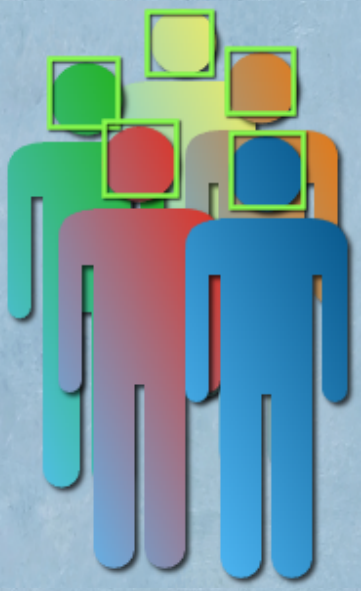
Biometría Facial con Deep Learning

Reconocimiento Facial

con Redes Neuronales Convolucionales



Universitat Oberta de Catalunya

Manuel Muñoz Plá

TFM Análisis de Datos

Máster Universitario en Ciberseguridad y Privacidad

Biometría Facial con Deep Learning

Manuel Muñoz Plá

Máster Universitario en Ciberseguridad y Privacidad
Análisis de datos
Enric Hernández Jiménez
Andreu Pere Isern Deyà

Enero 2023

Esta obra está sujeta a una licencia de
Reconocimiento-NoComercial-CompartirIgual
3.0 España de Creative Commons

Copyright © 2023 Manuel Muñoz Plá.
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU Free Documentation License".

© Manuel Muñoz Plá
Reservados todos los derechos. Está prohibido la reproducción total o parcial de esta obra por cualquier medio o procedimiento, comprendidos la impresión, la reprografía, el microfilme, el tratamiento informático o cualquier otro sistema, así como la distribución de ejemplares mediante alquiler y préstamo, sin la autorización escrita del autor o de los límites que autorice la Ley de Propiedad Intelectual.

Agradecimientos:

*A **Concha***
por su tiempo y sin cuyo apoyo
este trabajo no habría sido posible
MHQ

*A **Patricia** y **Daniel***
por su tiempo y paciencia

*A **Remedios** y **Manuel***

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Biometría Facial con Deep Learning*
Nombre del autor:	*Manuel Muñoz Plá*
Nombre del consultor/a:	*Enric Hernández Jiménez*
Nombre del PRA:	*Andreu Pere Isern Deyà*
Fecha de entrega (mm/aaaa):	*Enero 2023*
Titulación o programa:	Máster Universitario en Ciberseguridad y Privacidad
Área del Trabajo Final:	*Análisis de Datos*
Idioma del trabajo:	*Castellano*
Palabras clave	*Biometría, Facial, Reconocimiento, Deep Learning, Aprendizaje Profundo, Redes Neuronales Convolucionales, CNN, ConvNet*

Resumen del Trabajo

La biometría facial es una tecnología capaz de recabar los datos biométricos de un rostro. Aunque tanto la biometría como las redes neuronales tienen sus raíces teóricas en la mitad del siglo XX, su implementación y uso generalizado no ha sido posible hasta hace unos años. El notable incremento en la potencia computacional y la gran profusión de datos disponibles en la actualidad han facilitado el uso generalizado de las redes neuronales.

La biometría facial necesita medir las características fisiológicas para diferenciar un rostro del resto de la imagen. Esto supone la recopilación de una gran cantidad de características (datos). El análisis de grandes cantidades de datos, que en el caso de imágenes son datos no estructurados, no es una tarea fácil. En la última década el aprendizaje profundo (DL - Deep Learning) ha dado un vuelco al análisis de imágenes, donde el rendimiento de las redes neuronales convolucionales (ConvNet - Convolutional Neural Networks) ha superado a todas las técnicas previas, cosechando grandes éxitos en la detección y diferenciación de objetos en una imagen. Por todo esto, las ConvNet se han puesto a la cabeza en el tratamiento de datos con imágenes.

Una vez encontrado un rostro, mediante la biometría facial, además de detectar y/o reconocer rostro/s humano/s en una imagen o video, se puede procesar mucha información asociada: identificación, emociones, seguimiento en video, reconstrucciones faciales, y muchas otras. Este trabajo se centra en el Reconocimiento Facial porque reúne la mayoría de los procesos y tratamiento de datos que se necesitan en la biometría facial. Se realiza una revisión del ‘Estado del Arte en el Reconocimiento Facial’ y una prueba de concepto: ‘Detección y Reconocimiento Facial a través de una webcam’.

Abstract

Facial biometrics is a technology capable of collecting biometric data from a face. Although both biometrics and neural networks have their theoretical roots in the mid-20th century, their widespread implementation and use have only been possible in recent years. The remarkable increase in computational power and the wealth of data now available have facilitated the widespread use of neural networks.

Facial biometrics needs to measure physiological characteristics to differentiate a face from the rest of the image. This involves the collection of a large number of features (data). Analyzing large amounts of data, which in the case of images are unstructured data, is not an easy task. In the last decade, Deep Learning (DL) has turned image analysis on its head, where the performance of Convolutional Neural Networks (ConvNet) has surpassed all previous techniques, achieving great success in the detection and differentiation of objects in an image. For all these reasons, ConvNets have taken the lead in the image data processing.

Once a face has been found, through facial biometrics, in addition to detecting and/or recognizing human face/s in an image or video, a lot of associated information can be processed: identification, emotions, video tracking, facial reconstructions, and many others. This paper focuses on Facial Recognition because it brings together most of the processes and data processing needed in facial biometrics. A review of the 'State of the Art in Facial Recognition' and a proof of concept: 'Facial Detection and Recognition through a webcam' is performed.

Indice

1 - Introducción 1
1.1 - Contexto y justificación del Trabajo 1
Figura 1. Reconocimiento facial y privacidad. 1
Figura 2. Caras Propias - Eigenfaces. 2
Figura 3. Trazo del seguimiento del rostro en una imagen. 2
Figura 4. Capas intermedias y final - ConvNet. 4
Figura 5. Potencia Computacional. 4
1.2 - Objetivos y alcance del TFM 5
Figura 6. Etapas del reconocimiento facial. 5
1.3 - Impacto en sostenibilidad, ético-social y de diversidad 6
1.3.1 - Impactos Positivos 6
1.3.2 - Impactos Negativos 7
1.3.3 - Noticias relacionadas con el uso del reconocimiento facial: 7
1.4 - Enfoque y método seguido 8
1.4.1 - KDD 8
1.4.2 - SEMMA 8
1.4.3 - CRISP-DM 9
Figura 7. Diagrama CRISP-DM. 9
1.5 - Planificación del Trabajo 10
1.5.1 - PEC1 del 28/09/22 al 11/10/22 - 10 dias 10
1.5.2 - PEC2 del 12/10/22 al 08/11/22 - 20 días 10
1.5.3 - PEC3 del 09/11/22 al 06/12/22 - 20 días 10
1.5.4 - PEC4 del 07/11/22 al 10/01/23 - 25 días 10
1.5.5 - Elaboración y presentación de Video 11/01/23 al 17/01/23 - 5 dias 10
1.5.6 - Defensa del TFM del 23/01/23 a 27/01/23 - 5 dias 10
Figura 8. Gantt del TFM. 11

2 - Redes Neuronales 12
2.1 - Introducción. Redes neuronales 12
2.1.1 - Organización por capas 12
Figura 9. Relación entre AI - ML - DL. 12
Figura 10. Esquema Neurona. 13
Figura 11. Neurona artificial. 13
Figura 12. Función sigmoidea. 14
2.2 - Funciones de activación habituales 14
2.2.1 - Función Lineal 14
Figura 13. Función lineal. 14
2.2.2 - Función Umbral 14
Figura 14. Función umbral. 14
2.2.3 - Función Sigmoidea o logística. 15
Figura 15. Función sigmoidea. 15
2.2.4 - Función Tangencial Hiperbólica Tanh 15
Figura 16. Función tanh. 15
2.2.5 - Funcion ReLu 15
Figura 17. Función ReLu. 15
Figura 18. Dropout. 16
2.2.6 - ReLU con fugas. Leaky ReLu 16
Figura 19. Función Leaky ReLu. 16
2.2.7 - PReLu. Unidad Lineal Rectificada Paramétrica 16
Figura 20. Funcion PReLu. 16
2.2.8 - Función Softmax 16
Figura 21. Función softmax. 17

2.3 - Entrenamiento. Propagación / Retropropagación 17
Figura 22. Propagación. 17
2.4 - Redes Neuronales Convolucionales ConvNet 19
Figura 23. Filtro Sobel. 19
2.4.1 - Convoluciones de imágenes con más de una dimensión. 19
Figura 24. Convolución. 20
Figura 25. Capas RGB. 20
Figura 26: Red Convolucional Genérica. 21
2.5 - Capas Arquitectura Convolucional 22
2.5.1 - Capa convolucional. Convolutional layer 22
Figura 27. Ejemplo de convolución. 22
2.5.1.1 - Padding o relleno 23
Figura 28. Ejemplo de padding. 23
2.5.2 - Capa de agrupación. Pooling layer 23
Figura 29. Reducción Max Pooling. 23
Figura 30. Grupos de capas Convolución - Pooling. 24
2.5.2.1 - Max Pooling 24
Figura 31. Ejemplo de Max Pooling. 24
2.5.2.2 - Average Pooling 24
Figura 32. Ejemplo de Average Pooling. 24
Figura 33. Max Pooling durante la retropropagación. 25
2.5.4 - Capa ReLu 25
2.5.5 - Capa completamente conectada. Fully connected layer 25
2.5.5 - Capa de pérdida. Loss layer 26
Figura 34. Aspecto final de una ConvNet estándar. 26
2.5.6 Reglas generales para modelar una red convolucional 26
2.6 - Regularización 26
2.6.1 - Dropout. 26
2.6.2 - DropConnect. 27
2.6.3 - Agrupación estocástica. 27
2.6.4 - Decadencia de peso. 27
2.7 - Arquitecturas ConvNet más conocidas 28
2.7.1 - LeNet [33] 1998 28
Figura 35. LeNet. 28
2.7.2 - AlexNet [34] 2012 28
Figura 36. AlexNet. 28
2.7.3 - GoogLeNet [36] 2014 29
2.7.3.1 - El módulo inception 29
Figura 37. Módulos Inception. 29
Figura 38. Estructura GoogLeNet. 29
2.7.4 - VGGNet [35] 2014 29
Figura 39. VGGNet16. 30
2.7.5 - ResNet [38] 2015 30
Figura 40. Bloque residual. 31
2.7.6 - SeNet [103] 2017 32
Figura 41. Módulo Residual vs Módulo SE-ResNet. 32
2.8 - Detección de objetos y segmentación con ConvNet 32
Figura 42. Detección de rostros y sonrisa. 33
3 - Estado de Arte 33
3.1 - Introducción. Métodos de extracción de características 34
Figura 43. Puntos de referencia. EBGM. 34
Figura 44. Descripción de la cara LBP con tres niveles de información. 35
3.2 - Reconocimento Facial con Deep Learning 35
3.3 - Estado del Arte en la última década 38
3.3.1 - Estado del Arte 2014 38
Figura 45. DeepFace. 38
Figura 46. Arquitectura de DeepID2. 39
3.3.2 - Estado del Arte 2015 39

Figura 47. FaceNet. 40
3.3.3 - Estado del Arte 2016 40
Figura 48. Variaciones Faciales 40
Figura 49. Gráficas Tail. 41
Figura 50. Center Loss. 41
3.3.4 - Estado del Arte 2017 41
Figura 51. L2-Softmax. 42
Figura 52: vMF-FL. 42
3.3.5 - Estado del Arte 2018 42
Figura 53: A-Softmax. 43
Figura 54. LMCL de CosFace. 43
Figura 55: AM-Softmax. 43
Figura 56. Comparativa Relu / Maxout / MFM. 44
3.3.6 - Estado del Arte 2019 44
Figura 57:Training ArcFace. 44
Figura 58. ArcFace. Modelo discriminativo / generativo. 45
Figura 59. Clases subrepresentadas. 45
Figura 60. Pairwise Differential Siamese Network. 45
3.3.7 - Estado del Arte 2020 46
Figura 60. RCM Loss. 46
Figura 61: Límites de decisión. Funciones de Pérdida habituales 47
3.3.8 - Estado del Arte 2021 47
Figura 62. Group-aware. 47
Figura 63. GroupFace. 48
Figura 64. Dyn-arcFace loss. 48
3.3.9 - Estado del Arte 2022 48
Figura 65. Decision boundary. Fontera de Desicion. 49
Figura 66 Softmax (a) sin y (b) con Orthant Loss. 49
Figura 67: InterFace. Comparativa. 49
Tabla. Estado del Arte en el Reconocimiento Facial con ConvNet desde 2014 50

4 - Prueba de concepto 52
4.1 - Introducción 52
Figura 68. Etapas del Reconocimiento Facial. 52
Figura 69: Arriba, DeepFace. Abajo, Keras de DeepFace 53
4.2 - Entorno de desarrollo y herramientas 53
● Hardware 53
Figura 70: Captura pantalla Driver Nvidia RTX 2060. 54
Figura 71: Webcam PS3 utilizada. 54
● Software: Entorno y Herramientas 54
Figura 72: Captura pantalla. Versiones de tensorflow y keras. 56
4.3 - Prueba de Concepto. 56
4.3.1 - Reconocimiento facial Webcam con ConvNet 56
Figura 73: Captura de pantalla. Entorno env UOC. 56
4.3.2 - VGGFace2 Dataset 56
Figura 74: Distribuciones del dataset VGGFace2. 57
4.3.3 - DeepFace 57
Figura 75: Modelo DeepFace de 9 capas CNN-9. 57
Figura 76. De izquierda a derecha, Maria Valverde y Vanesa Romero. 58
Figura 77. Rotación para alinear los rostros. 58
Figura 78: Imágenes superpuestas. Izquierda sin alinear. Derecha alineadas. 59
4.4 - Jupyter Notebook. RF Webcam 60
4.4.1 - Importación Librerías 60
4.4.2 - Filtro de detección de rostros 60
Figura 79. Captura Jupyter Notebook. 60
4.4.3 - DeepFace con Keras 60
Figura 80. Captura de pantalla. DeepFace en Keras. 60
Figura 81. Pesos para DeepFace. Dataset VGGFace2. 61
4.4.4 - Carga de pesos 61

Figura 82. Esquema de carga de pesos a DeepFace. 61
Figura 83. Captura Notebook de carga de pesos a DeepFace. 62
4.4.5 - Matriculación de Rostros 62
Figura 84. Esquema de carga de imágenes para matricular a DeepFace. 62
Figura 85. Captura Notebook del proceso de matriculación con DeepFace. 63
4.4.6 - Reconocimiento WEBCAM 63
Figura 86. Diagrama del funcionamiento del reconocimiento facial con DeepFace. 63
4.4.7 - Prueba de reconocimiento facial en tiempo real 64
Figura 87: Capturas WEB. Manuel, Michael Palin y Emma Watson. 64

5 - Conclusiones 64

Figura 93: Captura WEB. Emma Watson. 66

6 - Líneas futuras 66

Figura 94. Comentario de Hinton. 67
Figura 95. Posiciones Faciales Relativas. 68
Figura 96: Figura con distintas vistas. 68
Figura 97: Diferentes vistas de la Estatua de la Libertad. 69

7 - Bibliografía 70

1 - Introducción

1.1 - Contexto y justificación del Trabajo

Los sistemas biométricos basan su funcionamiento en la adquisición de características de distintas partes del cuerpo humano como: los ojos, la retina, el iris, la palma de la mano, las huellas dactilares,... o del rostro completo: ancho de la cara, espacio de pupila a pupila, distancia entre la frente y la barbilla,... entre otras medidas posibles. Cada individuo posee una **métrica única**, unas características físicas (incluso de comportamiento, como la forma de andar) propias y distintivas en cada ser humano.

La biometría facial se centra en los datos que se pueden extraer del rostro. Juega un papel importante en la comunicación visual porque se puede extraer una gran cantidad de información. Al margen de la identidad se pueden estudiar emociones, e incluso a partir de ella intentar deducir intenciones. A partir de los datos recopilados y etiquetados, el tratamiento de la información es muy amplio y variado, según el entorno donde se aplique. Actualmente hay una gran preocupación en el tratamiento de estos datos. Toca de lleno temas tan sensibles como la privacidad [17] Múltiples empresas y organismos públicos que usan el reconocimiento facial se encuentran en la actualidad en procesos de regulación e incluso prohibición de sus actividades relacionadas con este campo.

Figura 1. Reconocimiento facial y privacidad. Fuente: [5]

La biometría facial no es un estudio reciente, a principios de la década de 1960, Bledsoe [6] desarrolló un sistema de clasificación manual para reconocer rostros usando los atributos faciales únicos de un individuo. Bledsoe marcaba manualmente características faciales mediante las coordenadas horizontales y verticales de los ojos, la nariz, la boca y la línea del cabello en la fotografía de una persona. Las métricas obtenidas del rostro se introducían y almacenaban en una tableta RAND [7], una precursora de las actuales tablets. Fue de los primeros dispositivos que tenían la capacidad de ingresar datos con un lápiz óptico para digitalizar la información. Una vez que creada la base de datos, Bledsoe introducía manualmente las métricas faciales de una persona, y el sistema proporcionaba la coincidencia más cercana.

Los estudios sobre el reconocimiento facial evolucionaron a fines de la década de 1980 gracias a los estudios de Sirovich y Kirby [8] para mejorar la precisión. Aplicaron técnicas para extraer las

características básicas de una imagen mediante el análisis de la imagen, mientras conservaban la mayor cantidad posible del conjunto de datos original. El objetivo era desarrollar un algoritmo que se ajustara a la variación de la luz y la expresión. Dio como resultado lo que denominan "Caras Propias - Eigenfaces", un conjunto de vectores propios derivados, imágenes faciales con dimensiones más bajas, lo que da como resultado una representación borrosa de una cara.

Figura 2. Caras Propias - Eigenfaces. Fuente: [8]

Aunque la imagen de la "Cara Propia" es borrosa, características como los ojos, la nariz y el contorno de la cara, son reconocibles. Sirovich y Kirby (1987) [8] demostraron que el análisis de componentes principales PCA [9] se puede utilizar en una colección de imágenes faciales para formar un conjunto de características básicas. Estas imágenes "Caras Propias", pueden combinarse linealmente para reconstruir imágenes en el conjunto de entrenamiento original.
En 1991 Turk y Pentland [10] detectaban el rostro dentro una imagen haciendo coincidir distancias y proporciones, usando el método estadístico de análisis de componentes principales (PCA) Incluso realizaban el seguimiento del rostro en una imagen.

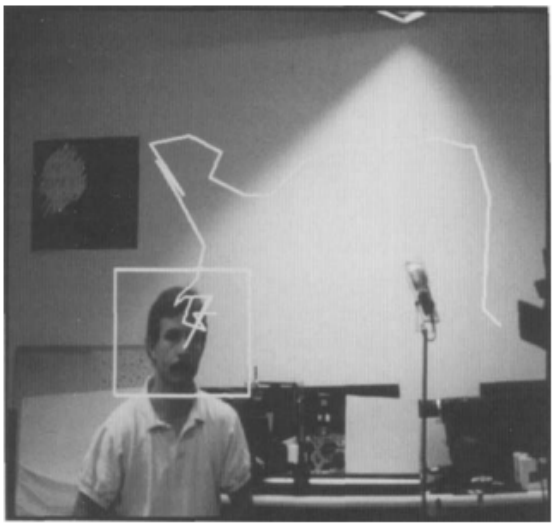


Figura 3. Trazo del seguimiento del rostro en una imagen. Fuente: [10]

Matthew Turk y Alex Pentland en 1991 [10] ya recomendaban el uso de redes neuronales para el reconocimiento facial:

"... The significant features are known as "eigenfaces, " because they are the eigenvectors (principal components) of the set of faces; they do not necessarily correspond to features such as eyes, ears, and noses. The projection operation characterizes an individual face by a weighted sum of the eigenface features, and so to recognize a particular face it is necessary only to compare these weights to those of known individuals. Some particular advantages of our approach are that it provides for the ability to learn and later recognize new faces in an unsupervised manner, and that it is easy to ***implement using a neural network architecture****"*

Reproducción de la parte final del Abstract del articulo: *Turk M, Pentland A. Eigenfaces for recognition. J Cogn Neurosci. 1991 Winter;3(1):71-86. doi: 10.1162/jocn.1991.3.1.71. PMID: 23964806* [10]

En 1998 la agencia Darpa presenta FERET (Face Recognition Technology), un programa para el desarrollo del reconocimiento facial junto a un dataset de 2400 imágenes de 850 personas [92]. En 2005 nace FRGC [93] (Face Recognition Grand Challenge), un concurso-desafío para el fomento y desarrollo del reconocimiento facial. Desde 1991 a 2012 predominaron los métodos estadísticos globales, destacando el estudio de componentes (PCA)
.
En 2012 Krizhevsky [14] gana el concurso ILSVRC-2012 con AlexNet y demuestra el potencial de las redes neuronales. Se produce un cambio en el enfoque gracias al aprendizaje profundo, pasando a buscar características mediante redes neuronales en el rostro previamente detectado. El tamaño y la calidad del conjunto de datos se muestra esencial, pues cuantas más imágenes mejor aprende el sistema.

Las redes neuronales, al igual que la biometría facial, también tienen bastante tiempo teórico (Warren McCulloch[11] / Walter Pitts[12] - 1943). Existen muchos tipos de redes neuronales, aunque las más relevantes para el reconocimiento facial son las redes neuronales convolucionales (ConvNet)[1-3]. Las ConvNet son muy eficientes para clasificar imágenes. Estas redes utilizan una arquitectura inspirada en datos biológicos tomados de experimentos fisiológicos realizados en la corteza visual animal.

El sistema visual humano está organizado en diferentes capas. Los ojos están conectados a un área del cerebro llamada Visual Cortex (V1). Este área tiene la función de discriminar propiedades básicas como pequeños cambios en la orientación visual, frecuencias espaciales y colores. Se estima que V1 consta de alrededor de 140 millones de neuronas, con millones de conexiones entre ellas. Luego, V1 se conecta a otras áreas V2, V3, V4, V5, V6 y realiza un procesamiento de imágenes cada vez más complejo y el reconocimiento de conceptos más sofisticados, como formas, rostros, animales, objetos,...

