476	0,481	0,487	0,492
Proposal 2: Without Concat	0,300	0,540	0,515	0,490	0,485	0,484	0,484

Tabla 27: Proximity Experiment: mAP@k all images seed 44.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,520	0,618	0,567	0,527	0,500	0,478	0,466
Baseline 2: CeNet	0,600	0,707	0,692	0,657	0,628	0,602	0,573
Proposal 1: With Concat	0,600	0,707	0,692	0,658	0,628	0,603	0,573
Proposal 2: Without Concat	0,640	0,685	0,617	0,577	0,549	0,530	0,520
Proximity 70 %							
Baseline 1: Xception	0,440	0,629	0,530	0,491	0,469	0,460	0,448
Baseline 2: CeNet	0,320	0,546	0,519	0,444	0,424	0,415	0,410
Proposal 1: With Concat	0,320	0,546	0,519	0,443	0,425	0,415	0,411
Proposal 2: Without Concat	0,320	0,549	0,485	0,435	0,410	0,397	0,385
Proximity 50 %							
Baseline 1: Xception	0,280	0,471	0,475	0,445	0,424	0,407	0,399
Baseline 2: CeNet	0,560	0,638	0,609	0,551	0,524	0,514	0,509
Proposal 1: With Concat	0,560	0,638	0,610	0,552	0,524	0,514	0,509
Proposal 2: Without Concat	0,560	0,691	0,599	0,560	0,540	0,527	0,518
Proximity 30 %							
Baseline 1: Xception	0,560	0,611	0,576	0,515	0,474	0,461	0,450
Baseline 2: CeNet	0,440	0,585	0,534	0,484	0,462	0,449	0,437
Proposal 1: With Concat	0,440	0,585	0,534	0,484	0,462	0,449	0,437
Proposal 2: Without Concat	0,160	0,410	0,452	0,454	0,452	0,447	0,444
Proximity 10 %							
Baseline 1: Xception	0,320	0,532	0,480	0,426	0,399	0,390	0,383
Baseline 2: CeNet	0,360	0,521	0,462	0,415	0,388	0,371	0,366
Proposal 1: With Concat	0,360	0,522	0,467	0,416	0,388	0,371	0,366
Proposal 2: Without Concat	0,280	0,476	0,439	0,412	0,412	0,409	0,408

Tabla 28: Proximity Experiment: mAP@k images with covid seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,400	0,587	0,519	0,487	0,474	0,461	0,452
Baseline 2: CeNet	0,600	0,734	0,670	0,596	0,556	0,531	0,515
Proposal 1: With Concat	0,600	0,734	0,670	0,595	0,556	0,531	0,515
Proposal 2: Without Concat	0,360	0,562	0,536	0,484	0,460	0,445	0,432
Proximity 70 %							
Baseline 1: Xception	0,440	0,532	0,524	0,477	0,459	0,449	0,442
Baseline 2: CeNet	0,480	0,585	0,534	0,479	0,461	0,448	0,436
Proposal 1: With Concat	0,440	0,575	0,525	0,475	0,459	0,446	0,435
Proposal 2: Without Concat	0,360	0,563	0,563	0,500	0,475	0,462	0,450
Proximity 50 %							
Baseline 1: Xception	0,480	0,622	0,594	0,548	0,528	0,521	0,520
Baseline 2: CeNet	0,320	0,496	0,472	0,461	0,457	0,455	0,458
Proposal 1: With Concat	0,320	0,489	0,472	0,461	0,457	0,455	0,458
Proposal 2: Without Concat	0,400	0,555	0,525	0,486	0,471	0,465	0,461
Proximity 30 %							
Baseline 1: Xception	0,480	0,583	0,546	0,483	0,451	0,434	0,425
Baseline 2: CeNet	0,240	0,413	0,425	0,412	0,418	0,420	0,424
Proposal 1: With Concat	0,240	0,413	0,425	0,412	0,418	0,420	0,424
Proposal 2: Without Concat	0,280	0,407	0,425	0,429	0,432	0,431	0,435
Proximity 10 %							
Baseline 1: Xception	0,280	0,572	0,522	0,474	0,454	0,444	0,440
Baseline 2: CeNet	0,080	0,308	0,322	0,308	0,293	0,287	0,285
Proposal 1: With Concat	0,080	0,308	0,322	0,307	0,293	0,287	0,285
Proposal 2: Without Concat	0,200	0,479	0,420	0,365	0,346	0,344	0,343