En este artículo de 2009, *H. Lee, R. Grosse, R. Ranganath, and A.Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Proceedings of the 26th Annual International Conference* [13], se muestra como el aprendizaje profundo copia de forma análoga esta organización basada en capas del sistema visual humano: las primeras capas de neuronas artificiales aprenden las propiedades básicas de las imágenes, mientras que las capas más profundas aprenden conceptos más sofisticados.

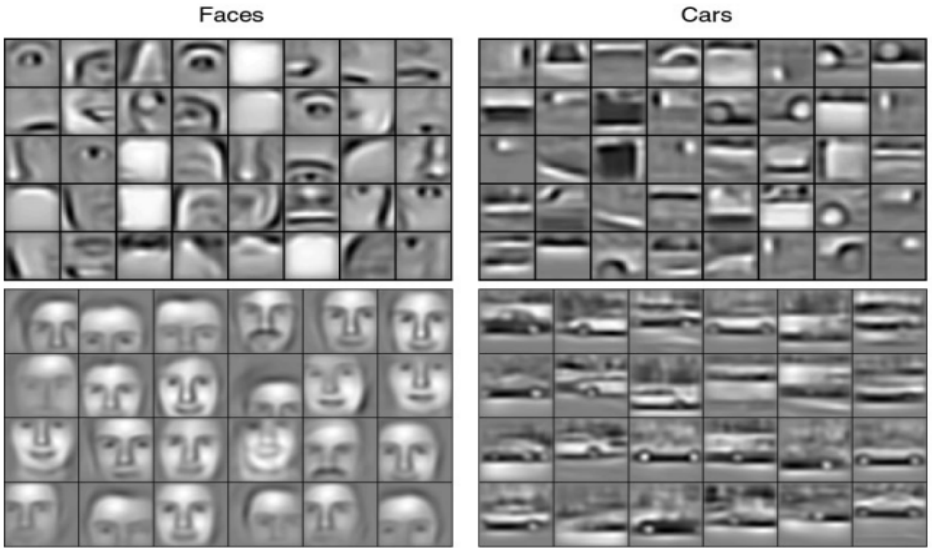


Figura 4. Capas intermedias y final - ConvNet. Fuente: [13]

En 2012 la confluencia de varias circunstancias supuso un punto de inflexión en la redes neuronales, y dio comienzo a la Era Moderna en el Deep Learning:

1. Un innovador algoritmo de aprendizaje rápido denominado Alexnet. El modelo fue propuesto por A. Krizhevsky, I. Sutskever y G.E. Hinton en el trabajo de investigación Imagenet Classification with Deep Convolution Neural Network [14] y fue el punto de inflexión de la IA.
2. Incremento exponencial de la potencia computacional [15] gracias la introducción de las GPU (Graphics Processing Unit) alrededor de 2011 para computación. En la actualidad además de las GPU se suman también las TPU (Tensor Processing Unit) y la computación en la nube. Actualmente es posible contratar casi cualquier nivel de computación necesario a costes relativamente bajos.
3. La disponibilidad de grandes colecciones de datos. La era digital ha generado una gran cantidad de datos: redes sociales, conexión masiva de dispositivos a internet, entre otras.

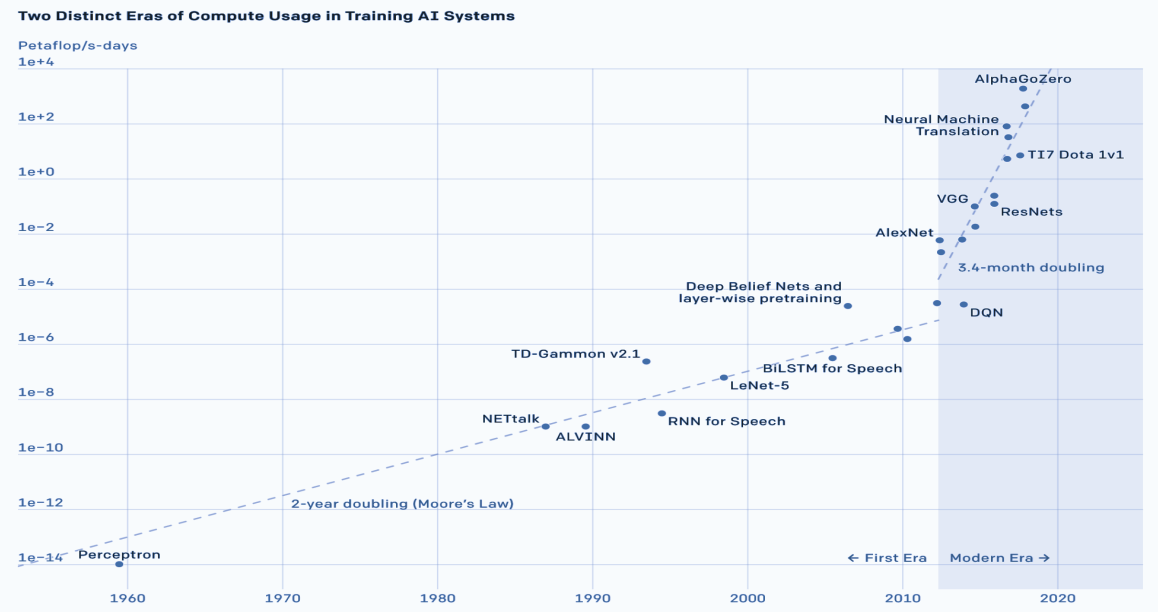


Figura 5. Potencia Computacional. Fuente: [15]

En la gráfica vemos claramente dos eras distintas de entrenamiento de sistemas de IA en términos de uso de cómputo:

- Una primera era desde 1959 a 2012, la energía requerida se duplicaba cada dos años. Siguen aproximadamente la ley de Moore.
- La era moderna, desde 2012 hasta ahora, se duplica cada 3,4 meses

Respecto al reconocimiento facial, en 2014 Taigman et al. [45] presentaron Deepface, un modelo convolucional de 9 capas sobre un dataset de 4 millones de imágenes (4.4M, 4K). Afirmaron que su propuesta se acercaba al rendimiento del ojo humano: Deepface 97,35% frente al ojo humano 97,53%.
Curiosamente en 2021 Meta [90] (Facebook) anunció el borrado de más de mil millones de datos faciales recopilados durante estos años ante la presión de los debates sobre la vulneración de la privacidad.

1.2 - Objetivos y alcance del TFM

El proceso de reconocimiento facial comienza con la adquisición de la imagen. La detección aísla un área de píxeles que corresponden a la/s cara/s en la imagen. Se extraen las características y finalmente se procede a la identificación:

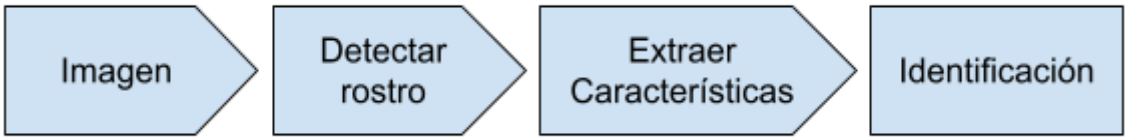


Figura 6. Etapas del reconocimiento facial. Fuente: propia

La captura de datos faciales para el reconocimiento presenta ciertas dificultades que pueden alterar la capacidad de obtener una captura de calidad, necesaria para para el reconocimiento facial:

- Iluminación inadecuada que puede causar sombras excesivas.
- Expresiones faciales, como sonreír, cerrar los ojos o fruncir el ceño.
- Artefactos como gafas, sombreros, bufandas, joyas, o simplemente una mano sobre el rostro en hábitos habituales como pueda ser un rascado o tapar los ojos por un deslumbramiento, pueden afectar la captura porque obstruyen parcialmente la medición.
- La presencia de ruido, como distorsión visual o decoloración con manchas (intencionada o accidental).
- Escala. Ocurren cuando la persona está más cerca o más lejos en la imagen.
- Pose y/o distintos ángulos del rostro.
- Varias imágenes de la misma persona, con variaciones de los puntos anteriores.

De estos puntos se deduce la importancia de un buen conjunto de datos sobre los que trabajar.
Esos dataset se podrían clasificar:

- Conjunto de datos de **uno a muchos**. Con distintas técnicas, partiendo de una imagen se crean un número variado de imágenes de la misma persona con distintas poses,

iluminaciones, artefactos, expresiones faciales, escala. Este tipo de tratamiento genera muchos parches.

- Conjunto de datos de **muchos a uno**. Es la situación inversa a la anterior, se cuenta con más de una imágen de la misma persona con distintas variaciones de pose, iluminación, entre otras. Se pueden utilizar técnicas para complementar e igualar el número de muestras por individuo (técnicas 3d, últimamente GAN, entre otras). Con estos tratamiento se recupera la vista canónica y es posible realizar el reconocimiento como si se tratase de un dataset en condiciones controladas.

En el reconocimiento facial no se trata con la red neuronal toda la imagen, solo se usa como entrada el área seleccionada durante la detección facial, una pequeña área de la imagen de algunos cientos de pixel de lado, 152x152, por ejemplo. Por tanto, al aplicar los puntos previos a un dataset se podría considerar como un pretratamiento de los datos.
En la actualidad hay una diferencia abismal entre los dataset públicos, que contiene varios miles de personas distintas, con los privados, como los de Facebook o Google [52], con varios millones de personas.

Una revisión del Estado del Arte nos mostrará la enorme variedad que existe para la detección y reconocimiento facial.
Aunque se podría partir de un dataset propio y entrenarlo, el posible tamaño del mismo no es comparable con los existentes. Existen muchos modelos preentrenados que usaremos para la prueba de concepto (POC). La POC de este trabajo nos permitirá a partir de una imagen webcam en tiempo real detectar los rostros que hay en ella e identificarlos previa matriculación.

Los objetivos y alcance del TFM:

- Revisión del Estado del Arte en el Reconocimiento Facial con Deep Learning.
    - Datasets y algoritmos existentes
- Diseñar e implementar una POC Prueba de Concepto sobre un dataset con un algoritmo ConvNet y comprobar que detecta e identifica un rostro en una imagen o video.
    - Dataset y algoritmo seleccionado para este trabajo
    - Modelo de detección de rostro en una imagen
    - Modelo de reconocimiento de rostros matriculados

1.3 - Impacto en sostenibilidad, ético-social y de diversidad

Reconocimiento facial como con cualquier tecnología tiene usos adecuados e inadecuados

1.3.1 - Impactos Positivos

- No es intrusivo, se puede realizar a distancia, sin tener que tocar un sensor como, por ejemplo, en la biometría de huella digital

- Reconocimiento facial es excelente para la vigilancia y el cumplimiento de la ley (con las garantías legislativas vigentes)
- Con el sustento legal adecuado, se puede usar en una variedad de situaciones y puede operar de forma encubierta, sin la cooperación del individuo. La tecnología puede ayudar a localizar niños desaparecidos y/o explotados, pacientes desorientados, salas UCI en hospitales, colegios de atención especial, entre otras muchas.
- No necesita grandes requisitos hardware. Casi cualquier tipo de cámara puede ser usada, por tanto puede ser una solución económica para la identificación y la autenticación
- Verificaciones a través del reconocimiento facial. Por ejemplo, detección del uso máscara durante una crisis médica, como una epidemia o toma de temperaturas (asociadas a la identificación) con las cámaras adecuadas

1.3.2 - Impactos Negativos

- Como todo sistema es vulnerable:
    - Según la toma e interpretación de datos puede generar resultados de detección y reconocimientos erróneos. A menos que se ajusten y entrenen correctamente, tienen un alto índice de no coincidencia, falsos reconocimientos.
    - A los sistemas de reconocimiento facial 2D les cuesta distinguir una cara en vivo de una foto. Se podría de esta manera realizar una suplantación de identidad usando una imagen de alta calidad o una máscara. Es un problema que se intenta superar con reconocimiento facial 3D, cámaras TOF que detectan la profundidad o con movimientos tras la toma inicial, como pedir girar la cabeza o parpadear.
    - Es manipulable. Se puede manipular la imagen inicial matriculada y falsear así una identificación / autenticación. Actualmente se agrava el problema con las imágenes sintéticas, se pueden generar imágenes falsas difícilmente detectables, por ejemplo, Deep Fakes generadas con GAN (Generative Adversarial Networks)

- Si no tiene un soporte legal adecuado, puede haber vulneración de la privacidad. Por ejemplo: Clearview AI extrae huellas faciales de miles de millones de personas, sin su consentimiento, y utiliza estas huellas faciales para ayudar a la policía a identificar a los sospechosos. Esto daña gravemente la privacidad, la libertad de expresión, la seguridad de la información y la justicia racial [16]. En 2021 Meta anunció el borrado masivo de datos faciales recopilados por Facebook [90].

El uso de este tipo de tecnología según sus fines puede ser: desde muy beneficioso, a restrictivo e incluso discriminatorio.

1.3.3 - Noticias relacionadas con el uso del reconocimiento facial:

- “Using LinkedIn profile pictures for facial recognition” [18]
- “10820 cámara con reconocimiento facial para rastrear infectados…” [19]
- “Mercadona paga una multa de 2,5 millones a Protección de Datos…” [20]
- “Policía y Guardia Civil preparan ABIS, el sistema de reconocimiento facial…” [21]

- "Nuevo problema de seguridad en las fronteras: engañan al sistema de reconocimiento facial. La compañía de ciberseguridad McAfee consigue engañar a un software de reconocimiento facial mezclando rostros de dos personas al mismo tiempo." [22]
- "Interior usará el reconocimiento facial en espectáculos para detectar personas con causas pendientes" [23]
- "Police arrested wrong man based on facial recognition fail… " [24]
- La UE plantea prohibir hasta 5 años el reconocimiento facial en lugares públicos para analizar sus riesgos [25]
- Meta borrará los datos de reconocimiento facial de más de mil millones de personas usados en Facebook: más golpes de efectos de Zuckerberg [90]

1.4 - Enfoque y método seguido

Existen múltiples algoritmos para realizar una detección y posterior identificación de un rostro. Con la revisión del Estado del Arte, de los dataset y algoritmos aplicados a estos, elegiremos el/los que más se acercen al objetivo marcado en este TFM. Durante el desarrollo del trabajo, se mostrará la descripción y funcionamiento de las redes convolucionales, que serán el soporte técnico de la POC. Para esto es necesario seguir un método de tratamiento de datos. El proceso de Análisis de Datos conlleva la recolección, transformación, limpieza y modelado de datos para descubrir la información útil. En la minería de datos los métodos más usados son:

1.4.1 - KDD

- **KDD (Knowledge Discover Database).** Consta de 5 fases:
    - **Selección**: se seleccionan las variables, o si se va a trabajar con muestras, los datos que deben ir a las mismas, su distribución, entre otras.
    - **Preprocesamiento**: los datos se limpian los datos y se adaptan a las herramientas disponibles para el análisis de datos
    - **Transformación**: Se aplican los métodos de reducción de la dimensión u otros métodos de transformación para poder aplicar las técnicas de Data Mining elegidas
    - **Minería de datos**: se resuelven las hipótesis planteadas con las técnicas de Minería de Datos escogidas, se emprende la búsqueda de nuevos hallazgos y se intenta detectar patrones en los datos
    - **Interpretación / Evaluación**: se extraen las conclusiones del estudio, y se evalúan los modelos de Minería de datos usados.

1.4.2 - SEMMA

- **SEMMA (Sample, Explore, Modify, Model and Assess)**. Consta de 5 fases:
    - **Muestrear**: se extraen muestras de los datos que sean representativas para el análisis, de tamaño adecuado para poder manipularlas en un tiempo y con unos recursos adecuados.
    - **Explorar**: se realiza el análisis preliminar de los datos
    - **Modificar**: se modifican los datos, se aplican transformaciones y realizan selecciones para crear las variables necesarias

- **Modelar**: se aplica los modelos de Minería de Datos, se obtienen funciones o combinaciones de las variables para predecir la variable objetivo.
- **Evaluar**: se evalúa la utilidad y fiabilidad de los insights obtenidos con el modelo y se estima el rendimiento

1.4.3 - CRISP-DM

- **CRISP-DM (Cross Industry Standard Process for Data Mining).** Consta de 6 fases:
  - **Entendimiento del Negocio**: Entender los objetivos y requisitos desde un punto de vista de Negocio, como paso previo a la definición del problema de Minería de Datos.
  - **Entendimiento de los datos**: Se recolectan los datos, se exploran, se detectan problemas con la calidad de los mismos, y se obtienen los primeros insights, subconjuntos de datos, primeras hipótesis.
  - **Preparación de los datos**: Se construye un dataset a partir de los datos en crudo.
  - **Modelado**: se seleccionan y aplican técnicas de modelado. Se calibran sus parámetros en búsqueda de los mejores resultados.
  - **Evaluación**: se evalúan los modelos y se revisan los pasos seguidos para la construcción de los Modelos, en relación a los objetivos de Negocio.
  - **Despliegue**: El/los modelo/s se “industrializa/n”. Se puede llegar a desplegar en un sistema o entorno de producción, o al menos se genera un entregable que el cliente pueda usar.

En este último modelo, CRISP-DM hay una fase inicial que es el entendimiento del negocio y una final que es el despliegue. Al tratarse de una Prueba de Concepto, previo a las fases propia del tratamiento de los datos, hay que definir la POC y determinar si sería válida para una posterior implementación en un producto real. Por tanto, el modelo CRISP-DM parece el que mejor se adapta a estos requerimientos, es por tanto el método que vamos a seguir en el TFM.

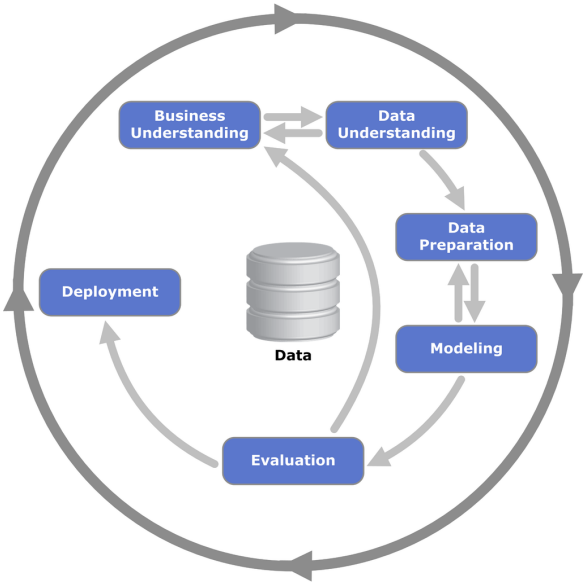


Figura 7. Diagrama CRISP-DM. Fuente: [26]

La Prueba de Concepto se realizará en Jupyter-Notebook, en local con GPU, implementando sobre un dataset elegido y las librerías necesarias, como ejemplo: numpy, opencv, tensorflow o keras entre otras.

1.5 - Planificación del Trabajo

La planificación del TFM se desarrolla en torno a las fecha de seguimiento y tareas a desarrollar.

TFM - Biometría facial con Deep Learning del 28/09/22 al 27/01/23 - Total 88 días

1.5.1 - PEC1 del 28/09/22 al 11/10/22 - 10 dias

- Objetivos y Alcance

1.5.2 - PEC2 del 12/10/22 al 08/11/22 - 20 días

- Recopilacion de Bibliografia
- Redaccion del Estado del Arte

1.5.3 - PEC3 del 09/11/22 al 06/12/22 - 20 días

- Prueba de Concepto POC
    - Datasets
    - ConvNet
- Diseño POC
- Recursos.
- Implementación
- Pruebas y resultados

1.5.4 - PEC4 del 07/11/22 al 10/01/23 - 25 días

- Redacción de la memoria

1.5.5 - Elaboración y presentación de Video 11/01/23 al 17/01/23 - 5 dias

- Presentación de video

1.5.6 - Defensa del TFM del 23/01/23 a 27/01/23 - 5 dias

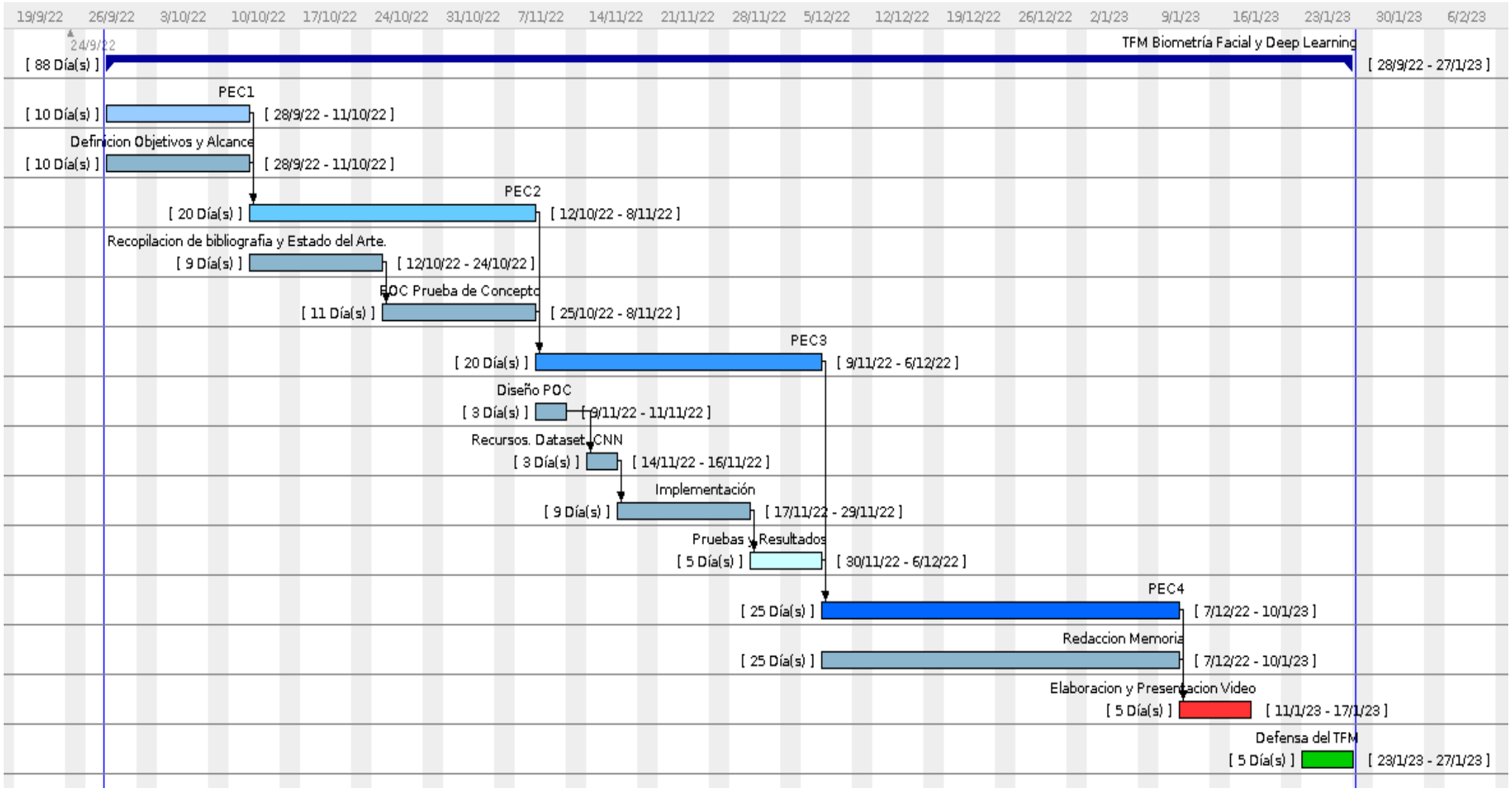


Figura 8. Gantt del TFM. Fuente: Propia

2 - Redes Neuronales

2.1 - Introducción. Redes neuronales

2.1.1 - Organización por capas

El aprendizaje profundo es un subconjunto especial del aprendizaje automático, las neuronas se estructuran capas y niveles para crear una red neuronal artificial capaz de aprender y tomar decisiones inteligentes por sí misma [1-4]

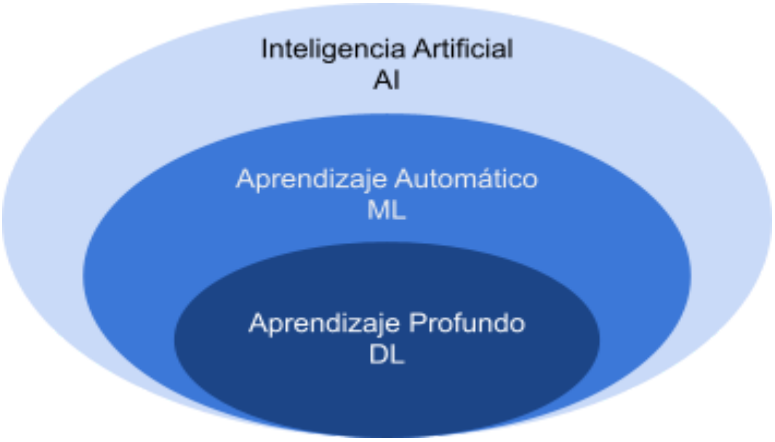


Figura 9. Relación entre AI - ML - DL. Fuente: Propia

Para las redes neuronales regulares, los tipos de capas más comunes son las capas completamente conectadas donde las neuronas entre dos capas adyacentes están completamente conectadas cada neurona con todas las neuronas de la siguiente capa. Las neuronas dentro de la misma capa no comparten conexiones.

Es convención que en una red neuronal de N capas, no se tiene en cuenta la primera capa de entrada. Las redes regulares completamente conectadas se denominan indistintamente "Redes neuronales artificiales" (ANN) o "Perceptrones multicapa" (MLP). Una red neuronal de una sola capa es una red sin capas ocultas donde la/s entrada/s se asigna/n directamente a la/s salida/s.

Las neuronas de la capa de salida generalmente no tienen una función de activación, a diferencia de todas las capas en una red neuronal. Se puede interpretar como una salida que tienen una función de activación de identidad lineal.
La última capa de salida:

- Clasificación: valores booleanos
- Regresión: valores de números reales arbitrarios

El número de neuronas que componen la red neuronal se denomina habitualmente número de parámetros que se componen de los pesos w y los sesgos b.

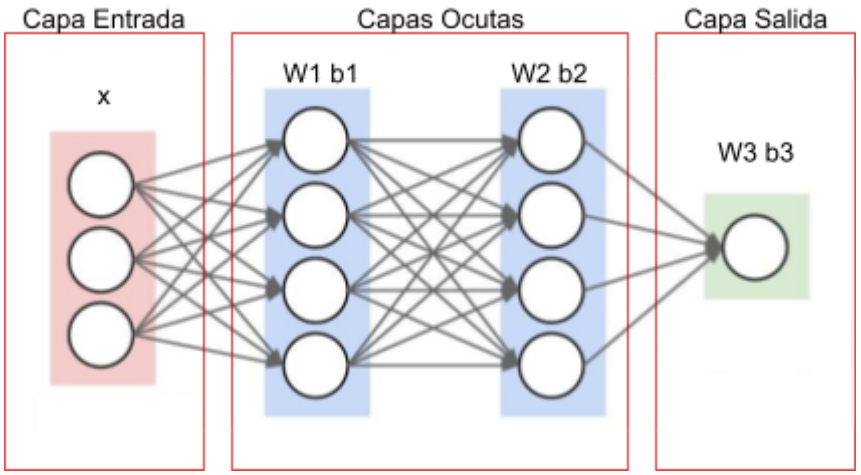


Figura 10. Esquema Neurona. Fuente: Propia

Antes de explicar las redes neuronales convolucionales (ConvNet ó CNN) hay que explicar sucintamente cómo funciona una red neuronal y sus parámetros habituales.

En la siguiente imagen vemos el modelo matemático de una neurona artificial.

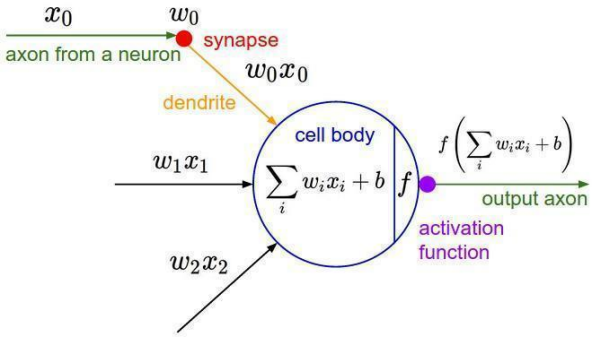


Figura 11. Neurona artificial. Fuente: [27]

Cada neurona recibe señales de entrada (dendritas) y produce una salida (axón). El axón de salida se conecta a través de la sinapsis con las dendritas de otras neuronas (una o varias), y de este modo reiterativo forman las redes neuronales. En el cuerpo de la neurona artificial se produce el sumatorio de las entradas x_i por un valor denominado peso w_i.
Así, en este modelo descrito en la imagen de una neurona artificial:

- **las señales** que viajan a lo largo de los axones comenzando por la izquierda con $\mathbf{X_0}$
- se van **multiplicando** por el **peso** $\mathbf{W_0}$ (variable)
- y junto al resto de **entradas** $\mathbf{X_i}$ se **suman** en el cuerpo de la neurona produciendo una salida

$$f\left(\sum_{i}^{n} w_i x_i + b\right)$$

En analogía con la neurona biológica, el valor sináptico W es variable y controlan el peso de la entrada X. El término b es el sesgo o bias.

Resumiendo, las dendritas llevan la señal al cuerpo celular donde se suman todas:

- Si la suma final está por encima de cierto umbral, la neurona se dispara enviando un pico a lo largo de su axón. En el modelo artificial, diríamos que se activa o dispara y comunica la información
- Por el contrario por debajo del umbral no se transmite información

Llegados a este punto ¿quien decide cual es el umbral que dispara a la neurona? Se hace necesaria una función que decida dicha activación: la función de activación f.

En los inicios de la redes neuronales, una de las funciones de activación más usada fué la función sigmoidea σ, que toma una entrada de valor real (la intensidad de la señal después de la suma) y la adapta para que oscile en los extremos del rango entre 0 y 1, osea valores que reales que se aproximen mucho a 0 o a 1, a esto habitualmente se le conoce como aplastamiento de la señal (a 0 ó a 1).

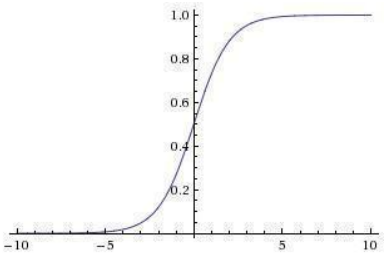


Figura 12. Función sigmoidea. Fuente: [27]

La función sigmoidea

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

2.2 - Funciones de activación habituales

Cada función de activación toma un nº real y realiza una operación matemática fija sobre él.

2.2.1 - Función Lineal

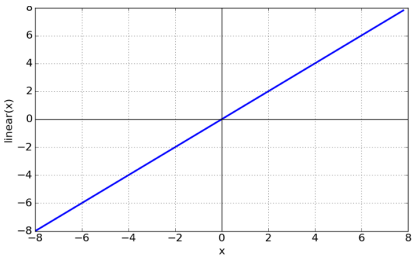
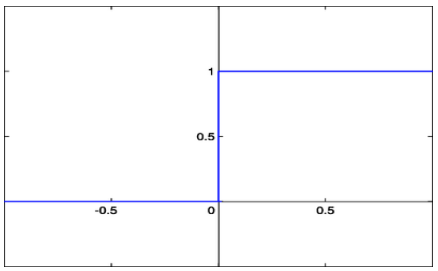


Figura 13. Función lineal. Fuente: [27]

No se usa en la actualidad, o como mucho en la capa final en redes neuronales con regresión. El problema fundamental es que al ser una función lineal la entrada se incrementa sin límite, no se puede normalizar y se vuelve inestable. Es conocida también como regresión lineal.

2.2.2 - Función Umbral



Es una función escalón, donde:

- Todos los valores de entrada negativos producen un 0. $f(x) = 0 \; si \; x \leq 0$
- Todos los valores de entrada positivos producen un 1. $f(x) = 1 \; si \; x > 0$

Figura 14. Función umbral. Fuente: [27]

Esta función se usa cuando se quiere clasificar o cuando se tiene salidas categóricas

2.2.3 - Función Sigmoidea o logística.

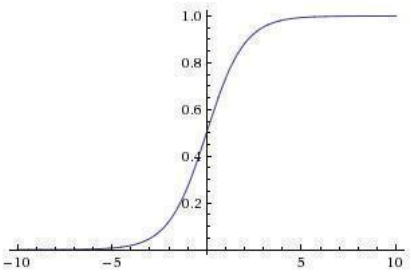


Figura 15. Función sigmoidea. Fuente: [27]

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

La salida es un valor continuo en el rango [0,1]:

- Los valores de entrada negativos producen un valor muy cercano al 0. $si\ x \leq 0$
- Los valores de entrada positivos producen valor muy cercano al 1. $si\ x > 0$

La función logística toma un número de valor real y lo "aplasta" en un rango entre 0 y 1
Su uso suele ser en problemas de clasificación, aunque actualmente en la práctica no se usa.

2.2.4 - Función Tangencial Hiperbólica Tanh

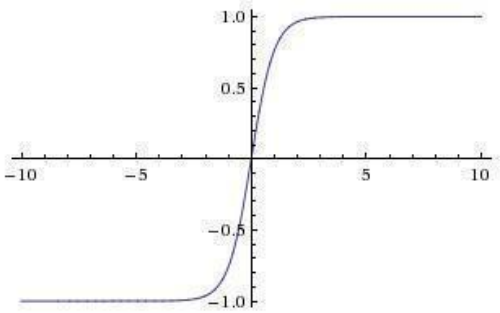


Figura 16. Función tanh. Fuente: [27]

$$tanh(x) = 2\sigma(2x) - 1$$

La salida es un valor continuo en el rango [-1,1]:

- los valores de entrada negativos producen un valor muy cercano al -1 $si\ x \leq 0$
- los valores de entrada positivos producen valor muy cercano al 1 $si\ x > 0$

Esta función es una variación de la función sigmoidea, aplasta los números reales para que oscilen en el rango [-1,1] Al igual que las redes con activación sigmoidea, sus activaciones se saturan, pero como diferencia la salida está centrada en cero. Proporciona un mejor rendimiento en la retropropagación pero no evita la desaparición del gradiente.

2.2.5 - Funcion ReLu

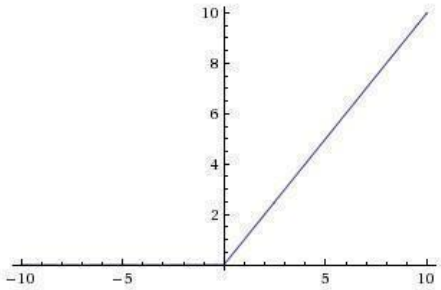


Figura 17. Función ReLu. Fuente: [27]

$$f(x) = max(0,\ x)$$

La salida es un valor continuo en el rango [0, x]:

- Los valores de entrada negativos producen un valor 0. $\forall\, x \in \Re$
- Los valores de entrada positivos producen valor función lineal de x. $\forall\, x \in \Re$

Es de las funciones más usadas en la redes neuronales profundas debido a que acelera la convergencia del descenso de gradiente en comparación con las sigmoide o tanh

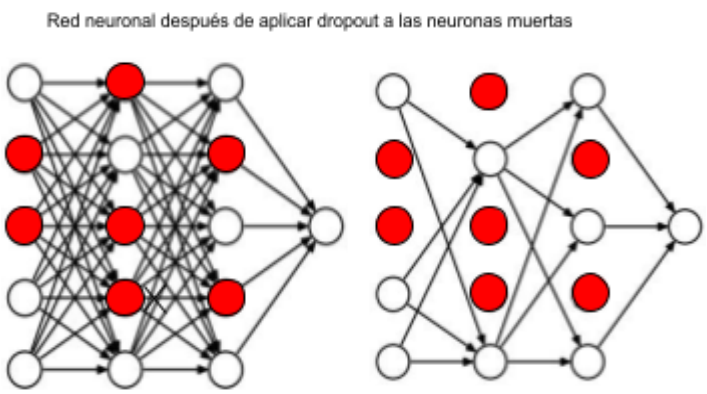
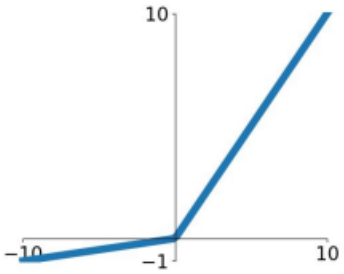


Figura 18. Dropout. Fuente: [27]

Es una función lineal que elimina los valores de entradas negativos. Las neuronas activadas con ReLU pueden ser frágiles durante el entrenamiento y pueden desactivarse. Por ejemplo, un gradiente de tamaño elevado podría hacer que los pesos se actualicen de tal manera que la neurona nunca se vuelva a activar. A pesar de esto, son más los beneficios que las deficiencias, y estos sobreajustes se suelen tratar con una capa DropOut.

2.2.6 - ReLU con fugas. Leaky ReLu



$$f(x) = max(0.1x,\ x)$$

La salida es un valor continuo en el rango [0.1x, x]:

- Los valores de entrada negativos producen un valor cercano al 0. $si\ x \leq 0$
- Los valores de entrada positivos producen valor función lineal de x. $si\ x > 0$

Figura 19. Función Leaky ReLu. Fuente: [27]

Los ReLU con fugas son un intento de solucionar el problema de la desactivación de neuronas. En lugar de que la función sea cero cuando x < 0, una ReLU con fugas tendrá una pequeña pendiente positiva de 0,01x.

2.2.7 - PReLu. Unidad Lineal Rectificada Paramétrica

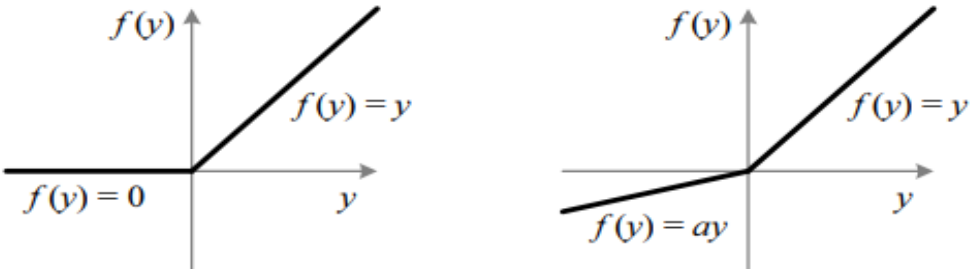


Figura 20. Funcion PReLu. Fuente: [27]

La pendiente en la región negativa también se puede convertir en un parámetro de cada neurona. PReLU mejora el ajuste del modelo con un costo computacional adicional casi nulo y un riesgo mínimo de sobreajuste

2.2.8 - Función Softmax

Es una generalización de la función sigmoide, función exponencial normalizada. La función softmax es utilizada como capa final de los clasificadores basados en redes neuronales. Se suele usar como una función de activación de salida para la clasificación multiclase. Los valores de salida tienen una probabilidad en el rango de [0,1] de forma excluyente de forma que todos los valores del vector de salida suman 1.

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \quad \text{para } j = 1,\ldots,K$$

Figura 21. Función softmax. Fuente: [27]

La salida de la función softmax puede ser utilizada para representar una distribución categórica– la distribución de probabilidad sobre K diferentes posibles salidas. Esta función toma unos valores de entrada y son transformados en una distribución de probabilidad, la suma de todas las probabilidades tiene que dar uno y aquella clase con mayor probabilidad será la escogida por la red neuronal. Es muy usada en las redes convolucionales.

2.3 - Entrenamiento. Propagación / Retropropagación

Hasta ahora hemos visto una neurona que toma datos y produce una salida que se denomina propagación directa. Habitualmente se asignan unos valores a w_i y a los sesgos b que posteriormente serán modificados.

$$f(\sum_{i}^{n} w_i x_i + b)$$

Para que una neurona aprenda hay que ajustar los pesos de W y el sesgo b.

El entrenamiento de una red neuronal implica dos etapas:

- propagación hacia adelante (Forward propagation)
- propagación hacia atrás (Backpropagation)

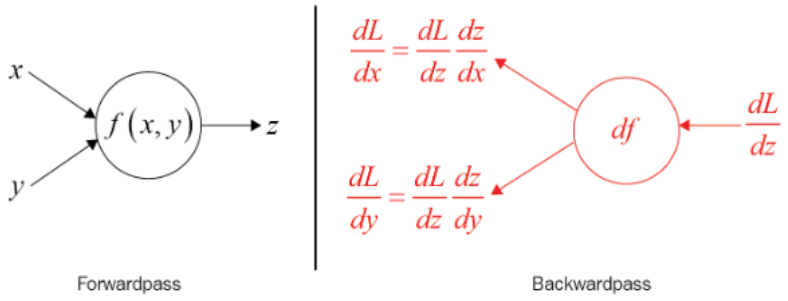


Figura 22. Propagación. Fuente: [27]

Pasos que se producen durante el entrenamiento de una red neuronal

- Una **iteración de entrenamiento** donde la red neuronal pasa por todos los ejemplos de entrenamiento se denomina **Epoch**. Este es uno de los hiperparámetros que se modificarán para entrenar una red neuronal.
- En cada paso, una red neuronal realiza una **propagación hacia adelante**, donde los datos viajan desde la entrada hasta la salida. Los datos de entrada se operan en el cuerpo de la neurona y se le aplica la función de activación para producir la salida prevista. La salida se compara con la previsión y se calcula el error o la pérdida.
- Con el **cálculo del error** realizado por la función de pérdida se ajustan los pesos y sesgos para acercarse a las salidas previstas (correctas, previamente etiquetadas). En esto consiste el aprendizaje neuronal
- Para **minimizar el error** en la red neuronal, el bucle de entrenamiento necesita un optimizador, que minimizará la pérdida
- Una vez que se calcula el error, la red neuronal comprueba los nodos de la red contribuyeron al error y su ponderación. De este modo se modifican los parámetros pertinentes en la siguiente época Epoch. Esta búsqueda del error hacia atrás (propagar el error hacia atrás) se denomina **propagación hacia atrás Backpropagation**. El Backpropagation usa la regla de la cadena del cálculo para propagar el error, el gradiente de error en orden inverso hasta que alcanza la capa de entrada. A medida que se propaga el error a través de la red, utiliza el descenso de gradiente para realizar ajustes precisos en los pesos y sesgos de la red utilizando el gradiente de error calculado anteriormente.

Función de pérdida: es una función que evalúa la desviación entre las predicciones realizadas por la red neuronal y los valores reales de las observaciones utilizadas durante el aprendizaje. Cuanto menor es el resultado de esta función, más eficiente es la red neuronal.

Optimizador: su función, es decir, reducir al mínimo la desviación entre el valor predicho y el valor real para una observación dada, se hace ajustando los distintos pesos de la red neuronal. Son funciones estadísticas que actúan sobre los parámetros. La mas habituales son:

- Stochastic Gradient Descent (SGD)
- Adagrad
- Adadelta
- RMSprop
- Adam
- Nadam

Backpropagation: De los datos obtenidos de la función de pérdida y del optimizador, se retrocede en una función denominada backpropagation para ajustar los parámetros. El método de cálculo de backpropagation evalúa las entradas de prueba adecuadas para garantizar que se obtienen el resultado en las salidas óptimas. El paso hacia atrás para una operación de convolución, tanto para los datos como para los pesos, también es una convolución pero con filtros invertidos espacialmente.