Tabla 29: Proximity Experiment: mAP@k images with covid seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,520	0,651	0,572	0,492	0,458	0,430	0,412
Baseline 2: CeNet	0,360	0,602	0,575	0,543	0,528	0,521	0,519
Proposal 1: With Concat	0,360	0,600	0,575	0,542	0,528	0,520	0,518
Proposal 2: Without Concat	0,600	0,681	0,584	0,525	0,517	0,511	0,504
Proximity 70 %							
Baseline 1: Xception	0,320	0,497	0,490	0,470	0,449	0,433	0,420
Baseline 2: CeNet	0,680	0,725	0,691	0,640	0,614	0,599	0,581
Proposal 1: With Concat	0,680	0,725	0,691	0,640	0,614	0,599	0,581
Proposal 2: Without Concat	0,360	0,570	0,530	0,499	0,485	0,471	0,461
Proximity 50 %							
Baseline 1: Xception	0,520	0,656	0,608	0,591	0,572	0,557	0,545
Baseline 2: CeNet	0,440	0,609	0,587	0,536	0,517	0,499	0,487
Proposal 1: With Concat	0,440	0,609	0,588	0,536	0,517	0,498	0,487
Proposal 2: Without Concat	0,480	0,594	0,540	0,510	0,492	0,481	0,470
Proximity 30 %							
Baseline 1: Xception	0,240	0,418	0,396	0,370	0,358	0,354	0,354
Baseline 2: CeNet	0,480	0,676	0,629	0,568	0,546	0,533	0,521
Proposal 1: With Concat	0,480	0,676	0,629	0,568	0,546	0,533	0,521
Proposal 2: Without Concat	0,280	0,499	0,512	0,486	0,481	0,476	0,476
Proximity 10 %							
Baseline 1: Xception	0,400	0,581	0,540	0,495	0,464	0,454	0,442
Baseline 2: CeNet	0,280	0,509	0,488	0,458	0,436	0,422	0,418
Proposal 1: With Concat	0,280	0,511	0,489	0,458	0,436	0,422	0,419
Proposal 2: Without Concat	0,400	0,612	0,551	0,485	0,474	0,467	0,460

Tabla 30: Proximity Experiment: mAP@k images with covid seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,200	0,339	0,353	0,313	0,293	0,280	0,273
Baseline 2: CeNet	0,320	0,581	0,539	0,505	0,491	0,484	0,480
Proposal 1: With Concat	0,320	0,581	0,540	0,505	0,492	0,484	0,480
Proposal 2: Without Concat	0,280	0,491	0,509	0,496	0,486	0,482	0,485
Proximity 70 %							
Baseline 1: Xception	0,280	0,458	0,421	0,357	0,341	0,327	0,320
Baseline 2: CeNet	0,560	0,733	0,675	0,623	0,591	0,570	0,553
Proposal 1: With Concat	0,560	0,733	0,675	0,623	0,592	0,570	0,553
Proposal 2: Without Concat	0,440	0,639	0,573	0,538	0,528	0,524	0,521
Proximity 50 %							
Baseline 1: Xception	0,400	0,542	0,556	0,511	0,495	0,490	0,485
Baseline 2: CeNet	0,480	0,711	0,641	0,601	0,581	0,557	0,541
Proposal 1: With Concat	0,480	0,711	0,641	0,602	0,581	0,557	0,541
Proposal 2: Without Concat	0,480	0,629	0,588	0,541	0,515	0,496	0,486
Proximity 30 %							
Baseline 1: Xception	0,400	0,535	0,529	0,477	0,457	0,454	0,454
Baseline 2: CeNet	0,400	0,518	0,474	0,393	0,369	0,360	0,359
Proposal 1: With Concat	0,400	0,518	0,474	0,393	0,370	0,360	0,359
Proposal 2: Without Concat	0,400	0,489	0,474	0,425	0,390	0,382	0,380
Proximity 10 %							
Baseline 1: Xception	0,480	0,641	0,619	0,545	0,511	0,491	0,473
Baseline 2: CeNet	0,400	0,576	0,559	0,538	0,520	0,504	0,488
Proposal 1: With Concat	0,400	0,576	0,559	0,538	0,521	0,504	0,488
Proposal 2: Without Concat	0,400	0,543	0,475	0,465	0,464	0,459	0,455

Tabla 31: Proximity Experiment: mAP@k images with covid seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,400	0,621	0,632	0,605	0,591	0,584	0,576
Baseline 2: CeNet	0,200	0,430	0,414	0,388	0,382	0,390	0,394
Proposal 1: With Concat	0,200	0,423	0,409	0,387	0,381	0,390	0,393
Proposal 2: Without Concat	0,520	0,642	0,563	0,494	0,468	0,457	0,442
Proximity 70 %							
Baseline 1: Xception	0,400	0,566	0,522	0,481	0,470	0,464	0,458
Baseline 2: CeNet	0,560	0,721	0,694	0,645	0,615	0,591	0,568
Proposal 1: With Concat	0,560	0,721	0,694	0,645	0,615	0,591	0,569
Proposal 2: Without Concat	0,800	0,820	0,749	0,695	0,662	0,640	0,618
Proximity 50 %							
Baseline 1: Xception	0,360	0,560	0,536	0,499	0,480	0,470	0,461
Baseline 2: CeNet	0,720	0,796	0,795	0,757	0,729	0,705	0,686
Proposal 1: With Concat	0,720	0,796	0,795	0,757	0,729	0,705	0,686
Proposal 2: Without Concat	0,720	0,751	0,723	0,688	0,677	0,665	0,656
Proximity 30 %							
Baseline 1: Xception	0,480	0,628	0,602	0,566	0,537	0,518	0,505
Baseline 2: CeNet	0,760	0,739	0,713	0,653	0,619	0,594	0,578
Proposal 1: With Concat	0,760	0,739	0,713	0,654	0,619	0,595	0,578
Proposal 2: Without Concat	0,600	0,702	0,653	0,606	0,582	0,573	0,570
Proximity 10 %							
Baseline 1: Xception	0,480	0,572	0,497	0,461	0,445	0,440	0,433
Baseline 2: CeNet	0,400	0,604	0,574	0,545	0,523	0,510	0,500
Proposal 1: With Concat	0,400	0,604	0,574	0,545	0,523	0,510	0,500
Proposal 2: Without Concat	0,240	0,563	0,560	0,547	0,537	0,529	0,521