2.4 - Redes Neuronales Convolucionales ConvNet

Las redes neuronales convolucionales (ConvNet) se componen múltiples capas para calcular la salida del conjunto de datos de la entrada, en este caso el dataset de imágenes. Desde 2012, las redes neuronales convolucionales son la mejor opción para el procesamiento y extracción de la información de imágenes

Podríamos dividir el tratamiento para obtener información de las imágenes en:

- Procesamiento de señales para poder realizar mejoras y/o cambios en la imagen original con el fin de hacerla más adecuada para extraer información.
- Extracción de los datos que nos interesan para el objetivo buscado. Se suelen utilizar técnicas de minería de datos

Tanto para procesar la imagen como para extraer información, la técnica más utilizada es la convolución. **La información que se extrae se almacena en vectores de características** y se usan como elementos que **representan al objeto o partes del objeto**.
Durante mucho tiempo se han definido filtros manualmente con el objetivo de mejorar la imagen o extraer características de la misma, como los bordes, por ejemplo. Existen filtros muy conocidos como los gaussianos para eliminar o añadir ruido, o como Sobel para la extracción de las derivadas direccionales de la imagen

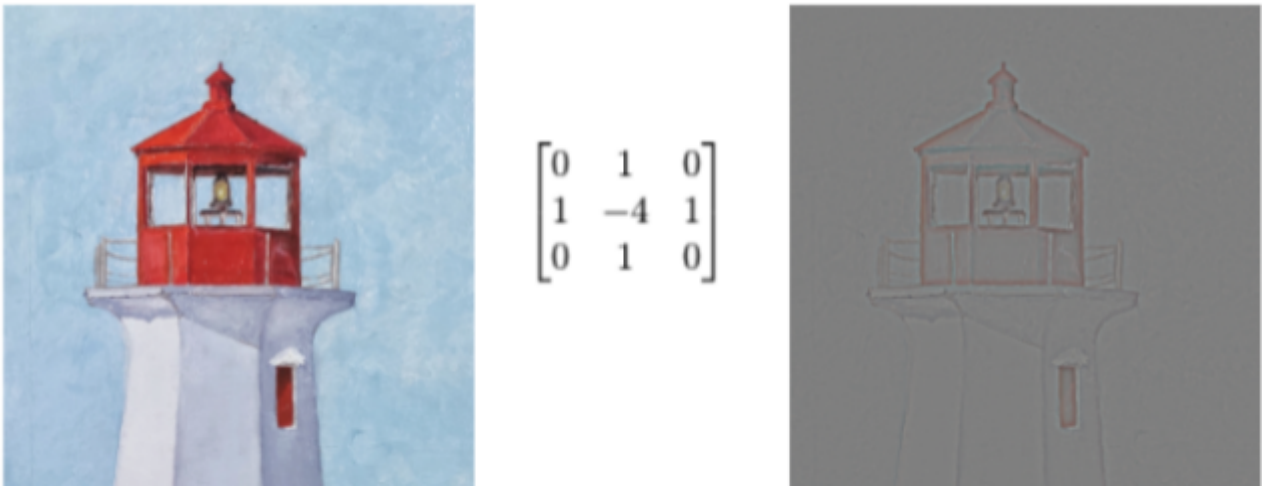


Figura 23. Filtro Sobel. Fuente: propia

Extraer funciones y/o características de una imagen es fundamental para analizar el contenido de la imagen y dar significado a los objetos que aparecen dentro de una imagen. La imagen es una **matriz**, un conjunto de **valores ordenados** ancho **w** × alto **h**. Si la imagen es en color (RGB) esa matriz, aparte de ancho y alto (w x h) tiene profundidad d, por tanto sería w x h x d. En este caso d=3, las tres capas RGB.

2.4.1 - Convoluciones de imágenes con más de una dimensión.

RBG es una forma de definir una imagen a color, pero existen otras como CMYK. Para este caso la profundidad **d** pasaría de 3 (RGB) a 4 (CMYK). Por tanto **frente a la convolución** veremos la imagen como un volumen, un **paralepípedo rectangular (W, H, D)**:

- W es la anchura
- H es la altura
- D es la profundidad

Las Redes Convolucionales se diseñaron para procesar imágenes. A diferencia de una red neuronal normal, las capas ConvNet tienen neuronas dispuestas en 3 dimensiones: ancho **W**, alto **H**, profundidad **D**. En las redes neuronales densas se tiene una matriz de pesos **(m,n)** para conectar una capa anterior de m neuronas con una de n. Aquí se conectarán las neuronas con capas convolucionales usando un bloque de **(w,h,d)** pesos. La reducción de complejidad viene dado porque la dimensión de **(w,h)** es muy inferior a **(m,n)**

Una operación de convolución básica que se aplica a una **imagen bidimensional I** como entrada, usando un kernel o filtro **K** bidimensional y que nos da como resultado una nueva imagen **S** sería por ejemplo:

$$(i,j) = (K \cdot I)(i,j) = \sum_m \sum_n = I(m,n)K(i-m, j-n)$$

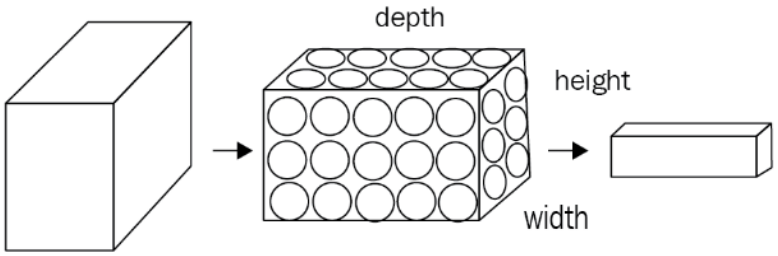


Figura 24. Convolución. Fuente: [27]

Es obvio que en una imagen los valores de D son distintos según:

- en escala de grises D = 1
- en color RGB D = 3
- y en color CMYK D = 4

Una imagen que tenga una profundidad D = n necesita n filtros. Luego de una imagen RGB se obtienen 3 mapas de características. Son mapas porque tienen una dimensión W x H.

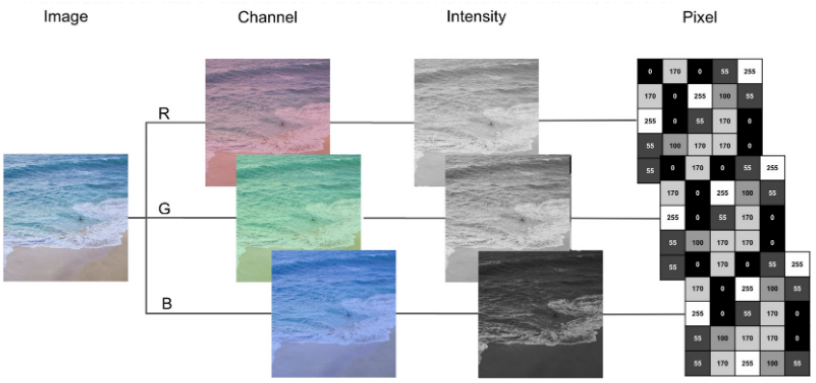


Figura 25. Capas RGB. Fuente: [94]

A veces según necesidad se vectorizan. Si estamos interesados en buscar características básicas y elementales, como las líneas horizontales o verticales, o un color en particular, entonces una sola característica puede ser suficiente. Sin embargo, si estamos interesados en la investigación de un patrón más complejo como es el rostro de una persona una sola característica no es suficiente para caracterizar la complejidad y la variabilidad de una imagen, como: pose, variaciones de luz, oclusiones, colores…

Definir manualmente los filtros capaces de extraer características es un proceso largo y complejo que no siempre conduce a los objetivos buscados. Para hacer esto, es posible utilizar las redes neuronales artificiales, colocar neuronas en cuadrículas 2k × 2k, que son utilizados como filtros (de convolución) para la extracción de las características y modificar estos valores durante la retropropagación para ir afinando el aprendizaje. Durante el proceso de entrenamiento se envía a la red las imágenes a clasificar y el valor de la etiqueta real que deben buscar (aunque no siempre es supervisado). Durante este proceso la pérdida también debe ser ajustada respecto al valor de la predicción, por ello es necesario añadir capas de clasificación totalmente conectadas a la red, para realizar la combinación de las características extraídas y obtener el valor de la clase buscada. Inicialmente no se conoce cuántas características son necesarias extraer para caracterizar en su totalidad a las entradas. Por esto es necesario añadir un filtro de profundidad (hiperparámetro) que son la cantidad de filtros de convolución que intentan generar aprendizaje.

Durante el paso hacia delante, propagación, el volumen de la entrada tiene un filtro extraído de la capa de convolución (con igual profundidad a la profundidad del volumen de entrada). El resultado de la convolución con el filtro genera un mapa de características. El conjunto de mapas de características, generado por n filtros genera un volumen con profundidad n que a su vez puede ser utilizado para extraer otras características. El número de parámetros aumenta a mayor número de capas de convolución. Esto necesita una potencia computacional enorme. Un método para reducir el número de parámetros sin reducir la efectividad del aprendizaje es la agrupación (pooling).

Las redes neuronales convolucionales son, en términos generales, modelos de redes de neuronas artificiales multicapas. Cada uno de los bloques representa una capa diferente de la red de neuronas convolucionales, los cuales tienen altura, anchura y profundidad. De izquierda a derecha podemos observar, las capas de entrada, las capas ocultas (convolucional, agrupación y capas de abandono) y las capas completamente conectadas. Después de la última capa, el modelo genera una clasificación.

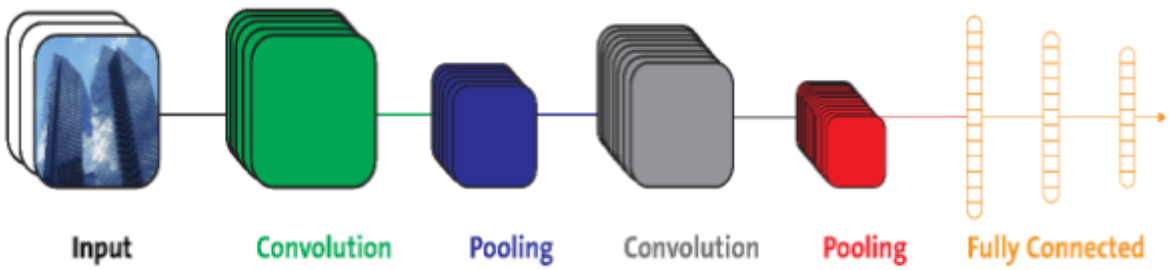


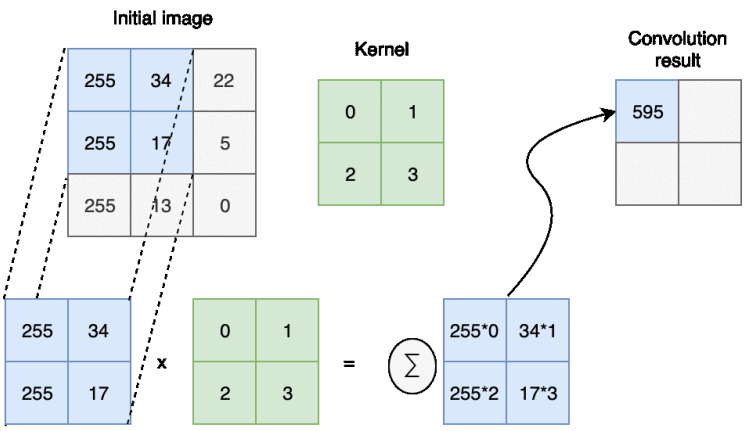
Figura 26: Red Convolucional Genérica. Fuente: [27]

2.5 - Capas Arquitectura Convolucional

La estructura de capas básicas de una red neuronal convolucional contiene **[INPUT - CONV - RELU - POOL - FC]**:

- **Capa de entrada - INPUT** [WxHxD]**:** Contiene los valores de píxeles raw de la imagen
- **Capa convolucional - CONV:** Su principal función es la extracción de características de los datos que recibe en la entrada mediante la convolución. De esta capa inicial se extrae un mapa de características que se envía a la siguiente capa como datos. Calcula la salida de las neuronas que están conectadas a las regiones locales en la entrada, cada una calculando un producto de punto entre sus pesos y una pequeña región a la que están conectadas en el volumen de entrada. Para una profundidad D con D filtros el volumen sería [WxHxD]. Los pesos que conectan los píxeles localmente son compartidos y son aprendibles Los mapas de características a su vez pueden servir como mapas de activación (según la arquitectura donde se usen). Una red neuronal convolucional suele contar con varias capas de convolución en su estructura.
- **Capa ReLu - RELU:** Es usada como función de activación $max(0, x)$ anulando los valores negativos y actuando sólo sobre los positivos
- **Capa de agrupación - POOL:** Esta capa reduce mediante un remuestreo la dimensionalidad del mapa de características manteniendo la información crucial. La función más habitual es max-pooling, aunque existen otras que dependen del tipo de arquitectura ConvNet. Esta capa no utiliza pesos aprendibles.
- **Capa Full-Conect o Densa - FC:** Es quien recoge la información de los mapas de características y capas de agrupación, para procesar el aprendizaje profundo de la red neuronal. Calculará los puntajes de clase, lo que resultará en un volumen de tamaño [1x1xK], donde cada uno de los K números corresponde a un puntaje de clase, entre las K categorías del conjunto. Como en las redes neuronales ordinarias, cada neurona en esta capa estará conectada a todas del volumen anterior.

2.5.1 - Capa convolucional. Convolutional layer



Es la capa principal de las redes neuronales convolucionales, siendo indispensable el uso de una o de más capas de este tipo en dichas redes. Los parámetros en una capa convolucional en la práctica se refiere a un conjunto de filtros entrenables. Cada filtro ocupa un espacio pequeño a lo largo de las dimensiones de anchura y altura, pero se extiende por toda la profundidad del volumen de entrada al que se aplica.

Figura 27. Ejemplo de convolución. Fuente: [29]

Durante la propagación o la retropropagación, cada filtro a lo largo de la anchura y de la altura del volumen de entrada, produce un mapa de activación bidimensional (mapa de características) para ese filtro. Se denomina mapa porque es una matriz bidimensional que representa una

pequeña área de la imagen original. Intuitivamente, la red aprenderá filtros que se activan cuando ven algún tipo de característica visual, como un borde de alguna orientación o una mancha de algún color en la primera capa, o eventualmente patrones completos en forma de panal o rueda en capas superiores de la red. Cada uno de ellos producirá un mapa de activación 2-dimensional separado. Se apilan estos mapas a lo largo de la dimensión de profundidad resultando el volumen de salida. A medida que el filtro se mueve a lo largo del área de la entrada, se efectúa un producto escalar entre los valores del filtro y los de la región de entrada a la que se aplica.

La convolución se podría interpretar como una salida de una neurona que mira solo una pequeña región en la entrada y comparte parámetros con todas las neuronas a la izquierda y a la derecha espacialmente, al aplicar el mismo filtro kernel. La salida de una capa convolucional supone poner en cola todos estos mapas de características para todos los filtros, a lo largo de la profundidad.

Retropropagación. El paso hacia atrás para una operación de convolución, tanto para los datos como para los pesos, también es una convolución pero con filtros invertidos espacialmente. Es el momento donde se ajustan los pesos.

2.5.1.1 - Padding o relleno

El padding es un marco con valor cero donde se envuelve la imagen a tratar para no perder información de las esquinas.

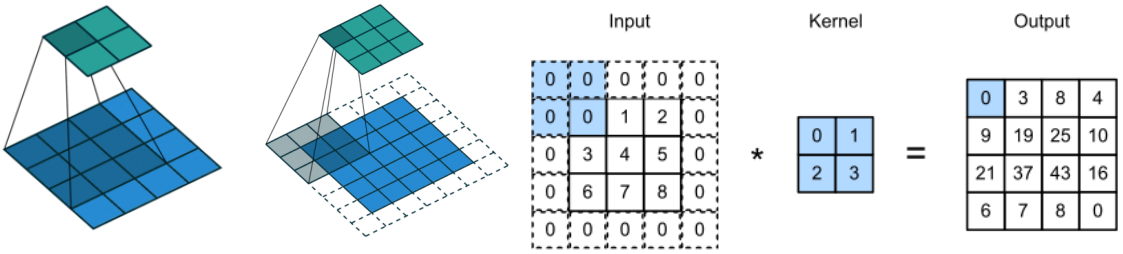
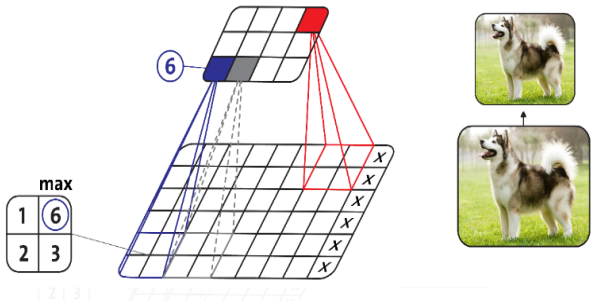


Figura 28. Ejemplo de padding. Fuente: [31]

2.5.2 - Capa de agrupación. Pooling layer



Las capas de agrupación realizan una simplificación de la información en la salida de la capa convolucional. Se colocan tras las capas convolucionales. La capa de agrupación toma los mapas de características producidos en la capa de convolución y los agrupa mediante una función, realizando una reducción de la dimensionalidad de la capa y disminuyendo la complejidad para evitar el sobreajuste.

Figura 29. Reducción Max Pooling. Fuente: [30]

La operación de agrupación consiste en extraer regiones 2k ×2k y aplicar una función f al mapa 2k×2k seleccionando un solo valor, moviendo S píxeles y repitiendo la operación. Esto se hace para todos los niveles del volumen de entrada. **La agrupación reduce el ancho W y el alto H, pero no toca la profundidad D**, se conserva la profundidad del volumen de entrada.

En la imagen de la Figura 30, de la primera agrupación de 20x20 se pasa a 10x10. Se puede observar que la profundidad sigue siendo la misma, 4. El pooling se aplica a cada mapa de características por separado.

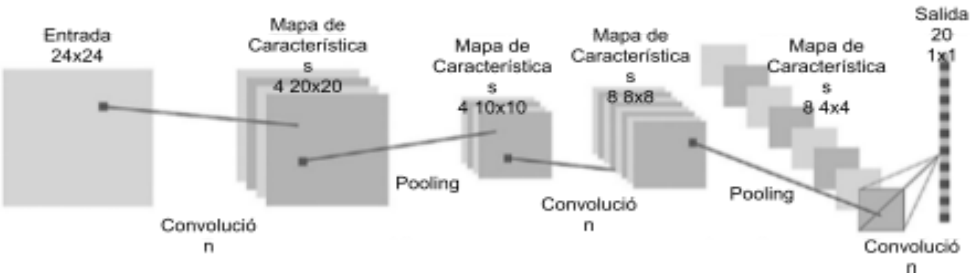


Figura 30. Grupos de capas Convolución - Pooling. Fuente: [27]

2.5.2.1 - Max Pooling

Se extrae en valor máximo del área seleccionada. Se extrae de este modo la característica más relevante. **Es la función de agrupación más utilizada.**

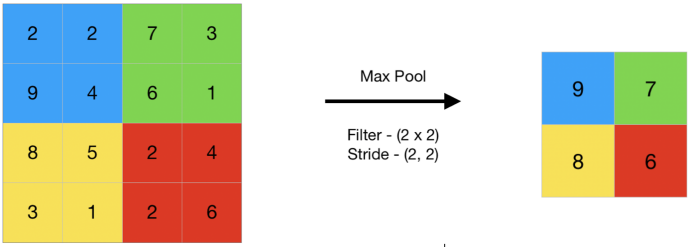


Figura 31. Ejemplo de Max Pooling. Fuente: [30]

2.5.2.2 - Average Pooling

Se escoge en este caso el valor medio del área seleccionada

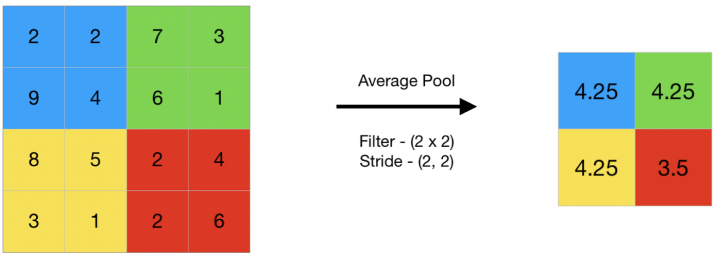


Figura 32. Ejemplo de Average Pooling. Fuente: [30]

La agrupación logra que la red pregunte si una determinada característica se encuentra en algún lugar de una región de la imagen. Una vez se ha encontrado una característica, su ubicación exacta no es tan importante como su ubicación aproximada en relación con otras características. Al reducir conservando la característica relevante hay menos características agrupadas, se reduce la cantidad de parámetros necesarios en las capas posteriores.

Aunque Max-Pooling es la más usada, no es la única técnica utilizada en la agrupación. Otra técnica es conocida como agrupación L2 (L2-pooling). En este caso, en lugar de tomar la activación máxima de una región específica de neuronas 2 × 2, se toma la raíz cuadrada de la suma de los cuadrados de las activaciones 2 × 2. Es similar a max-pooling, pero la agrupación L2 condensa aún más la información de la capa convolucional.

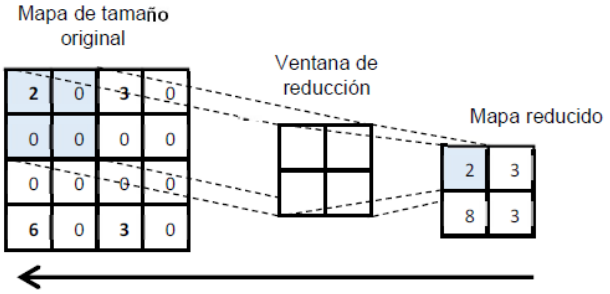


Figura 33. Max Pooling durante la retropropagación. Fuente:[126]

Springenberg et al. [126] proponen descartar la capa de agrupación en favor de una arquitectura que solo contenga capas convolucionales. Para reducir el tamaño de la representación, la agrupación máxima se puede reemplazar simplemente por una capa convolucional con mayor zancada sin pérdida de precisión. Los autocodificadores variacionales (VAE) o las redes generativas adversarias (GAN) entrenan mejor y dan mejores resultados descartando las capas pooling. Con el aumento de las GAN puede ser una línea que las arquitecturas futuras inserten muy pocas o ninguna capa de agrupación.

2.5.4 - Capa ReLu

ReLu (Rectified Linear Units) es un tipo de capa muy común en ConvNet. Se usa múltiples veces en una misma red de neuronas artificiales, frecuentemente después de cada capa convolucional. La función de esta capa es la de aumentar la propiedad de no linealidad de la función de activación, sin que se modifiquen los cuerpos receptores de la capa convolucional. ReLu suele definirse como f(x) = max(0, x) aunque puede haber variaciones lineales.En este tipo de capas a veces se usan la función Tanh o la sigmoide

2.5.5 - Capa completamente conectada. Fully connected layer

Todas las neuronas de esta capa se encuentran conectadas a todas las neuronas de la capa anterior. Por tanto, se encuentran conectadas a todos los mapas de características de la capa anterior. En este tipo de capa no utiliza la propiedad de conectividad local, ya que una capa completamente conectada se encuentra conectada a todo volumen de entrada, por lo que tiene un gran número de conexiones.

Por el contrario, las capas convolucionales se encuentran conectadas a una sola región local en la entrada. Además muchas de las neuronas de la capa convolucional comparten parámetros.

El único parámetro que se puede configurar en este tipo de capa es el número de neuronas que la componen. En estas capas se conectan sus K neuronas con todos los volúmenes de entrada de cada una de sus K neuronas. Su salida será un volumen [1 × 1 × K], donde cada uno de los K números corresponde a un puntaje de clase, entre las K categorías del conjunto. Después de este tipo de capa no suele tener sentido una capa convolucional.

La función principal de las capas completamente conectadas es llevar a cabo una especie de agrupación de la información que se ha obtenido hasta ese momento para cálculos usados en la clasificación final.

Habitualmente en las redes neuronales convolucionales, se suele utilizar más de una capa completamente conectada en serie y la última de ellas tendrá el parámetro K que es el número de clases que se encuentran presentes en el conjunto de datos. Los valores finales de K serán alimentados a la capa de salida, que a través de una cierta función probabilística realizará la clasificación. Hay más parámetros para configurar, como los valores de los pesos y el sesgo. Es habitual utilizar valores predeterminados en estos parámetros.

2.5.5 - Capa de pérdida. Loss layer

En esta capa es donde se comparan las predicciones con los valores reales de las imágenes. Cuando se trata de clasificar y elegir entre K posibles niveles se usa habitualmente un clasificador de pérdidas softmax. El uso de una función euclídea también es usual con el propósito de regresión contra las etiquetas de las imágenes.

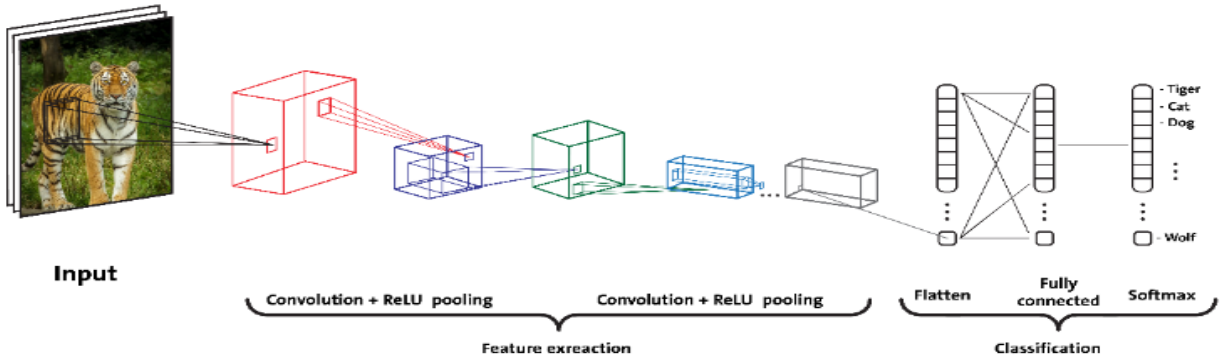


Figura 34. Aspecto final de una ConvNet estándar. Fuente: [27]

2.5.6 Reglas generales para modelar una red convolucional

La capa de entrada , la imagen, debe ser divisible por 2^n. Los más habituales suelen ser 32 (CIFAR-10), 64, 96, 224 (ImageNet), 384 y 512.

Las capas convolucionales deben usar filtros pequeños (entre 3x3 a un máximo de 5x5), usar un salto de S=1. Usar padding para rellenar el volumen de entrada con ceros en los márgenes para que la capa convolucional no altere las dimensiones espaciales de la entrada. El cálculo del padding en general es Padding = (Filtro -1)/2. Por ejemplo, si el Filtro = 3 el padding será de 1, P = (3-1)/2 = 1

La capa de agrupación más habitual es max-pooling con campos receptivos 2x2 (F = 2) y salto de 2 (S = 2).De esta forma se descarta el 75%. Dimensiones de agrupación superiores son muy infrecuentes, son muy agresivas, muy deficitarias y conducen a peores rendimientos.

2.6 - Regularización

Debido a que las redes neuronales disponen de múltiples capas, la capacidad de aproximarse a un determinado objetivo conduce a un sobreajuste. Se denomina sobreajuste al hecho de crear

un modelo tan ajustado a los datos de entrenamiento que haga que no generalice bien a los datos de test, y por tanto aprende patrones que no son generales, creando un modelo no exportable. Para prevenir este sobreajuste es recomendable regularizar los datos de entrada. Sin embargo, este proceso se realiza de forma diferente en el caso de las redes neuronales convolucionales, se suele usar:

2.6.1 - Dropout.

Es una técnica donde una serie de neuronas seleccionadas al azar se ignoran durante el entrenamiento. Esto significa que su contribución a la activación de las neuronas siguientes se elimina temporalmente en la propagación hacia delante y las actualizaciones de la neurona no se aplican a la neurona en la propagación hacia atrás. El efecto es que la red se vuelve menos sensible a los pesos específicos de las neuronas, dando como resultado: una red que generaliza mejor y que tiene un menor sobreajuste de los datos de entrenamiento.

2.6.2 - DropConnect.

Es una generalización de dropout, donde cada conexión es capaz de eliminarse con probabilidad $1 - p$. Cada unidad en esta capa ingresa datos de unidades aleatorias en la capa anterior, que cambian cada iteración. Esto ayuda a garantizar que los pesos no se sobreajusten.

2.6.3 - Agrupación estocástica.

La activación es seleccionada de forma completamente aleatoria. La agrupación estocástica no requiere hiperparámetros y puede usarse como heurística, es decir, con otras técnicas de regularización

2.6.4 - Decadencia de peso.

Funciona de forma similar a las regularizaciones:

- **L1**. El coste agregado es proporcional al valor absoluto de los coeficientes de los pesos
- **L2.** El coste agregado es proporcional al cuadrado del valor de los coeficientes de los pesos. Se penalizan grandes vectores de peso

Dropout es muy utilizado en las redes neuronales convolucionales. Es una técnica efectiva para prevenir el sobreajuste y mejorar la eficiencia computacional. Esta forma de regularización hace que una red de neuronas artificiales se vuelva más pequeña durante la iteración.
La regularización L2 es el método estándar utilizado en las redes neuronales convolucionales, porque tiende a penalizar los pesos demasiado grandes y favorece a aquellos que son pequeños en su proporción con respecto a la totalidad de la matriz.

2.7 - Arquitecturas ConvNet más conocidas

2.7.1 - LeNet [33] 1998

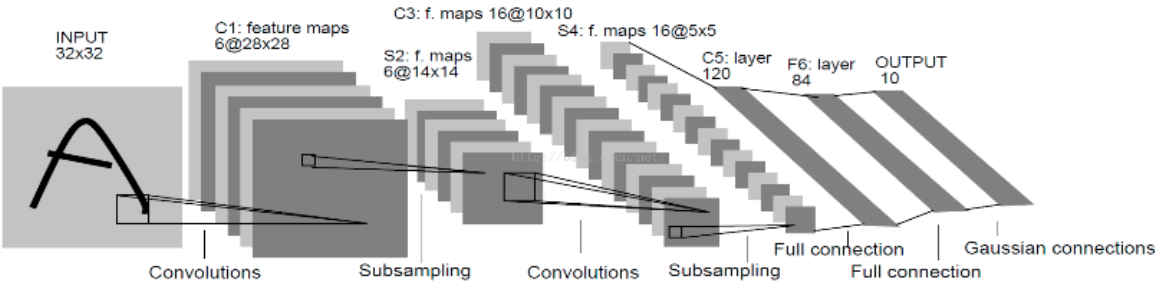


Figura 35. LeNet. Fuente: [33]

Es considerada como el padre de las ConvNet. LeNet fue descrita en 1998 por Lecun et al. [33]. Es una de las primeras redes neuronales convolucionales entrenadas con el algoritmo de retropropagación para el reconocimiento de dígitos de escritura a mano. Esta red toma como entrada una imagen de 32 x 32, pasa a las capas de convolución C1 y luego a la capa de submuestreo S2. Actualmente la capa de submuestreo se reemplaza por una capa de agrupación. Luego, hay otra secuencia de capas de convolución C3 seguida de una capa de agrupación (submuestreo) S4. Finalmente, hay tres capas completamente conectadas, incluida la capa de SALIDA al final. Esta red se utilizó para el reconocimiento de códigos postales en las oficinas de correos. LeNet consta de siete capas entrenables, dos convolucionales, dos de agrupación y tres capas totalmente conectadas.

2.7.2 - AlexNet [34] 2012

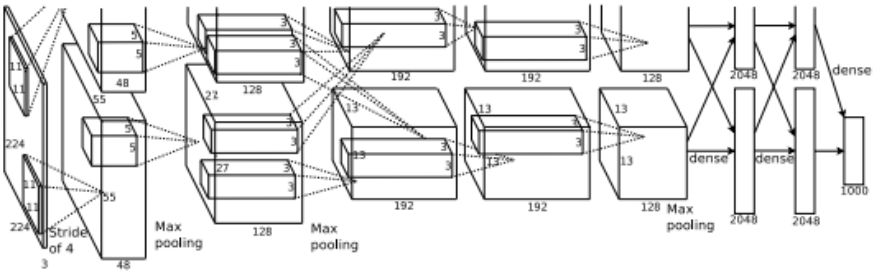


Figura 36. AlexNet. Fuente: [34]

Es un punto de inflexión en las ConvNet. Fue premio en 2012 del conocido desafío ImageNet: ILSVRC-2012 AlexNet [34], desarrollado en la Universidad de Toronto por Alex Krizhevsky y la tutela de Jeffrey Hinton. AlexNet consta de cinco capas convolucionales, algunas de las cuales son seguidas por max-pooling, y tres capas completamente conectadas con 1000 vías softmax, como se muestra en la Figura, entre otras técnicas, como abandono, unidad lineal rectificada (ReLU) y aumento de datos. En la primera ejecución, se utilizó una función de activación de ReLU y una caída de 0,5 para combatir el sobreajuste.
AlexNet usó dos GPU interconectadas, por las limitaciones computacionales de procesamiento existentes en 2012. Actualmente hay GPU muy potentes localmente, además de la computación en la nube. Consta de 8 capas, 5 convolucionales y 3 totalmente conectadas.

2.7.3 - GoogLeNet [36] 2014

El nombre de GoogLeNet proviene porque muchos de sus creadores trabajaban en Google. En 2014 Szegedy et al. Presentaron GoogLeNet [36] y ganaron el primer puesto de ILSVRC14. Su rendimiento es un poco mejor que VGGNet. GoogLeNet es una red de 22 capas que se ejecuta muy rápido debido a la introducción de "Inception", un nuevo módulo de inicio que reduce la cantidad de parámetros a solo 5 millones; eso es 12 veces menos que AlexNet. Tiene menor uso de memoria y menor uso de energía consecuentemente.

2.7.3.1 - El módulo inception

Con el objetivo de reducir sustancialmente el número total de conexiones en la red, para GoogLeNet se creó una nueva estructura denominada módulo inception, a partir de la cual, crearon su nueva arquitectura. Estos módulos sustituyen a las capas vistas en otros modelos de red. La red GoogLeNet consta de nueve de estos módulos

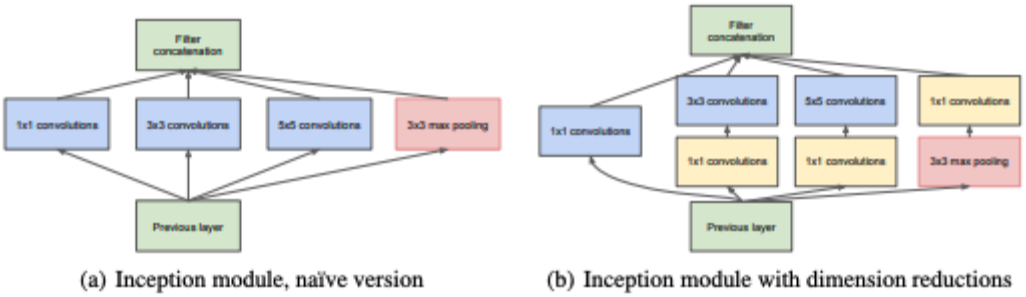


Figura 37. Módulos Inception. Fuente: [37]

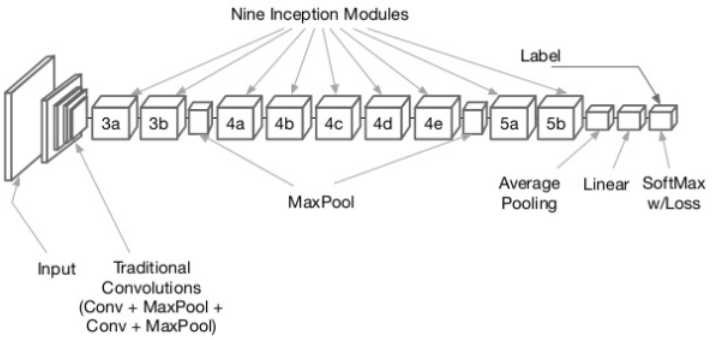


Figura 38. Estructura GoogLeNet. Fuente: [27][36]

En lugar de elegir un tamaño de filtro particular como en las arquitecturas anteriores, GoogLeNet aplica tres filtros de tamaños 1 x 1, 3 x 3 y 5 x 5 en el mismo parche, con una agrupación y concatenación máxima de 3 x 3 en un solo vector de salida. El uso de convoluciones de 1 x 1 disminuye las dimensiones donde el cálculo aumente con las convoluciones de 3 x 3 y 5 x 5. Las convoluciones 1 x 1 con la función de activación ReLU se utilizan antes que las costosas convoluciones 3 x 3 y 5 x 5.

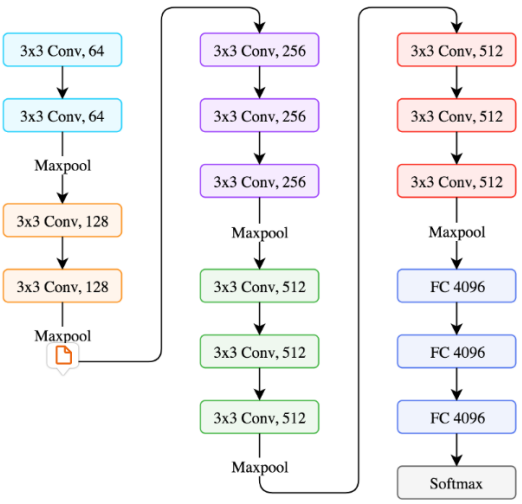
2.7.4 - VGGNet [35] 2014

Fue el subcampeón en el desafío ImageNet 2014, del Grupo de Geometría Visual de la Universidad de Oxford, Simonyan et al.[35]. Tiene dos versiones:

- VGG16 es una red neuronal de 16 capas, sin contar la capa de agrupación máxima y la capa de softmax.
- VGG19 consta de 19 capas. Existe un modelo preentrenado en Keras para Theano y TensorFlow.

Se aumentó la profundidad de la red agregando más capas de convolución, con pequeños filtros de convolución de 3 x 3 en todas las capas y duplicando la cantidad de mapas de características después de la agrupación (Max Pooling 2 × 2). El tamaño de entrada predeterminado de una imagen para este modelo es 224 x 224 x 3. La imagen pasa a través de una pila de capas de convolución con un paso de 1 píxel y un relleno de 1. La agrupación máxima se realiza en una ventana de 2 x 2 píxeles con un paso de 2, luego otra pila de capas de convolución seguidas de tres capas completamente conectadas. Las dos primeras capas completamente conectadas tienen 4096 neuronas cada una, y las terceras capas completamente conectadas son responsables de la clasificación con 1000 neuronas. La capa final es una capa softmax.
La mayoría de estos parámetros se encuentran en la primera capa totalmente conectada, y desde entonces se descubrió que estas capas FC se pueden eliminar sin degradar el rendimiento, lo que reduce significativamente el número de parámetros necesarios.

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					



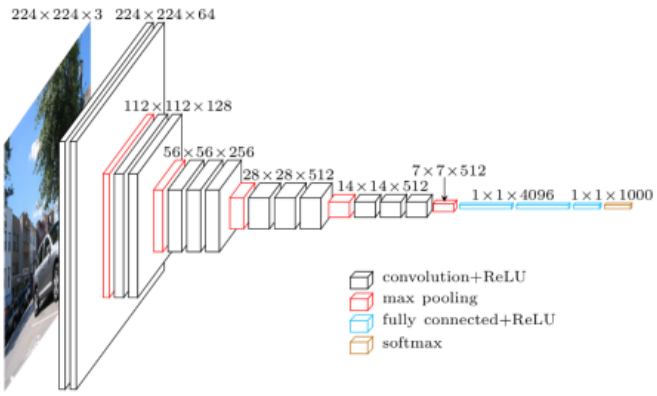


Figura 39. VGGNet16. Fuente: [35][95]

Su principal contribución fue mostrar que la profundidad de la red es un componente crítico para un buen rendimiento. En general en una red neuronal al agregar capas, el rendimiento aumenta sólo hasta cierta profundidad y luego disminuye rápidamente. Esto se debe a que después de

cierta profundidad, agregar capas adicionales y retroalimentar, da como resultado un mayor error de entrenamiento y un mayor error de validación.

2.7.5 - ResNet [38] 2015

En la competencia ImageNet ILSVRC de 2015, el ganador fue ResNet [38]. ResNet es una especie de VGGNet en el sentido de que usa la misma estructura reiteradamente para profundizar la red. A diferencia de VGGNet, tiene diferentes variaciones de profundidad, como 34, 50, 101 y 152 capas. Son muchas capas en comparación con las 8 capas de AlexNet, las 19 capas de VGGNet y las 22 capas de GoogLeNet. La arquitectura ResNet es una pila de bloques residuales. Cuenta con conexiones de salto especiales y un uso intensivo de la normalización por lotes. A la arquitectura también le faltan capas totalmente conectadas al final de la red. ResNets son actualmente modelos de red neuronal convolucional de última generación más utilizados en la práctica.

La idea principal de ResNets es saltar capas agregando conexiones a la red neuronal. Cada bloque residual tiene 3 x 3 capas de convolución. Después de la última capa de conversión, se agrega una capa GAP (Global Average Pooling). GAP es una operación de agrupación diseñada para reemplazar capas completamente conectadas en las ConvNet clásicas. La idea es generar un mapa de características para cada categoría correspondiente de la tarea de clasificación en la última capa. En lugar de agregar capas completamente conectadas sobre los mapas de características, se toma el promedio de cada mapa de características y el vector resultante se introduce directamente en la capa softmax. Una ventaja de la agrupación promedio global, sobre las capas completamente conectadas, es que es más nativo a la estructura de convolución al hacer cumplir las correspondencias entre los mapas de características y las categorías. Por lo tanto, los mapas de características pueden interpretarse fácilmente como mapas de confianza de categorías. Otra ventaja es que no hay ningún parámetro para optimizar en la agrupación promedio global, por lo que se evita el sobreajuste en esta capa. Además, la agrupación promedio global suma la información espacial, por lo que es más sólida para las traducciones espaciales de la entrada. ResNet usa una ConvNet estándar y agrega conexiones que saltan algunas capas de convolución a la vez. Cada bypass da un bloque residual.

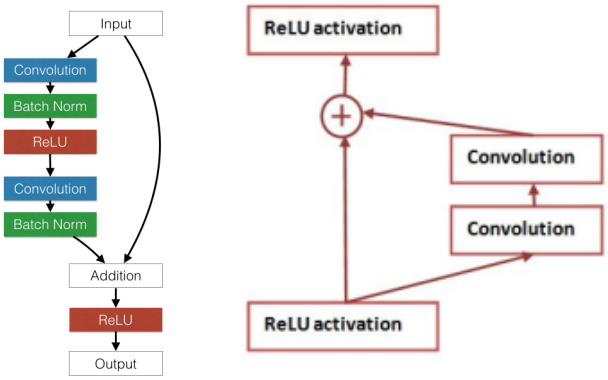


Figura 40. Bloque residual. Fuente: [38]

2.7.6 - SeNet [103] 2017

Hu et al. [103] Senet ganó el ILSVRC -2017 con la presentación de Squeeze-and-Excitation (SE) Networks. El bloque de construcción central de las redes neuronales convolucionales es el operador de convolución, que permite a las redes extraer las características fusionando información espacial y de canal dentro de los campos receptivos locales en cada capa. El bloque SE recalibra dinámicamente la extracción de características a nivel de canal mediante el modelado explícito de interdependencias entre canales. Los bloques SE aportan mejoras significativas en el rendimiento de las ConvNet con un ligero coste computacional

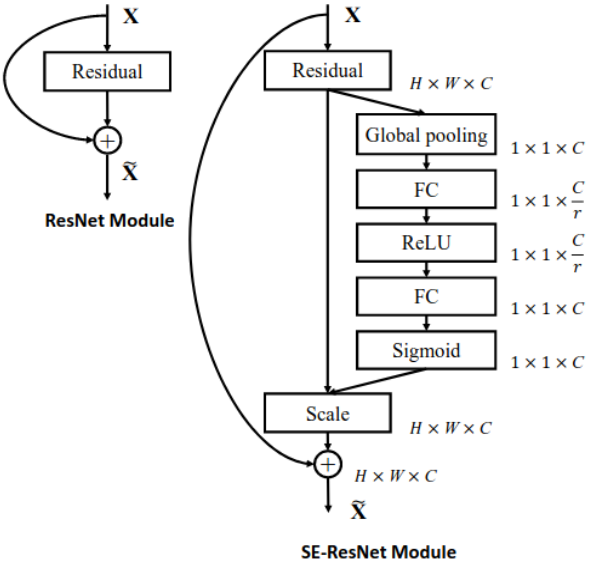


Figura 41. Módulo Residual vs Módulo SE-ResNet. Fuente: [103]

2.8 - Detección de objetos y segmentación con ConvNet

La clasificación clasifica la imagen completa en una de las clases con respecto a la entidad que tiene la máxima probabilidad de detección en la imagen. Cuando el interés no es sobre la totalidad de la imagen sino sobre partes de ella hay que usar el etiquetado en lugar de clases, donde estas etiquetas son todas las clases de la penúltima capa de clasificación Softmax con probabilidad por encima de un umbral determinado. Sin embargo, la probabilidad de detección aquí varía ampliamente según el tamaño, la ubicación (escala), la pose, la luz, entre otras.Esto es lo que se denomina detección de objetos.