Tabla 32: Proximity Experiment: mAP@k images with covid seed 44.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,320	0,510	0,522	0,522	0,539	0,553	0,563
Baseline 2: CeNet	0,120	0,280	0,305	0,240	0,305	0,366	0,413
Proposal 1: With Concat	0,120	0,280	0,305	0,240	0,305	0,366	0,413
Proposal 2: Without Concat	0,200	0,268	0,284	0,341	0,390	0,418	0,438
Proximity 70 %							
Baseline 1: Xception	0,720	0,736	0,672	0,633	0,618	0,614	0,612
Baseline 2: CeNet	0,600	0,654	0,644	0,633	0,639	0,643	0,648
Proposal 1: With Concat	0,600	0,654	0,639	0,633	0,639	0,643	0,648
Proposal 2: Without Concat	0,520	0,716	0,723	0,719	0,715	0,709	0,704
Proximity 50 %							
Baseline 1: Xception	0,760	0,820	0,759	0,706	0,687	0,674	0,666
Baseline 2: CeNet	0,640	0,734	0,698	0,665	0,639	0,625	0,613
Proposal 1: With Concat	0,640	0,734	0,698	0,664	0,639	0,625	0,613
Proposal 2: Without Concat	0,440	0,630	0,647	0,621	0,612	0,603	0,596
Proximity 30 %							
Baseline 1: Xception	0,640	0,765	0,721	0,680	0,664	0,653	0,646
Baseline 2: CeNet	0,680	0,768	0,666	0,632	0,628	0,625	0,624
Proposal 1: With Concat	0,680	0,768	0,671	0,632	0,628	0,625	0,624
Proposal 2: Without Concat	0,720	0,822	0,772	0,734	0,704	0,680	0,662
Proximity 10 %							
Baseline 1: Xception	0,880	0,856	0,819	0,772	0,751	0,739	0,730
Baseline 2: CeNet	0,600	0,684	0,655	0,644	0,648	0,651	0,651
Proposal 1: With Concat	0,560	0,678	0,652	0,643	0,647	0,651	0,651
Proposal 2: Without Concat	0,560	0,734	0,698	0,681	0,673	0,666	0,659

Tabla 33: Proximity Experiment: mAP@k images without covid seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,480	0,648	0,633	0,618	0,615	0,610	0,605
Baseline 2: CeNet	0,200	0,429	0,430	0,448	0,465	0,482	0,499
Proposal 1: With Concat	0,200	0,429	0,430	0,448	0,465	0,482	0,499
Proposal 2: Without Concat	0,320	0,493	0,473	0,510	0,534	0,554	0,565
Proximity 70 %							
Baseline 1: Xception	0,560	0,699	0,629	0,593	0,588	0,586	0,582
Baseline 2: CeNet	0,360	0,494	0,535	0,527	0,544	0,562	0,577
Proposal 1: With Concat	0,360	0,501	0,539	0,528	0,545	0,563	0,578
Proposal 2: Without Concat	0,760	0,800	0,725	0,660	0,637	0,626	0,618
Proximity 50 %							
Baseline 1: Xception	0,800	0,830	0,787	0,732	0,705	0,690	0,675
Baseline 2: CeNet	0,760	0,763	0,726	0,689	0,661	0,645	0,635
Proposal 1: With Concat	0,760	0,763	0,726	0,688	0,661	0,645	0,635
Proposal 2: Without Concat	0,600	0,676	0,646	0,649	0,639	0,630	0,622
Proximity 30 %							
Baseline 1: Xception	0,520	0,729	0,726	0,712	0,697	0,690	0,682
Baseline 2: CeNet	0,600	0,738	0,682	0,670	0,668	0,665	0,663
Proposal 1: With Concat	0,600	0,738	0,682	0,670	0,669	0,665	0,663
Proposal 2: Without Concat	0,560	0,720	0,679	0,623	0,597	0,583	0,571
Proximity 10 %							
Baseline 1: Xception	0,680	0,726	0,669	0,642	0,636	0,629	0,625
Baseline 2: CeNet	0,840	0,849	0,819	0,799	0,788	0,783	0,776
Proposal 1: With Concat	0,840	0,849	0,819	0,799	0,788	0,783	0,776
Proposal 2: Without Concat	0,840	0,854	0,815	0,759	0,734	0,719	0,707

Tabla 34: Proximity Experiment: mAP@k images without covid seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,560	0,716	0,678	0,637	0,637	0,634	0,632
Baseline 2: CeNet	0,600	0,722	0,705	0,683	0,670	0,665	0,659
Proposal 1: With Concat	0,600	0,722	0,704	0,683	0,671	0,665	0,660
Proposal 2: Without Concat	0,560	0,700	0,660	0,631	0,597	0,580	0,570
Proximity 70 %							
Baseline 1: Xception	0,320	0,557	0,544	0,556	0,562	0,564	0,566
Baseline 2: CeNet	0,360	0,406	0,399	0,328	0,347	0,387	0,403
Proposal 1: With Concat	0,360	0,406	0,397	0,327	0,347	0,387	0,403
Proposal 2: Without Concat	0,560	0,611	0,582	0,571	0,585	0,590	0,597
Proximity 50 %							
Baseline 1: Xception	0,480	0,647	0,636	0,624	0,615	0,606	0,600
Baseline 2: CeNet	0,560	0,721	0,708	0,651	0,625	0,614	0,604
Proposal 1: With Concat	0,560	0,721	0,708	0,651	0,625	0,614	0,604
Proposal 2: Without Concat	0,600	0,712	0,675	0,624	0,606	0,596	0,591
Proximity 30 %							
Baseline 1: Xception	0,560	0,741	0,711	0,678	0,666	0,659	0,653
Baseline 2: CeNet	0,520	0,678	0,660	0,661	0,658	0,652	0,649
Proposal 1: With Concat	0,520	0,678	0,660	0,661	0,658	0,652	0,649
Proposal 2: Without Concat	0,600	0,690	0,672	0,639	0,621	0,610	0,604
Proximity 10 %							
Baseline 1: Xception	0,560	0,746	0,721	0,698	0,686	0,674	0,666
Baseline 2: CeNet	0,720	0,749	0,673	0,651	0,648	0,640	0,634
Proposal 1: With Concat	0,720	0,749	0,673	0,651	0,648	0,640	0,633
Proposal 2: Without Concat	0,640	0,764	0,735	0,704	0,687	0,670	0,657

Tabla 35: Proximity Experiment: mAP@k images without covid seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,600	0,733	0,740	0,734	0,734	0,738	0,742
Baseline 2: CeNet	0,640	0,712	0,653	0,610	0,592	0,581	0,574
Proposal 1: With Concat	0,640	0,712	0,653	0,610	0,592	0,581	0,574
Proposal 2: Without Concat	0,640	0,764	0,722	0,671	0,647	0,629	0,614
Proximity 70 %							
Baseline 1: Xception	0,800	0,824	0,780	0,760	0,747	0,737	0,729
Baseline 2: CeNet	0,360	0,603	0,521	0,449	0,432	0,441	0,450
Proposal 1: With Concat	0,360	0,603	0,521	0,449	0,432	0,441	0,450
Proposal 2: Without Concat	0,600	0,698	0,658	0,605	0,587	0,574	0,566
Proximity 50 %							
Baseline 1: Xception	0,320	0,590	0,585	0,568	0,547	0,539	0,536
Baseline 2: CeNet	0,680	0,802	0,747	0,706	0,689	0,677	0,666
Proposal 1: With Concat	0,680	0,802	0,747	0,706	0,689	0,677	0,666
Proposal 2: Without Concat	0,520	0,679	0,681	0,664	0,651	0,640	0,630
Proximity 30 %							
Baseline 1: Xception	0,680	0,738	0,672	0,643	0,624	0,608	0,601
Baseline 2: CeNet	0,680	0,748	0,693	0,685	0,679	0,674	0,670
Proposal 1: With Concat	0,680	0,748	0,693	0,685	0,679	0,674	0,670
Proposal 2: Without Concat	0,480	0,669	0,684	0,667	0,645	0,628	0,619
Proximity 10 %							
Baseline 1: Xception	0,440	0,635	0,593	0,577	0,576	0,579	0,581
Baseline 2: CeNet	0,240	0,490	0,498	0,505	0,504	0,502	0,502
Proposal 1: With Concat	0,240	0,490	0,499	0,505	0,504	0,503	0,502
Proposal 2: Without Concat	0,640	0,737	0,667	0,625	0,609	0,600	0,595

Tabla 36: Proximity Experiment: mAP@k images without covid seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Proximity 100 %							
Baseline 1: Xception	0,560	0,688	0,634	0,600	0,589	0,580	0,578
Baseline 2: CeNet	0,560	0,672	0,647	0,629	0,622	0,616	0,608
Proposal 1: With Concat	0,560	0,674	0,648	0,630	0,622	0,616	0,608
Proposal 2: Without Concat	0,600	0,738	0,685	0,661	0,648	0,641	0,635
Proximity 70 %							
Baseline 1: Xception	0,600	0,723	0,654	0,626	0,611	0,600	0,592
Baseline 2: CeNet	0,320	0,437	0,432	0,427	0,450	0,467	0,478
Proposal 1: With Concat	0,320	0,437	0,432	0,427	0,450	0,467	0,478
Proposal 2: Without Concat	0,280	0,489	0,445	0,395	0,387	0,397	0,415
Proximity 50 %							
Baseline 1: Xception	0,760	0,789	0,715	0,666	0,646	0,631	0,625
Baseline 2: CeNet	0,040	0,032	0,055	0,128	0,179	0,233	0,268
Proposal 1: With Concat	0,040	0,032	0,055	0,128	0,179	0,233	0,268
Proposal 2: Without Concat	0,000	0,120	0,135	0,140	0,157	0,177	0,216
Proximity 30 %							
Baseline 1: Xception	0,680	0,796	0,762	0,723	0,703	0,693	0,686
Baseline 2: CeNet	0,200	0,379	0,358	0,353	0,359	0,377	0,393
Proposal 1: With Concat	0,200	0,379	0,358	0,353	0,359	0,377	0,393
Proposal 2: Without Concat	0,200	0,458	0,483	0,501	0,510	0,523	0,535
Proximity 10 %							
Baseline 1: Xception	0,520	0,724	0,705	0,704	0,693	0,681	0,670
Baseline 2: CeNet	0,040	0,346	0,345	0,407	0,439	0,464	0,483
Proposal 1: With Concat	0,040	0,346	0,345	0,407	0,439	0,464	0,483
Proposal 2: Without Concat	0,360	0,517	0,470	0,432	0,433	0,439	0,446