La detección de objetos se realiza con cuadros delimitadores y etiquetas de clase junto con la probabilidad de detección de todas las entidades identificadas en una imagen. El resultado de este sistema se puede utilizar para habilitar múltiples casos de uso avanzados que funcionan en la clase específica de los objetos detectados.

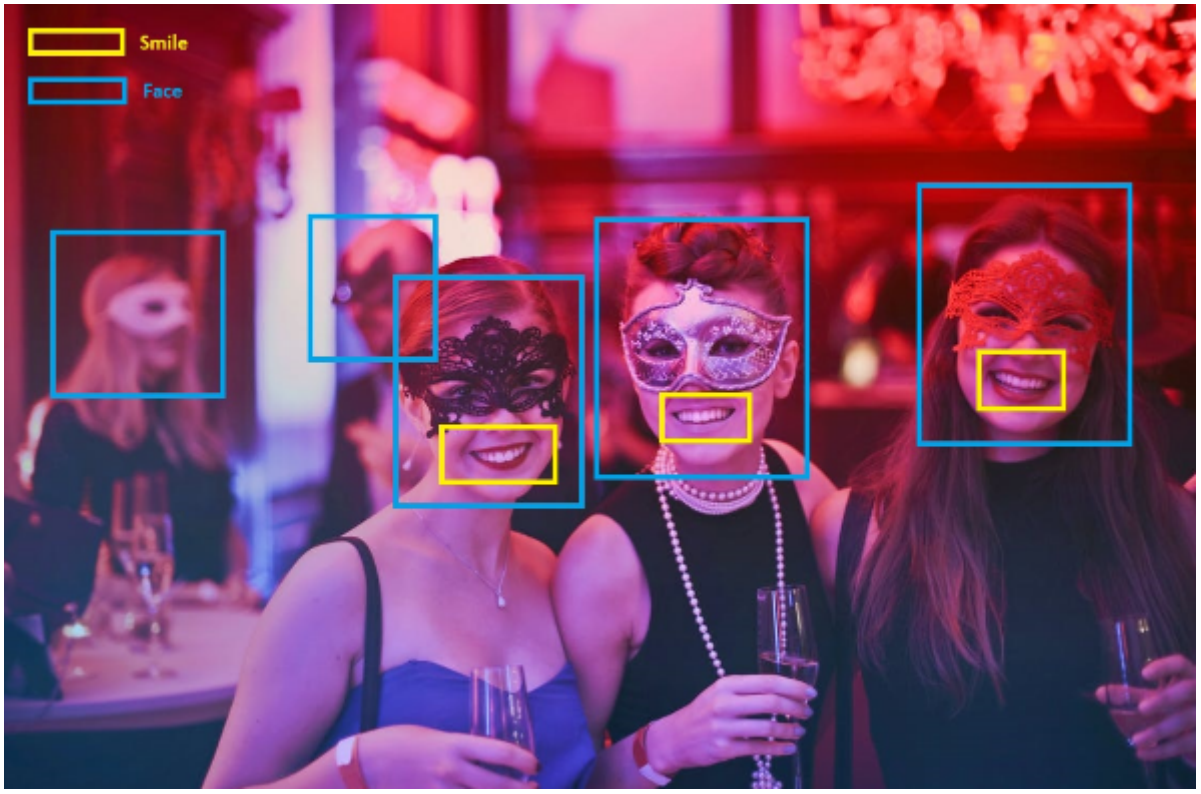


Figura 42. Detección de rostros y sonrisa. Fuente: [27]

Por ejemplo, la función de reconocimiento facial, antes de identificar las personas es necesario detectar todos los rostros en esa imagen y después pasar estos rostros a través del módulo de reconocimiento/clasificación de rostros para obtener/clasificar las identidades. Hay modelos entrenados muy potentes y eficientes de detectores de objetos disponibles para la mayoría de las partes detectables del cuerpo humano (ojo, cara, parte superior del cuerpo, entre otras.), expresiones humanas populares (como una sonrisa) y muchos otros objetos generales. La detección de objetos se consideran retos más desafiantes que la clasificación de imágenes.

Bibliotecas como OpenCV y algunas otras, han tenido una rápida inclusión en los paquetes de software para teléfonos inteligentes, proyectos robóticos y muchos otros para proporcionar capacidades de detección de objetos específicos como rostros, ojos, sonrisas, entre otras. Entre otros filtros incluyen Haar. Las características de Haar o similares a Haar son formaciones de rectángulos con una densidad de píxeles variable. Las características de Haar resumen la intensidad de píxeles en las regiones rectangulares adyacentes en ubicaciones específicas en la región de detección. Basado en la diferencia entre las sumas de las intensidades de píxeles en todas las regiones, clasifican las diferentes subsecciones de la imagen.

3 - Estado de Arte

3.1 - Introducción. Métodos de extracción de características

El reconocimiento facial tiene una canalización clásica descrita por Taigman et al. [45] Con algunas variaciones en sus fases, que todos los sistemas la cumplen en esencia.

Detección > Alineación > Normalización > Representación > Verificación [45]

Una vez detectado un rostro en una imagen, se segmenta ese área para extraer las características que servirán para la detección. La mayoría de los sistemas alinean los rostros de las áreas seleccionadas (detección) con alguna estructura canónica predefinida. Este simple pre-tratamiento mejora mucho la extracción de características, pues coloca los elementos comunes del rostro en una disposición espacial similar y comunes a todas las entradas.

Desde los primeros pasos en la extracción de características se han utilizado diversos métodos:

- **Métodos holísticos.** Se basa en que cualquier conjunto de imágenes faciales contienen redundancias que pueden eliminarse aplicando la descomposición del vector (tensor), extraer una dimensión espacial menor (subespacio) de vectores base.
    - PCA Principal Component Analysis - Eigenfaces (caras propias)
    - LDA Linear Discriminative Analysis - Fisherfaces
    - ICA Independent Component Analysis es una generalización de PCA
    - DWT Discrete Wavelet Transform
    - DCT Discrete Cosine Transform

- **Métodos geométricos.** Se basan en localizar, fijar y focalizar espacios (mapas de prominencia) de los que extraer características. La frente y las mejillas, por ejemplo, tienen estructuras simples y patrones menos distintivos en comparación con la nariz o los ojos. A través de la localización se pueden fijar puntos de referencias comunes a todos los rostros: ojos, punta de nariz, comisuras de la boca, entre otros. Se pueden establecer distancias, ángulos y regiones entre puntos de referencias.
    - EGM elastic graph matching
    - EBGM [96] elastic bunch graph-matching
    - MEGM morphological elastic graph matching
    - D-LGS dense local graph structure

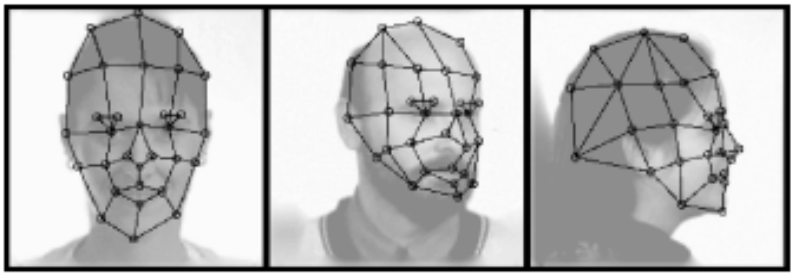


Figura 43. Puntos de referencia. EBGM. Fuente: [96]

- **Métodos Descriptores de Texturas Locales**. Se basa en la extracción de características basadas en el valor del píxel en zonas locales
    - LBP [97] Local Binary Pattern. La imagen facial se divide en diferentes bloques LBP se seleccionan y combinan en un histograma mejorado que se usa como descriptor facial.
    - 1DLBP One-Dimensional Local Binary Pattern. Variación de LBP
    - LPQ local phase quantization. Se basa en la cuantificación en regiones locales de la fase de transformada de Fourier
    - BSIF binarized statistical image features. Entrena automáticamente una colección fija de filtros LBP y/o LPQ a partir de un número limitado de imágenes originales para obtener una forma estadísticamente significativa de las imágenes

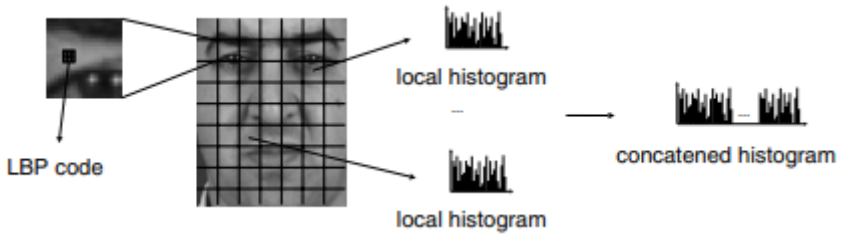


Figura 44. Descripción de la cara LBP con tres niveles de información. Fuente: [97]

- **Métodos de Deep Learning**. Basados en redes neuronales para la extracción de características, a través de las principales arquitecturas o variaciones de ellas:
    - Alexnet [34]
    - VGGNet [35]
    - GoogLeNet [36]
    - ResNet [38]
    - SeNet [103]

El Aprendizaje Profundo es el método más utilizado en el Reconocimiento Facial desde 2014 tras la aparición de Deepface. A continuación se detallaran los principales hitos de esta técnica en el reconocimiento facial

La metodología no siempre es tan homogénea y clasificada como el esquema anterior. Por ejemplo, en muchas ocasiones se emplean métodos geométricos, en el pretratamiento del dataset, previos a los métodos de aprendizaje profundo, como ocurre con DeepFace que alinea y frontaliza los rostros antes de pasarlos por el entrenamiento de la red neuronal profunda.

3.2 - Reconocimento Facial con Deep Learning

El requisito previo para un Reconocimiento Facial con Aprendizaje Profundo es contar con un conjunto de datos de entrenamiento lo suficientemente grande. Zhou et al. [98] sugirió que conjuntos de datos grandes (cuanto mayor mejor) mejoran el rendimiento del Reconocimiento Facial con Deep Learning. En general, los conjuntos de datos públicos cuentan con pocos miles de identidades diferentes (≈10k) en contraposición con los privados, como Google con más de 8 millones de identidades [52].

Para abordar este problema, **CASIA-Webface** / Yi et al. [99] proporcionó el primer conjunto de datos de capacitación pública ampliamente utilizado para el propósito de capacitación profunda, consta de ≈0,5 millones de imágenes de ≈10k celebridades recopiladas de la web. Otros conjuntos de datos muy utilizados actualmente son: **MS-Celeb-1M** / Guo et al. [100] que depurada tiene 3.8M imagenes de 85K identidades, **VGGface2** / Cao et al. [101] que tiene ≈3M imagenes de ≈9K identidades y **Megaface** / Kemelmacher-Shlizerman et al.[102] que tiene 4.7M de imágenes con 672K identidades.

VGGface2 proporciona un conjunto de datos de **entrenamiento en profundidad (muchas imágenes por identidad)**. La profundidad del conjunto de datos obliga al modelo entrenado a abordar una amplia gama de variaciones intraclase, como la iluminación, la edad y la pose.
MS-Celeb-1M proporciona un conjunto de datos de **entrenamiento en amplitud (pocas imágenes por identidad)**. La amplitud del conjunto de datos garantiza que el modelo entrenado cubra suficientemente la variable de personas distintas.
Cao et al. [101] realizó estudios sistemáticos sobre el entrenamiento de modelos usando VGGface2 y MSCeleb-1M, y encontró un modelo óptimo entrenando primero en MS-Celeb-1M (amplitud) y luego afinando en VGGface2 (profundidad).

Dado que todos los rostros humanos comparten una forma y textura similares, la representación aprendida de una pequeña proporción de rostros puede generalizarse bien al resto. Basado en esto, una forma sencilla de mejorar el rendimiento generalizado es incluir tantos ID como sea posible en el conjunto de entrenamiento [98]

Las arquitecturas típicas de ConvNet, AlexNet [34], VGGNet [35], GoogleNet [36], ResNet [38] se usan como modelos de referencia en el Reconocimiento Facial aplicadas directamente o con algunas modificaciones.

Softmax es la función de pérdida más utilizada en la clasificación de imágenes. Es una función que se centra en la separabilidad de las características y se usa como señal de supervisión en el reconocimiento en general, pero **respecto al Reconocimiento Facial no realiza una buena discriminacion de las características faciales**, y por tanto en **diferenciación de rostros**. Esto ocurre porque la diferenciación en el mapa de características faciales son muy próximas entre diferentes identidades. Hay que tener en cuenta que la localización espacial en el mapa de píxeles que sirve como entrada de todas las identidades (tras la frontalización/alineación) son muy parecidas y aproximadas. Cuando se obtiene un mapa de características del rostro, softmax produce a veces más variaciones en una misma identidad que diferencias entre distintas identidades, situación que induce a errores en la identificación. **Surge por tanto la necesidad de crear funciones que sean más discriminatorias en los mapas de características faciales.** Veremos durante el recorrido por el Estado del Arte que **muchos trabajos están enfocados en crear nuevas funciones de pérdida** para **aumentar la separabilidad de características** y que sean **más discriminatorias** de tales características. Muchas de las funciones de pérdidas propuestas son variaciones de softmax para aumentar la separabilidad y discriminacion de características. Otras están basadas en cuantificar la distancia euclídea, en la distancia del coseno y/o en variaciones en los márgenes angulares.

Las primeras redes como Deepface [45] y DeepID [46] usaban la pérdida softmax, pero pronto se denotó las deficiencias discriminatorias explicadas y se comenzó a explorar nuevas funciones de pérdida para diferenciar mejor las características faciales. Entre 2014 a 2017 se usó la pérdida

basada en la distancia euclidiana y tras esto la pérdida basada en el margen angular/coseno, además de la normalización de características y pesos.

Además del **Conjunto de Datos** y la **Distribución** de las Clases dentro del dataset, otro factor importante que afecta al rendimiento obtenido por la red neuronal es la **Función de Pérdida**. Esta es quien dirige el aprendizaje de la red para optimizar los pesos durante el entrenamiento. En la actualidad la mayoría de las funciones de pérdidas utilizadas provienen de variaciones de softmax, agregando una penalización o con modificaciones estructurales de softmax. Principales funciones de pérdidas de este estado del arte:

- **Funciones de Perdida basada en la Distancia Euclidiana**: tienen como objetivo mejorar la capacidad de discriminación de las características, maximizando o minimizando la distancia entre clases.
    - **Función de Pérdida Contrastiva** [47] La Pérdida Contrastiva requiere que la red tome dos tipos de pares de muestras como entradas: los pares de muestras positivas (dos caras de la misma clase) y los pares de muestras negativas (dos imágenes de caras de diferentes clases). Contrastive Loss minimiza la distancia euclidiana de los pares positivos y penaliza los pares negativos que tienen una distancia menor que un umbral
    - **Función de Pérdida de Triplete** [51] Triplet Loss utiliza el triplete como entrada que incluye una muestra positiva, una muestra negativa y un ancla. Un ancla es también una muestra positiva, que inicialmente está más cerca de algunas muestras negativas que de algunas muestras positivas. Durante el entrenamiento, los pares positivos de anclaje se juntan mientras que los pares negativos de anclaje se separan tanto como sea posible. Sin embargo, la selección de los pares de muestras y los tripletes es laboriosa y requiere mucho tiempo tanto para la pérdida de contraste como para la pérdida de triplete.
    - **Función de Pérdida Central** [56] Center Loss agrega una penalización a Softmax al calcular y restringir las distancias entre las muestras dentro de la clase y el centro de la clase correspondiente.
    - **Función de Pérdida de Rango** [55] La Pérdida de Rango calcula las distancias de las muestras dentro de cada clase, y elige el par de dos muestras que tienen el mayor distancia como el intra-clase de restricción; simultáneamente, calcula la distancia de cada pareja de clase de los centros (también conocido como centro de pareja), y obliga a que el centro de par que tiene la menor distancia para tener un mayor margen de que el umbral designado.
    - **Función de Pérdida Marginal** [60] La Pérdida Marginal considera todos los pares de muestras en un lote y obliga a los pares de muestras de diferentes clases a tener un margen mayor que un umbral θ, mientras se obliga a las muestras de la misma clase a tener un margen menor que el umbral θ. Forzar a las dos muestras más alejadas de una clase a tener una distancia menor que la de dos muestras más cercanas de diferentes clases, dificulta la convergencia del procedimiento de entrenamiento.

- **Función de Pérdida basadas en la Distancia del Coseno**
    - **L-Softmax** [57] L-Softmax reformula la salida de la capa softmax de $W \cdot f$ a $|W| \cdot |f| \cdot cos\theta$ para transformar la distancia euclidiana a la distancia del coseno, y también agregar restricciones angulares multiplicativas a $cos\theta$ para ampliar los márgenes angulares entre diferentes identidades

- **L2-Softmax** [59] L2-Softmax Loss restringe la norma L2 del descriptor de característica a un valor constante. L2-Softmax Loss aporta una mejor interpretación geométrica y presta una atención similar a las muestras de rostros de buena y mala calidad.
- **A-Softmax** [64] A-Softmax aplica la normalización de peso, por lo que $W \cdot f$ se reformula a $|f| \cdot \cos\theta$ que simplifica el objetivo de entrenamiento. Sin embargo, después de usar las mismas restricciones angulares multiplicativas, tanto L-Softmax como A-Softmax Loss son difíciles de converger.
- **AM-Softmax** [69] Para mejorar la convergencia de A-Softmax, Wang et al. [69] propone AM-Softmax que reemplaza las restricciones angulares multiplicativas con las restricciones angulares aditivas, es decir, transforma $\cos(m\theta)$ a $\cos\theta - m$. Además, AM-Softmax también aplica la normalización de características e introduce el factor de escala global $s=30$ que hace que $|W| \cdot |f| = s$. Por lo tanto, el objetivo de entrenamiento $|W| \cdot |f| \cdot \cos\theta$ se simplifica de nuevo a $s \cdot \cos\theta$.
- **ArcFace** [86] ArcFace también utiliza las restricciones angulares aditivas, pero cambia $\cos(m\theta)$ a $\cos(\theta+m)$, lo que hace que tenga una mejor interpretación geométrica. Tanto AM-Softmax como ArcFace adoptan la normalización de peso y la normalización de características que restringen todas las características para que se encuentren en una hiperesfera.

Estas Funciones de Pérdida o variaciones de ellas son las más utilizadas por los distintos autores en la última década (2012-2022)

3.3 - Estado del Arte en la última década

3.3.1 - Estado del Arte 2014

DeepFace es presentado por Taigman et al. [45] (Facebook), es un enfoque de varias etapas que utiliza un modelo de forma 3D genérico para alinear caras. Es una red neuronal profunda de 9 capas, entrenada sobre un dataset de más de 4000 identidades y más de 4 millones de imágenes. DeepFace fue uno de los primeros trabajos que logró una precisión muy alta (>97%) en el conjunto de datos LFW utilizando ConvNet. Inspirado por este estudio, el enfoque de la investigación de reconocimiento facial se ha trasladado a enfoques basados en el aprendizaje profundo, en solo tres años, la precisión aumentó drásticamente.

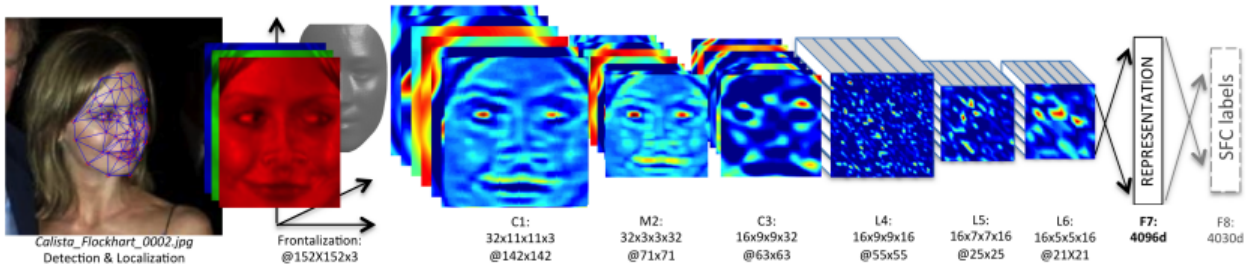


Figura 45. DeepFace. Fuente: [45]

La serie DeepID presentada en varios artículos de Sun et al. [46-48], donde aumentaron constantemente el rendimiento en LFW. En DeepId [45], propusieron las denominadas características de identidad ocultas profundas (DeepID) para aprender las representaciones de características faciales de alto nivel de la tarea de verificación. Las características se obtienen de cada capa convolucional y predicen alrededor de 10.000 clases de identidad en el conjunto de entrenamiento. Esas características se extraen de diferentes regiones faciales para dar forma a representaciones completas y complementarias. Deepid incrementó el rendimiento al 97.45% en LFW [46] con caras débilmente alineadas en comparación con DeepFace [45]
Una de las principales dificultades en el reconocimiento facial es desarrollar una representación de características eficiente para reducir las variaciones intra-personales (dentro de una misma identidad) y aumentar las variaciones inter-personales (entre identidades distintas). La serie DeepId intento realizar esta discriminación con las funciones de verificación de identificación profunda (DeepID2) [47] Las características se aprendieron con variaciones de ConvNet profundas bajo dos señales de supervisión:

- Señal de identificación: aumentando las variaciones inter-personales separando las derivadas de identidades distintas.
- Señal de verificación: disminuyendo las variaciones intra-personales juntando las derivadas de la misma identidad.

La fusión de estas dos señales de supervisión da como resultado características mucho mejores que cualquiera de las dos. Sun et al. presentaron en [48] una red neuronal convolucional profunda de alto rendimiento (DeepID2+) para el reconocimiento facial mejorando a DeepID2 [47], aumentando la dimensión de las representaciones ocultas y uniendo la supervisión a las primeras capas convolucionales.

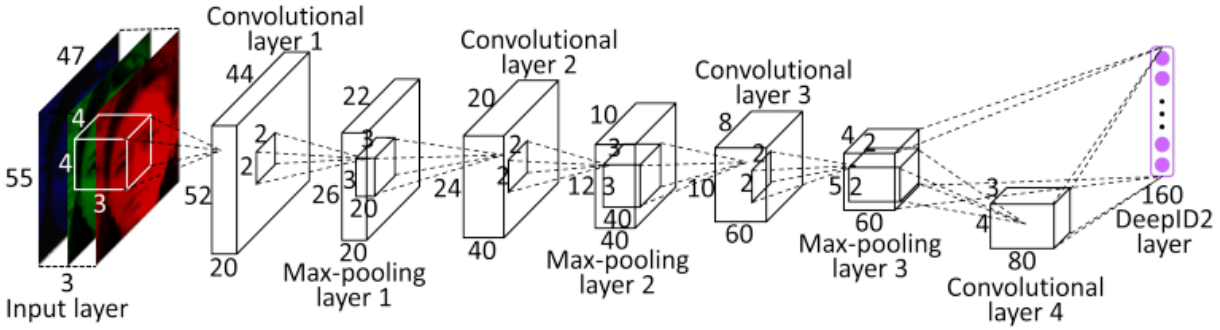


Figura 46. Arquitectura de DeepID2. Fuente:[47]

3.3.2 - Estado del Arte 2015

A finales de 2014 y principios de 2015, de nuevo Sun et al. [49] propusieron dos arquitecturas de redes neuronales más profundas para el reconocimiento facial denominada DeepID3, más profunda que DeepID2+. Estas dos arquitecturas se reconstruyen a partir de elementos cruciales de GoogLeNet [52] y VGGNet [53], capas de convolución e inicio apiladas. Se agregaron señales de supervisión de verificación de identificación facial conjunta durante el conjunto de entrenamiento tanto para la extracción de características finales como para las capas intermedias. Alcanzaron una precisión del 99,53% en LFW.
Taigman et al. [50] aplicaron un método de arranque semántico para reemplazar el submuestreo aleatorio ingenuo (naive Bayes) [50] del conjunto de entrenamiento para seleccionar de una extensa base de datos un conjunto de entrenamiento eficiente.

Liu et al. [51] propusieron un enfoque de dos etapas para el reconocimiento facial, que extraía características de baja dimensión pero alta discriminación mediante la fusión de una ConvNet profunda multiparche con aprendizaje métrico profundo. Encontraron que los parches locales son menos sensibles a la variación, principalmente expresiones y poses.
FaceNet fué un modelo GoogLeNet propuesto por Schroff et al. [52]. Implementaron una función de pérdida de tripletes logrando un rendimiento del 99,63 %. Se entrenó en un dataset de más de 200 millones de imágenes faciales y 8 Millones de identidades. El triplete constaba de dos parches faciales coincidentes y un parche facial no coincidente

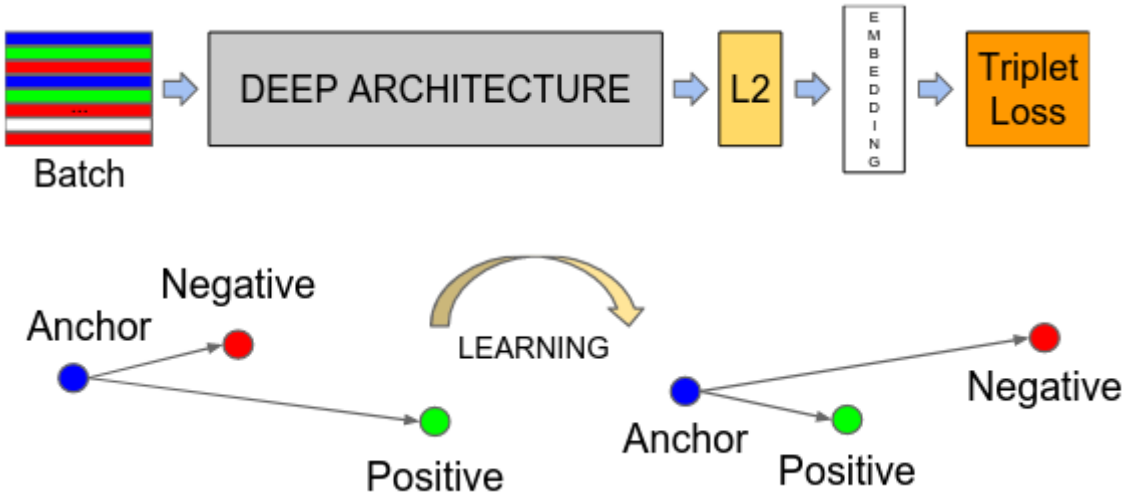


Figura 47. FaceNet. Fuente: [52]

La función Triplet Loss:

- Misma Identidad: disminuye la distancia entre el ancla (anchor) y un positivo
- Distinta Identidad: incrementa la distancia entre el ancla y un negativo

3.3.3 - Estado del Arte 2016

Parkhi et al. [53] realizaron dos contribuciones:

- Desarrollaron un método para recopilar un conjunto de datos a gran escala de internet (VGGface) con poco ruido de etiqueta.
- Trabajaron diversas arquitecturas de ConvNet para el reconocimiento facial basadas en VGGNet [53] con la exploración de una función de pérdida de triplete y alineación facial.

Para permitir una comparación directa usaron su dataset VGGFace para el entrenamiento y la evaluación se hicieron sobre base de datos de referencia LFW.

Masi et al. [54] aumentaron sus datos de entrenamiento al generar nuevas imágenes faciales con variaciones de apariencia específicas, que incluían forma, pose y expresión usando una cara genérica en 3D.

Figura 48. Variaciones Faciales Fuente:[54]

El aumento de datos tiene como objetivo aumentar el conjunto de datos al realizar transformaciones en las imágenes sin cambiar las etiquetas. Es una técnica que se ha utilizado habitualmente para mejorar el rendimiento de las ConvNet y evitar el sobreajuste.

La distribución de cola larga es un término estadístico que se refiere específicamente al hecho de que un pequeño número de características genéricas aparecen con frecuencia mientras que otras específicas aparecen mucho menos. Para investigar el efecto de cola larga en el reconocimiento facial profundo, Zhang et al. [55] propusieron una nueva función denominada Pérdida de Rango que utiliza el valor medio armónico para reducir las diferencias intra-personales y, al mismo tiempo, ampliar las diferencias inter-personales.

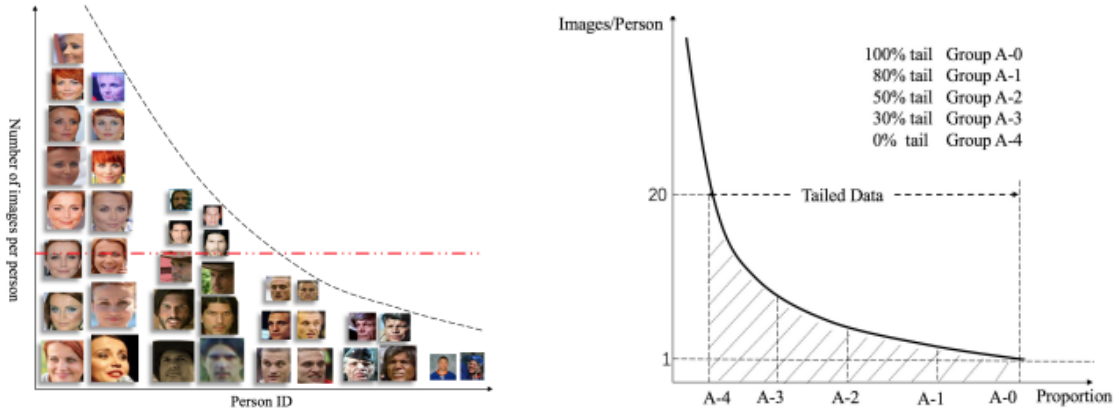


Figura 49. Gráficas Tail. Fuente: [55]

Liu et al. [57] propusieron una pérdida Softmax de margen grande generalizada (L-Softmax), que combina los componentes más utilizados en las arquitecturas ConvNet profundas, la pérdida de entropía cruzada, la pérdida Softmax y la capa final completamente conectada. Lograron aumentar el rendimiento de la verificación y la clasificación.

Wen et al. [56] fueron los pioneros en la introducción de una señal de supervisión que denominaron Pérdida Central. Esta función de pérdida extrae, para cada clase, un centro para las características faciales, reduce las distancias entre las características y los centros de clase coincidentes. Por lo tanto, se mejora la discriminación de las características faciales aprendidas y se minimizan las variaciones en la característica intraclase.

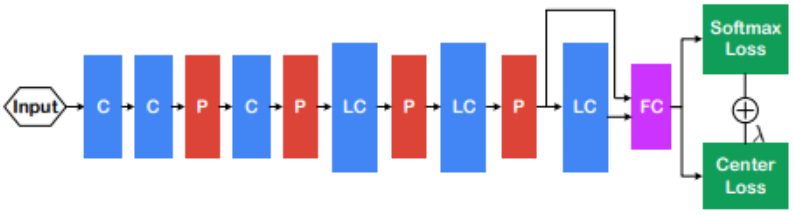


Figura 50. Center Loss. Fuente: [56]

3.3.4 - Estado del Arte 2017

Wu et al. [58] propusieron una función de pérdida invariante del centro, que alineaba el centro de cada identidad para hacer cumplir las características faciales extraídas y tener representaciones más genéricas de todas las identidades.

Ranjan et al. [59] propusieron la normalización de características y la restricción L2 en la pérdida de Softmax para entrenar la verificación facial. La pérdida L2-Softmax realiza un aprendizaje de características compacto al restringir las características profundas para que se encuentren dentro de la hiperesfera unitaria.

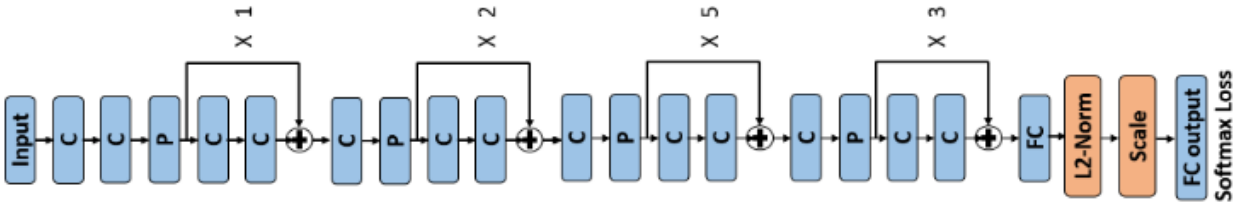


Figura 51. L2-Softmax. Fuente: [59]

Deng et al. [60] propusieron la función de pérdida marginal, aumentando las separaciones entre clases y reduciendo las variaciones intraclase con la supervisión conjunta de Softmax y la pérdida marginal, para aumentar el poder discriminativo de las características profundas.

Wang et al. [61] propusieron la pérdida contrastiva, NormFace, para mejorar la verificación facial. Aplicaron operaciones de normalización L2 en las incrustaciones y los vectores de peso de la capa de salida antes de Softmax.

Liu et al. [62] propusieron una pérdida de aprendizaje métrica denominada COCO (congenerous cosine) para el reconocimiento individual. Optimizaron y compararon la distancia del coseno entre rasgos profundos, haciendo que la pérdida tuviera una variación intraclase máxima menor que la distancia interclase mínima.

Hasnat et al. [63] propusieron modelar el aprendizaje de características profundas como una combinación de distribuciones de von Mises-Fisher, mediante la integración de modelos mixtos de von Mises-Fisher (vMF) con el modelo de ConvNet profunda. Crearon una nueva función de pérdida denominada vMF-ML que permitía el aprendizaje discriminativo.

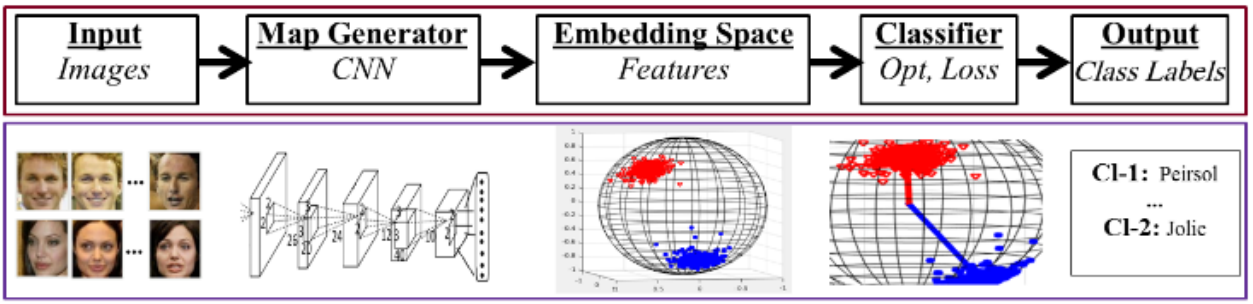


Figura 52: vMF-FL. Fuente: [63]

3.3.5 - Estado del Arte 2018

Liu et al. [64] propusieron una pérdida Softmax angular (A-Softmax) que permite a las redes neuronales convolucionales (ConvNet) aprender características angularmente discriminativas. Geométricamente, la pérdida A-Softmax puede considerarse como la imposición de restricciones discriminativas en un colector hiperesférico que denominaron Deep Hypersphere Embedding - SphereFace

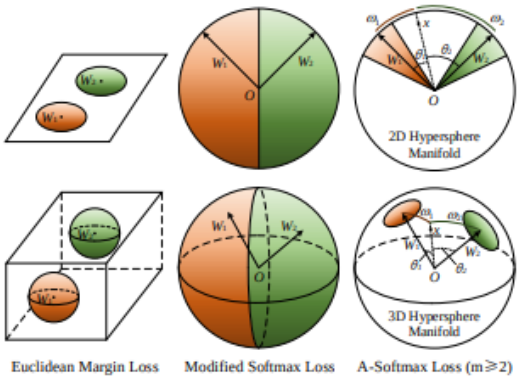


Figura 53: A-Softmax. Fuente: [64]

Zheng et al. [65] propusieron la normalización suave de características mediante una función de pérdida denominada Pérdida de Anillo. Normalizaron todas las muestras de características faciales a través del aumento convexo de la función de pérdida estándar (como Softmax). La red aprende al restringir los vectores de características faciales en la hiperesfera unitaria.

Wang et al. [68] propusieron una función de pérdida denominada pérdida de coseno de gran margen (LMCL) para aprender características más discriminatorias para el reconocimiento facial. Reformularon la pérdida Softmax como una pérdida de coseno al normalizar L2, tanto en las características como en los vectores de peso para eliminar las variaciones radiales. En consecuencia, para optimizar el margen de decisión en el espacio angular, introdujeron un margen de coseno, CosFace. La varianza intraclase mínima y la varianza interclase máxima se logran en virtud de la normalización y la maximización del margen de decisión del coseno.

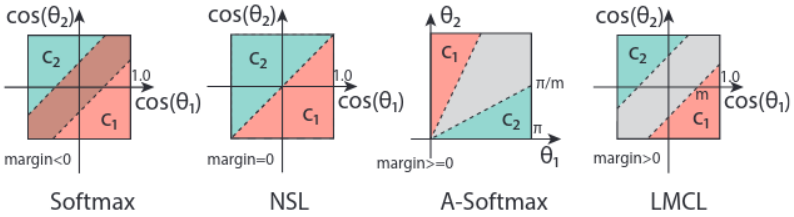


Figura 54. LMCL de CosFace. Fuente:[68]

Wang et al. [69] propusieron imponer un margen aditivo a la pérdida de Softmax, AM-Softmax. El margen se formuló a través de una similitud de coseno con pesos y características normalizados. De esta forma se extraían mejores representaciones de rostros del aprendizaje.

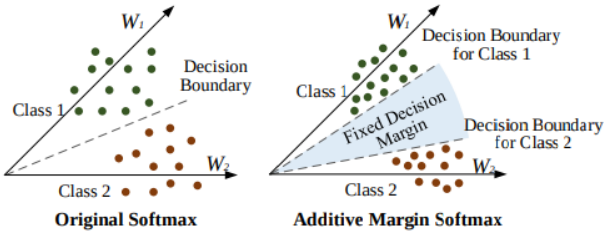
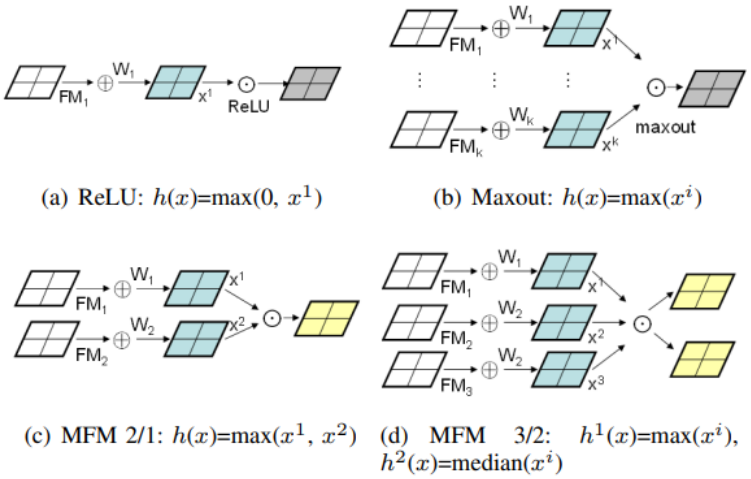


Figura 55: AM-Softmax. Fuente: [69]

Wu et al. [70] Propusieron una red convolucional ligera Light-ConvNet para conjuntos de datos masivos con etiquetas ruidosas. Los autores introdujeron por primera vez max-feature-map (MFM) en cada capa convolucional. MFM no solo separa señales ruidosas de las señales de

información, también selecciona características entre dos mapas de características. También propusieron tres redes reduciendo costes computacionales y el número de parámetros. Así mismo, presentaron un bootstrapping semántico que predice qué red es más consistente con etiquetas ruidosas.



(a) ReLU: $h(x)$=max(0, x^1) (b) Maxout: $h(x)$=max(x^i)

(c) MFM 2/1: $h(x)$=max(x^1, x^2) (d) MFM 3/2: $h^1(x)$=max(x^i), $h^2(x)$=median(x^i)

Figura 56. Comparativa Relu / Maxout / MFM. Fuente: [70]

3.3.6 - Estado del Arte 2019

Para abordar el aprendizaje desequilibrado en clase utilizando ConvNet profundo, Hayat et al. [71] propusieron la primera función de pérdida híbrida basada en una medida de afinidad en el espacio euclidiano para crear un clasificador de gran margen generalizable. La pérdida propuesta combina el agrupamiento y la clasificación en una sola formulación, que reduce las variaciones dentro de la clase y al mismo tiempo logra distancias máximas entre clases.

Para mejorar la discriminación en el reconocimiento facial, Deng et al. [86] propusieron ArcFace, una pérdida de margen angular aditiva, con una interpretación geométrica optimizada que mejora el margen de distancia geodésica al hacer coincidir el arco y el ángulo en la hiperesfera normalizada. Muchos datasets durante el análisis de rostros para el reconocimiento facial y la predicción de atributos de rostros (características), muestran una distribución de clases desequilibrada, la mayoría de los datos pertenecen a algunas clases mayoritarias. Por el contrario, las clases minoritarias a menudo solo tienen unas pocas instancias

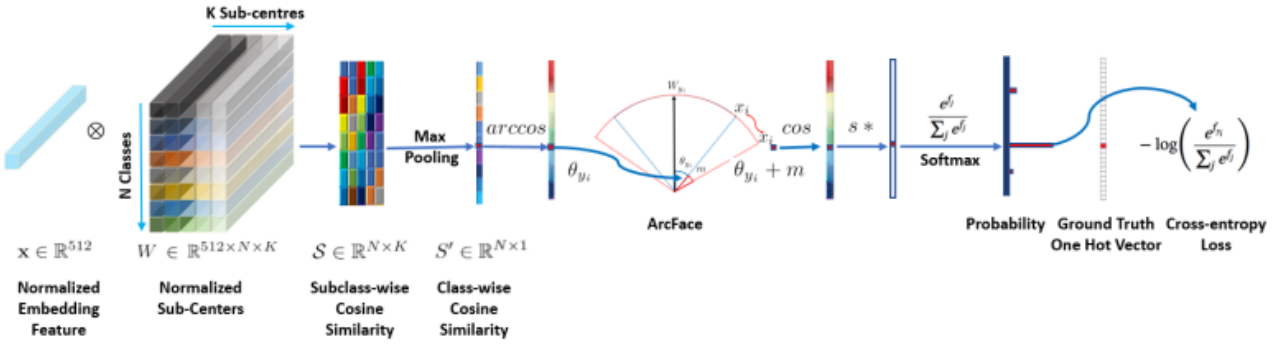


Figura 57:Training ArcFace. Fuente: [86]

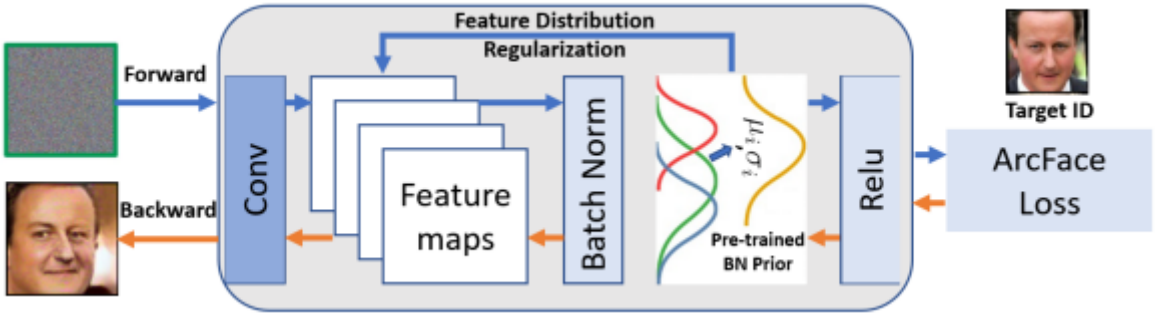


Figura 58. ArcFace. Modelo discriminativo / generativo. Fuente: [86]

Para acometer y mitigar estas deficiencias en la distribución de los datos Huang et al. [72] estudiaron la efectividad de los esquemas de estrategias existentes para equilibrar datos. Demostraron que se puede extraer una representación profunda más discriminatoria al hacer cumplir que la red mantenga los márgenes entre grupos, tanto dentro del grupo como entre las clases. El enfoque propuesto, conocido como incrustación local de gran margen basado en grupos (CLMLE), mantiene los márgenes angulares, tallando así límites de clase localmente más equilibrados. Yin et al. [75] también trataron el desequilibrio de los datos de entrenamiento. Propusieron un aprendizaje de transferencia de características (FTL) que adapta la distribución de características de las clases subrepresentadas (UR), logrando un entrenamiento menos sesgado.