Tabla 37: Proximity Experiment: mAP@k images without covid seed 44.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,200	0,399	0,411	0,441	0,440	0,445	0,451
BYOL	0,480	0,619	0,582	0,546	0,535	0,531	0,529
SiamSiam	0,360	0,578	0,554	0,533	0,527	0,525	0,522
SimCLR	0,240	0,447	0,443	0,436	0,443	0,450	0,453
SSL Methods z							
Barlow Twins	0,200	0,392	0,414	0,435	0,438	0,444	0,451
BYOL	0,540	0,656	0,613	0,571	0,549	0,541	0,535
SiamSiam	0,600	0,719	0,642	0,568	0,552	0,546	0,543
SimCLR	0,280	0,459	0,439	0,411	0,408	0,405	0,406

Tabla 38: Proximity Experiment: mAP@k all images seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,340	0,559	0,526	0,495	0,491	0,488	0,487
BYOL	0,460	0,584	0,550	0,496	0,481	0,472	0,468
SiamSiam	0,440	0,591	0,566	0,525	0,505	0,499	0,495
SimCLR	0,360	0,548	0,544	0,528	0,515	0,506	0,503
SSL Methods z							
Barlow Twins	0,380	0,540	0,540	0,530	0,528	0,524	0,521
BYOL	0,420	0,617	0,573	0,528	0,518	0,511	0,508
SiamSiam	0,360	0,555	0,537	0,517	0,510	0,505	0,502
SimCLR	0,360	0,485	0,478	0,466	0,462	0,463	0,463

Tabla 39: Proximity Experiment: mAP@k all images seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,540	0,657	0,599	0,551	0,526	0,510	0,497
BYOL	0,460	0,594	0,590	0,568	0,560	0,558	0,555
SiamSiam	0,380	0,565	0,539	0,524	0,512	0,504	0,502
SimCLR	0,220	0,537	0,543	0,508	0,501	0,496	0,490
SSL Methods z							
Barlow Twins	0,600	0,656	0,621	0,569	0,541	0,526	0,515
BYOL	0,440	0,652	0,612	0,577	0,564	0,555	0,550
SiamSiam	0,440	0,600	0,583	0,541	0,527	0,521	0,518
SimCLR	0,340	0,508	0,509	0,484	0,476	0,472	0,470

Tabla 40: Proximity Experiment: mAP@k all images seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,460	0,555	0,511	0,505	0,503	0,505	0,501
BYOL	0,360	0,536	0,526	0,502	0,492	0,484	0,479
SiamSiam	0,620	0,666	0,627	0,585	0,563	0,547	0,537
SimCLR	0,540	0,668	0,605	0,524	0,501	0,493	0,488
SSL Methods z							
Barlow Twins	0,520	0,640	0,577	0,554	0,546	0,541	0,535
BYOL	0,540	0,628	0,613	0,570	0,554	0,541	0,532
SiamSiam	0,460	0,634	0,608	0,566	0,550	0,541	0,535
SimCLR	0,460	0,635	0,605	0,559	0,532	0,520	0,514

Tabla 41: Proximity Experiment: mAP@k all images seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,420	0,624	0,603	0,585	0,574	0,564	0,559
BYOL	0,480	0,625	0,578	0,556	0,546	0,544	0,540
SiamSiam	0,520	0,646	0,627	0,590	0,579	0,571	0,565
SimCLR	0,240	0,374	0,395	0,389	0,400	0,412	0,422
SSL Methods z							
Barlow Twins	0,540	0,671	0,608	0,584	0,575	0,569	0,562
BYOL	0,540	0,652	0,608	0,565	0,551	0,542	0,537
SiamSiam	0,580	0,684	0,650	0,623	0,598	0,582	0,571
SimCLR	0,400	0,544	0,503	0,473	0,463	0,455	0,450

Tabla 42: Proximity Experiment: mAP@k all images seed 44.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,120	0,255	0,234	0,245	0,230	0,229	0,238
BYOL	0,400	0,529	0,485	0,434	0,412	0,406	0,406
SiamSiam	0,520	0,616	0,571	0,528	0,515	0,509	0,505
SimCLR	0,160	0,343	0,342	0,336	0,345	0,355	0,361
SSL Methods z							
Barlow Twins	0,080	0,241	0,239	0,230	0,223	0,225	0,236
BYOL	0,640	0,734	0,638	0,583	0,554	0,546	0,538
SiamSiam	0,600	0,723	0,635	0,572	0,562	0,559	0,559
SimCLR	0,280	0,453	0,426	0,386	0,375	0,371	0,372

Tabla 43: Proximity Experiment: mAP@k images with covid seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,200	0,522	0,491	0,438	0,431	0,427	0,428
BYOL	0,520	0,616	0,583	0,495	0,465	0,441	0,430
SiamSiam	0,240	0,395	0,377	0,335	0,323	0,332	0,340
SimCLR	0,080	0,332	0,351	0,355	0,354	0,352	0,357
SSL Methods z							
Barlow Twins	0,400	0,572	0,560	0,530	0,521	0,511	0,506
BYOL	0,360	0,574	0,535	0,478	0,472	0,465	0,463
SiamSiam	0,280	0,478	0,478	0,449	0,447	0,443	0,440
SimCLR	0,080	0,274	0,303	0,309	0,310	0,317	0,319

Tabla 44: Proximity Experiment: mAP@k images with covid seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,560	0,655	0,596	0,551	0,524	0,510	0,499
BYOL	0,760	0,768	0,738	0,656	0,616	0,594	0,576
SiamSiam	0,440	0,551	0,524	0,497	0,481	0,470	0,466
SimCLR	0,200	0,477	0,495	0,429	0,404	0,387	0,373
SSL Methods z							
Barlow Twins	0,560	0,617	0,601	0,554	0,536	0,531	0,527
BYOL	0,440	0,656	0,600	0,572	0,565	0,561	0,559
SiamSiam	0,360	0,521	0,520	0,484	0,472	0,468	0,467
SimCLR	0,400	0,524	0,511	0,470	0,453	0,441	0,436

Tabla 45: Proximity Experiment: mAP@k images with covid seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,640	0,682	0,618	0,573	0,540	0,524	0,508
BYOL	0,440	0,537	0,504	0,460	0,441	0,425	0,414
SiamSiam	0,640	0,639	0,585	0,533	0,507	0,488	0,474
SimCLR	0,600	0,684	0,584	0,414	0,350	0,327	0,310
SSL Methods z							
Barlow Twins	0,600	0,740	0,652	0,597	0,569	0,552	0,538
BYOL	0,400	0,535	0,541	0,507	0,493	0,482	0,473
SiamSiam	0,440	0,599	0,568	0,517	0,497	0,489	0,483
SimCLR	0,320	0,505	0,466	0,384	0,345	0,330	0,326

Tabla 46: Proximity Experiment: mAP@k images with covid seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,720	0,757	0,679	0,620	0,584	0,555	0,539
BYOL	0,400	0,536	0,476	0,446	0,435	0,434	0,432
SiamSiam	0,640	0,710	0,671	0,603	0,581	0,563	0,550
SimCLR	0,320	0,492	0,434	0,387	0,376	0,366	0,358
SSL Methods z							
Barlow Twins	0,600	0,718	0,683	0,631	0,606	0,590	0,574
BYOL	0,520	0,671	0,612	0,570	0,546	0,533	0,526
SiamSiam	0,560	0,628	0,590	0,553	0,524	0,504	0,491
SimCLR	0,280	0,424	0,396	0,364	0,352	0,347	0,345

Tabla 47: Proximity Experiment: mAP@k images with covid seed 44.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,280	0,544	0,589	0,636	0,651	0,660	0,665
BYOL	0,560	0,710	0,678	0,658	0,658	0,656	0,653
SiamSiam	0,200	0,539	0,537	0,539	0,540	0,541	0,539
SimCLR	0,320	0,552	0,544	0,537	0,541	0,544	0,545
SSL Methods z							
Barlow Twins	0,320	0,543	0,590	0,639	0,653	0,662	0,666
BYOL	0,440	0,579	0,589	0,560	0,543	0,537	0,532
SiamSiam	0,600	0,715	0,649	0,564	0,541	0,532	0,526
SimCLR	0,280	0,465	0,452	0,436	0,440	0,439	0,440

Tabla 48: Proximity Experiment: mAP@k images without covid seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,480	0,596	0,562	0,553	0,551	0,549	0,547
BYOL	0,400	0,552	0,517	0,496	0,497	0,503	0,506
SiamSiam	0,640	0,787	0,754	0,714	0,688	0,666	0,650
SimCLR	0,640	0,764	0,736	0,702	0,677	0,659	0,648
SSL Methods z							
Barlow Twins	0,360	0,507	0,520	0,529	0,535	0,538	0,536
BYOL	0,480	0,660	0,610	0,578	0,565	0,557	0,553
SiamSiam	0,440	0,631	0,596	0,585	0,573	0,567	0,564
SimCLR	0,640	0,697	0,653	0,624	0,613	0,609	0,606

Tabla 49: Proximity Experiment: mAP@k images without covid seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,520	0,660	0,603	0,552	0,528	0,510	0,495
BYOL	0,160	0,419	0,442	0,480	0,504	0,521	0,534
SiamSiam	0,320	0,578	0,553	0,550	0,543	0,537	0,537
SimCLR	0,240	0,596	0,592	0,587	0,599	0,604	0,606
SSL Methods z							
Barlow Twins	0,640	0,695	0,642	0,584	0,546	0,521	0,504
BYOL	0,440	0,649	0,623	0,582	0,564	0,549	0,541
SiamSiam	0,520	0,679	0,646	0,597	0,581	0,573	0,570
SimCLR	0,280	0,492	0,508	0,497	0,499	0,502	0,505

Tabla 50: Proximity Experiment: mAP@k images without covid seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,280	0,428	0,404	0,437	0,467	0,486	0,494
BYOL	0,280	0,536	0,549	0,544	0,543	0,543	0,544
SiamSiam	0,600	0,694	0,670	0,638	0,619	0,606	0,600
SimCLR	0,480	0,653	0,626	0,634	0,651	0,659	0,665
SSL Methods z							
Barlow Twins	0,440	0,539	0,502	0,512	0,524	0,530	0,532
BYOL	0,680	0,722	0,684	0,632	0,616	0,600	0,592
SiamSiam	0,480	0,668	0,648	0,615	0,604	0,593	0,586
SimCLR	0,600	0,765	0,744	0,734	0,719	0,710	0,701

Tabla 51: Proximity Experiment: mAP@k images without covid seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
SSL Methods h							
Barlow Twins	0,120	0,491	0,526	0,551	0,564	0,573	0,579
BYOL	0,560	0,713	0,679	0,667	0,658	0,653	0,648
SiamSiam	0,400	0,581	0,584	0,576	0,576	0,578	0,581
SimCLR	0,160	0,256	0,356	0,391	0,425	0,458	0,485
SSL Methods z							
Barlow Twins	0,480	0,624	0,532	0,537	0,544	0,547	0,550
BYOL	0,560	0,633	0,604	0,560	0,556	0,551	0,547
SiamSiam	0,600	0,740	0,709	0,692	0,671	0,660	0,652
SimCLR	0,520	0,664	0,611	0,582	0,573	0,563	0,556

Tabla 52: Proximity Experiment: mAP@k images without covid seed 44.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,460	0,668	0,609	0,579	0,544	0,548	0,538

Tabla 53: Proximity Experiment: mAP@k all images seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,460	0,616	0,610	0,549	0,550	0,555	0,527

Tabla 54: Proximity Experiment: mAP@k all images seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,520	0,567	0,588	0,567	0,549	0,578	0,558

Tabla 55: Proximity Experiment: mAP@k all images seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,440	0,707	0,635	0,564	0,540	0,543	0,542

Tabla 56: Proximity Experiment: mAP@k all images seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,380	0,638	0,588	0,585	0,572	0,550	0,540

Tabla 57: Proximity Experiment: mAP@k all images seed 44.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,440	0,668	0,588	0,578	0,534	0,543	0,509

Tabla 58: Proximity Experiment: mAP@k images with covid seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,400	0,570	0,586	0,537	0,527	0,530	0,495

Tabla 59: Proximity Experiment: mAP@k images with covid seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,520	0,512	0,593	0,523	0,520	0,542	0,519

Tabla 60: Proximity Experiment: mAP@k images with covid seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,480	0,656	0,638	0,536	0,538	0,525	0,512

Tabla 61: Proximity Experiment: mAP@k images with covid seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,320	0,695	0,566	0,581	0,543	0,498	0,518

Tabla 62: Proximity Experiment: mAP@k images with covid seed 44.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,480	0,668	0,630	0,579	0,553	0,552	0,567

Tabla 63: Proximity Experiment: mAP@k images without covid seed 13.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,520	0,662	0,633	0,560	0,572	0,579	0,559

Tabla 64: Proximity Experiment: mAP@k images without covid seed 134.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,520	0,622	0,582	0,610	0,578	0,613	0,597

Tabla 65: Proximity Experiment: mAP@k images without covid seed 2023.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,400	0,758	0,631	0,591	0,542	0,561	0,571

Tabla 66: Proximity Experiment: mAP@k images without covid seed 233.

Fuente: Elaboración Propia.

	mAP@1	mAP@5	mAP@10	mAP@20	mAP@30	mAP@40	mAP@50
Random							
Sample	0,440	0,580	0,609	0,588	0,600	0,602	0,562

Tabla 67: Proximity Experiment: mAP@k images without covid seed 44.

Fuente: Elaboración Propia.

```
import random
from typing import Callable, Tuple
from kornia import augmentation as aug
from kornia import filters
from kornia.geometry import transform as tf
import torch
from torch import nn, Tensor

class RandomApply(nn.Module):
    def __init__(self, fn: Callable, p: float):
        """
        Random Apply

        p: Probability of apply the transformation
        """
        super().__init__()
        self.fn = fn
        self.p = p

    def forward(self, x: Tensor) -> Tensor:
        return x if random.random() > self.p else self.fn(x)

def ssl_augmentation(image_size: Tuple[int, int] = (224, 224)) -> nn.Module:

    return nn.Sequential(
        tf.Resize(size=image_size),
        RandomApply(aug.ColorJitter(
            brightness=0.8, contrast=0.8, saturation=0.8, hue=0.2), p=0.8),
        aug.RandomGrayscale(p=0.2),
        aug.RandomHorizontalFlip(p=0.5),
        RandomApply(filters.GaussianBlur2d(
            kernel_size=(3, 3), sigma=(1.5, 1.5)), p=0.1),
        aug.RandomResizedCrop(size=image_size),
        # Imagenet stats
        aug.Normalize(
            mean=torch.tensor([0.485, 0.456, 0.406]),
            std=torch.tensor([0.229, 0.224, 0.225]),
        )
    )
```

REFERENCIAS BIBLIOGRÁFICAS

- [1961] Barlow, Horace B and others (1961). Possible principles underlying the transformation of sensory messages. *Sensory communication*, 1(01):217–233.
- [Bengio et al., 2000] Bengio, Y., Ducharme, R., y Vincent, P. (2000). A neural probabilistic language model. *Advances in neural information processing systems*, 13.
- [Chen et al., 2020] Chen, T., Kornblith, S., Norouzi, M., y Hinton, G. (2020). A simple framework for contrastive learning of visual representations. En *International conference on machine learning*, pp. 1597–1607. PMLR.
- [Chen y He, 2021] Chen, X. y He, K. (2021). Exploring simple siamese representation learning. En *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758.
- [Chollet, 2017] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258.
- [Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., y Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. En *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee.
- [Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., y Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- [Dosovitskiy et al., 2020] Dosovitskiy, Alexey and Beyer, Lucas and Kolesnikov, Alexander and Weissenborn, Dirk and Zhai, Xiaohua and Unterthiner, Thomas and Dehghani, Mostafa and Minderer, Matthias and Heigold, Georg and Gelly, Sylvain and others (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.
- [Dumoulin y Visin, 2016] Dumoulin, V. y Visin, F. (2016). A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*.
- [Grill et al., 2020] Grill, Jean-Bastien and Strub, Florian and Altché, Florent and Tallec, Corentin and Richemond, Pierre and Buchatskaya, Elena and Doersch, Carl and Avila Pires, Bernardo and Guo, Zhaohan and Gheshlaghi Azar, Mohammad and others (2020). Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284.
- [Gu et al., 2019] Gu, Z., Cheng, J., Fu, H., Zhou, K., Hao, H., Zhao, Y., Zhang, T., Gao, S., y Liu, J. (2019). Ce-net: Context encoder network for 2d medical image segmentation. *IEEE transactions on medical imaging*, 38(10):2281–2292.

- [He et al., 2022] He, K., Chen, X., Xie, S., Li, Y., Dollár, P., y Girshick, R. (2022). Masked autoencoders are scalable vision learners. En *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009.
- [He et al., 2016] He, K., Zhang, X., Ren, S., y Sun, J. (2016). Deep residual learning for image recognition. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.
- [Krizhevsky et al., 2017] Krizhevsky, A., Sutskever, I., y Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.
- [LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., y Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- [Li et al., 2018] Li, Z., Zhang, X., Müller, H., y Zhang, S. (2018). Large-scale retrieval for medical image analytics: A comprehensive review. *Medical image analysis*, 43:66–84.
- [Liu et al., 2016] Liu, Z., Luo, P., Qiu, S., Wang, X., y Tang, X. (2016). Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1096–1104.
- [Lowe, 1999] Lowe, D. G. (1999). Object recognition from local scale-invariant features. En *Proceedings of the seventh IEEE international conference on computer vision*, volumen 2, pp. 1150–1157. Ieee.
- [McCulloch y Pitts, 1943] McCulloch, W. S. y Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.
- [Simonyan y Zisserman, 2014] Simonyan, K. y Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- [Soares et al., 2020] Soares, E., Angelov, P., Biaso, S., Froes, M. H., y Abe, D. K. (2020). Sars-cov-2 ct-scan dataset: A large dataset of real patients ct scans for sars-cov-2 identification. *MedRxiv*.
- [Sotomayor et al., 2021] Sotomayor, C. G., Mendoza, M., Castañeda, V., Farías, H., Molina, G., Pereira, G., Härtel, S., Solar, M., y Araya, M. (2021). Content-based medical image retrieval and intelligent interactive visual browser for medical education, research and care. *Diagnostics*, 11(8):1470.
- [Szegedy et al., 2015] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., y Rabinovich, A. (2015). Going deeper with convolutions. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.
- [Xie et al., 2017] Xie, S., Girshick, R., Dollár, P., Tu, Z., y He, K. (2017). Aggregated residual transformations for deep neural networks. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500.

- [Ye *et al.*, 2021] Ye, M., Shen, J., Lin, G., Xiang, T., Shao, L., y Hoi, S. C. (2021). Deep learning for person re-identification: A survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [Zbontar *et al.*, 2021] Zbontar, J., Jing, L., Misra, I., LeCun, Y., y Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. En *International Conference on Machine Learning*, pp. 12310–12320. PMLR.
- [Zhao *et al.*, 2020] Zhao, J., Zhang, Y., He, X., y Xie, P. (2020). Covid-ct-dataset: a ct scan dataset about covid-19. *arXiv preprint arXiv:2003.13865*.