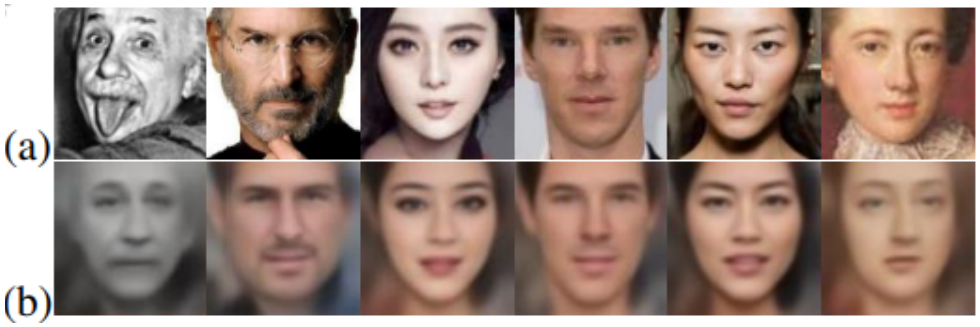


Figura 59. Clases subrepresentadas. Fuente:[75]

Las oclusiones dificultan el reconocimiento facial, Song et al. [73] propusieron para evitar las oclusiones en áreas faciales variables. Presentaron un marco de red siamesa diferencial por pares (PDSN) para encontrar la correspondencia entre los elementos de características corruptos y los bloques faciales ocluidos para modelos de ConvNet profundos que dan como resultado un sistema robusto de reconocimiento facial bajo oclusiones.

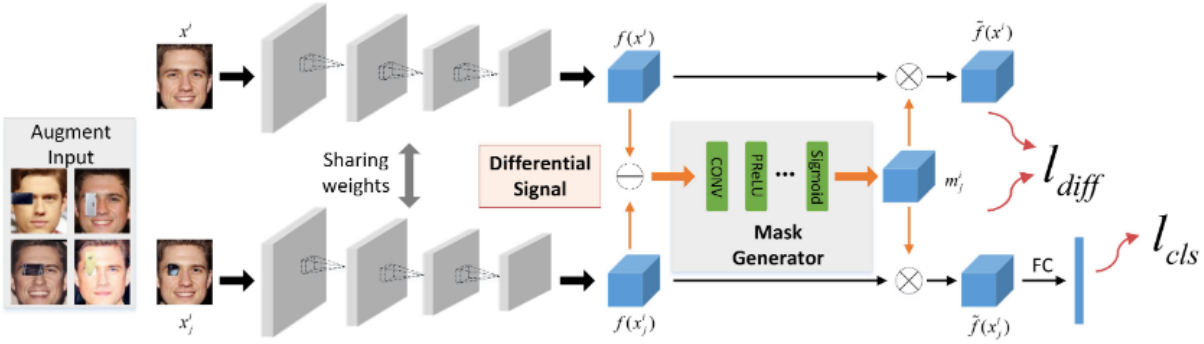


Figura 60. Pairwise Differential Siamese Network. Fuente: [73]

3.3.7 - Estado del Arte 2020

Ben Fredj et al. [76] usaron un aumento de datos agresivo con información perturbadora al azar y condiciones complicadas de apariencia facial. Una de las ideas principales fue utilizar la estrategia de fusión adaptativa de softmax loss y center loss, mejorando el rendimiento y haciendo que el modelo sea más flexible y eficiente.

Wei et al. [67] propusieron resolver el problema del sesgo de margen introduciendo un margen mínimo para pares completos de clases. Presentaron una función de pérdida llamada pérdida de margen mínimo (MML), cuyo objetivo es ampliar el margen de los pares de centros de clase que esten demasiado cercanos para mejorar la capacidad de discriminación de las características profundas. MML supervisa el proceso de entrenamiento junto con center loss y Softmax loss para equilibrar todos los márgenes de clase independientemente de sus distribuciones de clase.
Sun et al. [77] sugirieron una función de pérdida denominada pérdida de margen angular entre clases (IAM), destinada a aumentar la variación entre clases de forma adaptativa al penalizar los ángulos entre clases más pequeños y lograr que el margen angular sea más grande entre clases, lo que puede aumentar significativamente la discriminación de los rasgos faciales. La pérdida IAM pretende actuar como un término de regularización para la pérdida Softmax habitualmente utilizada y sus variaciones recientes. Además, los autores proporcionaron un análisis del rango de hiperparámetro de regularización y sus efectos.

Wu et al. [79] investigaron el impacto de los errores de cuantificación en el reconocimiento facial y propusieron la pérdida del margen constante de rotación (RCM) para el entrenamiento eficiente del reconocimiento facial de low-bit al minimizar los errores individuales, necesarios para presentar poder discriminativo.

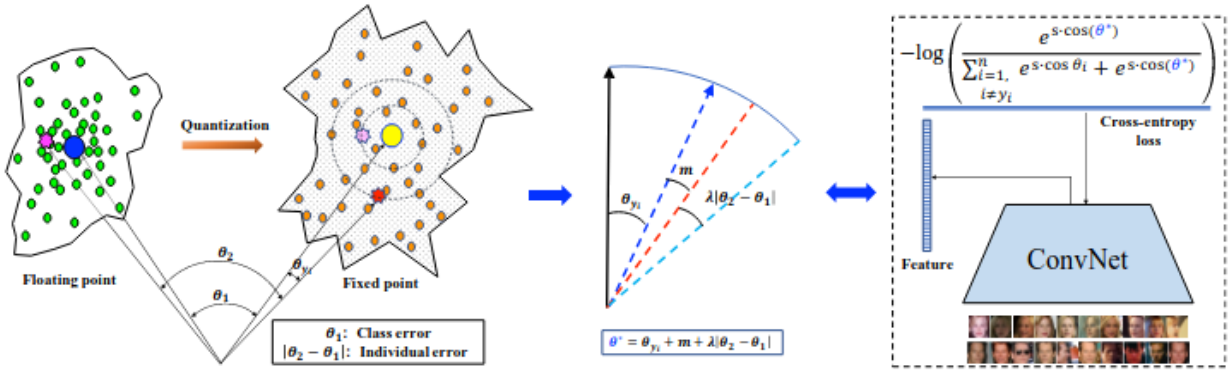


Figura 60. RCM Loss. Fuente: [79]

Para aprender las relaciones de características globales de las imágenes faciales alineadas, Ling et al. [78] propusieron una red neuronal basada en la atención (ACNN) para incorporar rasgos faciales discriminativos. Con esto querían reducir la redundancia de información entre canales y concentrarse en los componentes más informativos de los mapas de rasgos faciales. El módulo de atención está compuesto por dos bloques denominados bloque de atención de canal y bloque de atención espacial.

Para eliminar la gran variación intraclase de la pérdida de softmax, Wu y Wu [80] introdujeron la restricción de la similitud del coseno en el proceso de entrenamiento. Se han propuesto dos funciones de pérdida útiles denominadas coseno de gran margen (LMC) y coseno de gran

margen discriminativo (DLMC). LMC impone la similitud de coseno intraclase entre una muestra y el vector de peso correspondiente en la última capa de producto interno por encima de un margen dado. DLMC mantiene la separabilidad entre clases y la compactabilidad intraclase simultáneamente en el espacio de características normalizado. Las funciones de pérdida propuestas pueden mejorar la discriminabilidad. Específicamente, como un coseno de gran margen discriminativo especializado (SDLMC), que ha demostrado ser una variante de la pérdida de triplete y presenta la ventaja intrínseca sobre el problema de la verificación facial.

Huang et al. [81] estudiaron cómo el diseño de funciones de pérdida basadas en el margen puede aumentar el margen de características entre diferentes clases para mejorar la discriminabilidad. Adoptaron estrategias basadas en la minería para enfatizar las muestras mal clasificadas, logrando buenos resultados. Los autores propusieron una pérdida denominada CurricularFace, que ajusta de manera adaptativa la importancia relativa de las muestras fáciles y difíciles durante las diferentes etapas del entrenamiento

Loss	Decision Boundary
Softmax	$\cos\theta_{y_i} = \cos\theta_j$
SphereFace	$\cos(m\theta_{y_i}) = \cos\theta_j$
CosFace	$\cos\theta_{y_i} - m = \cos\theta_j$
ArcFace	$\cos(\theta_{y_i} + m) = \cos\theta_j$
MV-Arc-Softmax	$\cos(\theta_{y_i} + m) = \cos\theta_j$ (easy) $\cos(\theta_{y_i} + m) = t\cos\theta_j + t - 1$ (hard)
CurricularFace (Ours)	$\cos(\theta_{y_i} + m) = \cos\theta_j$ (easy) $\cos(\theta_{y_i} + m) = (t + \cos\theta_j)\cos\theta_j$(hard)

Figura 61: Límites de decisión. Funciones de Pérdida habituales . Fuente: [81]

3.3.8 - Estado del Arte 2021

Kim et al. [82] propusieron una nueva arquitectura en reconocimiento facial que denominan GroupFace. Utiliza múltiples representaciones del grupo de reconocimiento, simultáneamente, para mejorar la calidad de la función de incrustación. El método propuesto proporciona etiquetas auto distribuidas que equilibran el número de muestras pertenecientes a cada grupo sin anotaciones humanas adicionales, y aprende las representaciones del grupo, reduciendo el espacio de búsqueda de la identidad objetivo.

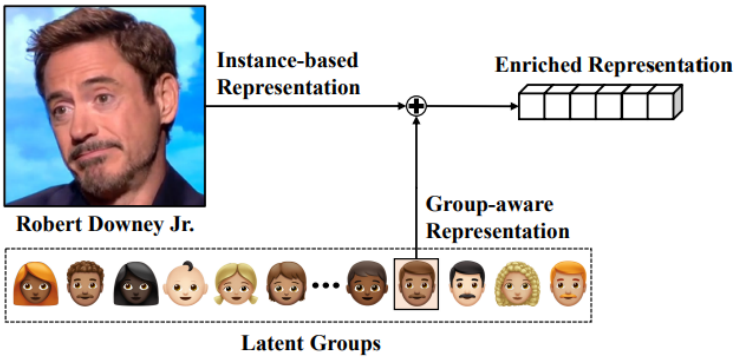


Figura 62. Group-aware. Fuente: [82]

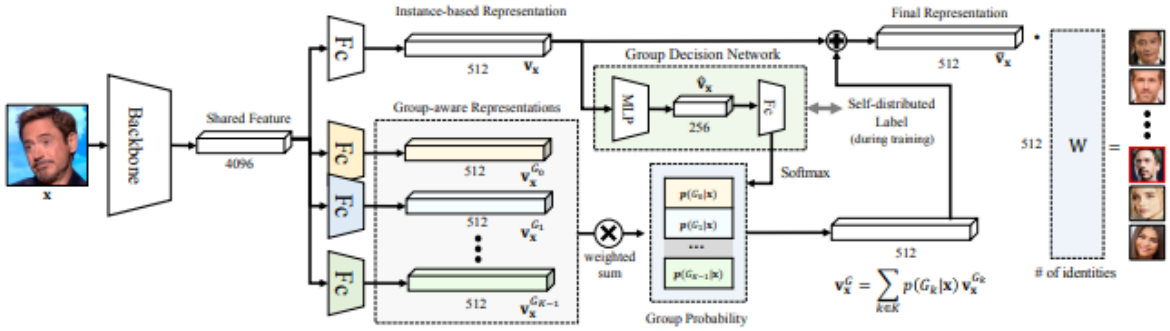


Figura 63. GroupFace. Fuente: [82]

Jiao et al.[83] investigaron que cuando se utilizaban las funciones de pérdida se producía un sobreajuste del conjunto de datos de entrenamiento y se reducía el aprendizaje. Propusieron una función de pérdida llamada Dyn-arcFace (Dynamic Additive Angular Margin Loss for Deep Face Recognition) para reducir el grado de sobreajuste causado por el margen angular aditivo fijo

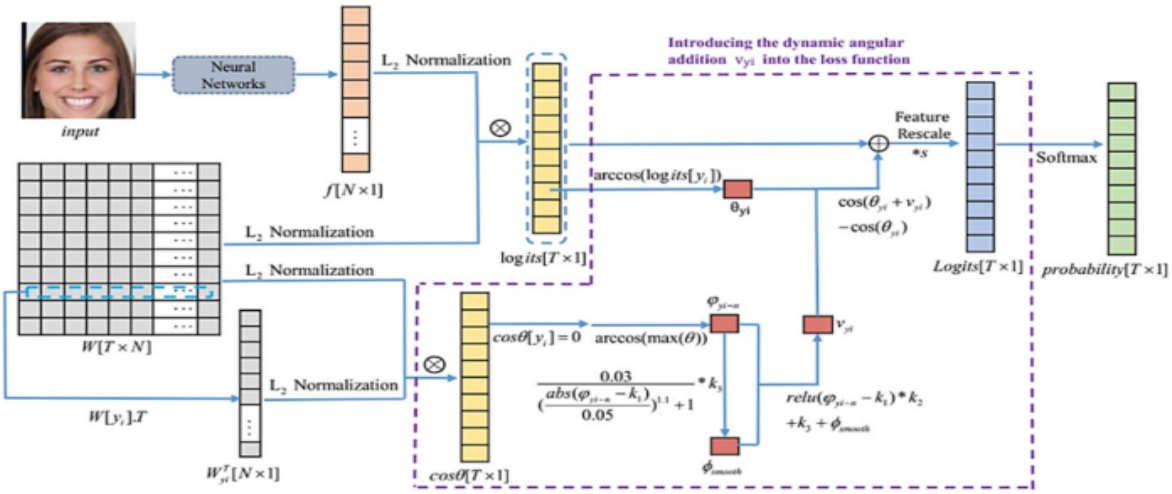


Figura 64. Dyn-arcFace loss. Fuente: [83]

3.3.9 - Estado del Arte 2022

Boutros et al. [84] indican que las pérdidas softmax de penalización marginal, como ArcFace y CosFace, asumen que la distancia geodésica entre y dentro de las diferentes identidades se puede aprender igualmente utilizando un margen de penalización fijo. Los autores proponen relajar la restricción del margen de penalización fijo y crear una pérdida de margen de penalización elástica que denominan ElasticFace. La idea principal es utilizar valores de margen aleatorios extraídos de una distribución normal en cada iteración de entrenamiento. En los experimentos muestran la superioridad de ElasticFace sobre las pérdidas ArcFace y CosFace, utilizando la misma transformación geométrica, en un gran conjunto de datasets de referencia principales. Según sus autores ElasticFace mejora el rendimiento de reconocimiento facial de última generación en siete de los nueve principales datasets de referencia: LFW [43], AgeDB-30 [40], Cross-age LFW (CALFW) [41], Cross-Pose LFW (CPLFW) [42], Celebrities in Frontal-Profile in the Wild (CFP-FP) [113], IARPA Janus Benchmark-B (IJB-B) [114], IARPA Janus Benchmark-C (IJB-C) [115], MegaFace [102], and MegaFace (R) [117].

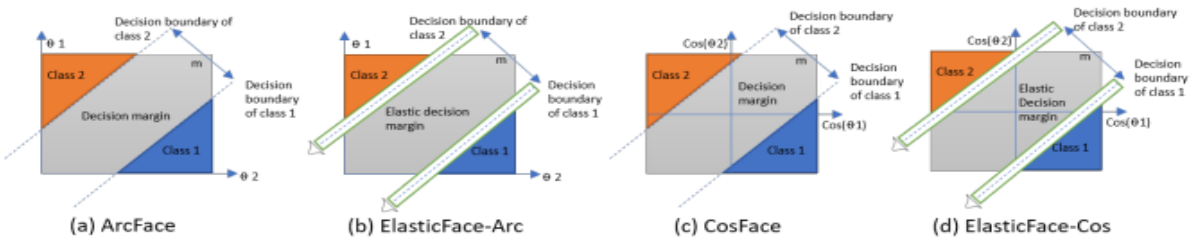


Figura 65. Decision boundary. Fontera de Desicion. Fuente: [84]

Hun et al. [85] presentan una función de pérdida denominada pérdida ortante aditiva (Orthant loss), que se puede combinar con funciones de pérdida basadas en softmax para mejorar la capacidad de discriminación de características. La pérdida de Orthant hace que las características se alejen del origen utilizando la función softmax reescalada y un margen aditivo. Los autores indican que ArcFace combinado con la pérdida de Orthant comprime y aleja aún más los espacios de características del origen en comparación con ArcFace

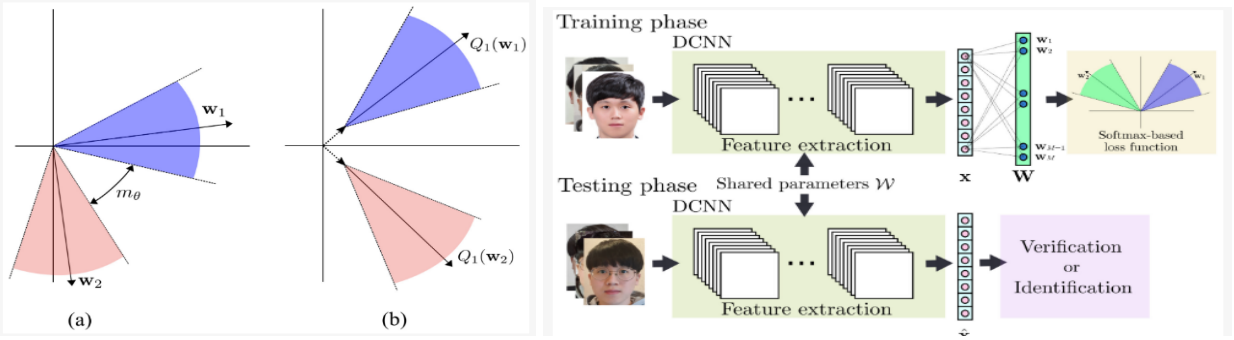


Figura 66 Softmax (a) sin y (b) con Orthant Loss. Fuente: [85]

Sang et al. [87] proponen una nueva función de pérdida, InterFace, liberando la restricción de agregar una penalización de margen solo entre la característica de profundidad y el peso correspondiente para impulsar la separabilidad de las clases al agregar las penalizaciones de margen correspondientes entre las características de profundidad y todos los pesos. InterFace mejora el rendimiento del reconocimiento facial de última generación en cinco de los trece puntos de referencia principales.

Method	Training Dataset	LFW Accuracy(%)
Arcface [5]	MS1MV2 [5], [7]	99.82(3)
CosFace [24]	private	99.73
Dynamic-AdaCos [28]	MS1MV2 [7], [28] + CASIA [27]	99.73
AdaptiveFace [14]	clean MS1M [5], [26]	99.62
UniformFace [6]	MS1MV2 [5], [7] + VGGFace2 [3]	99.8
GroupFace [13]	MS1MV2 [5], [7]	**99.85**(1)
CircleLoss [23]	clean MS1M [7], [23]	99.73
CurricularFace [10]	MS1MV2 [5], [7]	99.80
Dyn-arcFace [11]	MS1MV2 [5], [7]	99.80
MagFace [18]	MS1MV2 [5], [7]	99.83(2)
Partial-FC-ArcFace [1]	MS1MV2 [5], [7]	99.83(2)
Partial-FC-CosFace [1]	MS1MV2 [5], [7]	99.83(2)
ElasticFace-Arc [2]	MS1MV2 [5], [7]	99.82(3)
ElasticFace-Cos [2]	MS1MV2 [5], [7]	99.82(3)
InterFace$_{cid\&ct}$	MS1MV2 [5], [7]	99.83(2)
InterFace$_{did\&ct}$	MS1MV2 [5], [7]	99.80
InterFace$_{did\&dt}$	MS1MV2 [5], [7]	99.83(2)

Figura 67: InterFace. Comparativa. Fuente: [87]

Tabla. Estado del Arte en el Reconocimiento Facial con ConvNet desde 2014

Método	Año	Arquitectura	Núm. Redes	Función Pérdida	Dataset	Accuracy %		Autores
						LFW	YTF	
DeepFace	2014	CNN-9	3	Softmax	Facebook (4.4 M, 4 K) *	97.35	-	Taigman et al. [45]
DeepId	2014	CNN-9	60	Softmax	CelebFaces + [116] (202 k, 10 k) *	97.45	-	Sun et al. [46]
DeepID2	2014	CNN-9	25	Contrastive Loss	CelebFaces+ (202 k, 10 k) *	99.15	-	Sun et al. [47]
DeepID2+	2014	CNN-9	25	Contrastive Loss	WDRef [153] + CelebFaces + (290 k, 12 k) *	99.47	93.2	Sun et al. [48]
DeepID3	2015	VGGNet-10	50	Contrastive Loss	WDRef + CelebFaces + (290 k, 12 k)	99.53	-	Sun et al. [49]
Web-Scale	2015	CNN-9	4	Contrastive Loss	Private Database (4.5 M, 55 K) *	98.37	-	Taigman et al.[50]
Targeting Ultimate Acc - Baidu	2015	CNN-9	10	Triplet Loss	Private Database (1.2 M, 18 K) *	99.77	-	Liu et al. [51]
FaceNet	2015	GoogLeNet-22	1	Triplet Loss	Google (200 M, 8 M) *	99.63	95.12	Schroff et al. [52]
VGGFace	2015	VGGNet-16	1	Triplet Loss	VGGFace (2.6 M, 2.6 K)	98.95	97,3	Parkhi et al. [53]
Augmentation	2016	VGGNet-19	1	Softmax	CASIA WebFace (494 k, 10 k)	98.06	-	Masi et al. [54]
Range Loss	2016	VGGNet-16	1	Range Loss	CASIA WebFace + MS-Celeb-1M (5 M, 100 k)	99.52	93.70	Zhang et al. [55]
Center Loss	2016	LeNet±7	1	Center Loss	CASIA WebFace + CACD2000 + Celebrity + (0.7 M, 17 k)	99.28	94.9	Wen et al. [56]
L-Softmax	2016	VGGNet-18	1	L-Softmax	CASIA-WebFace (490 k, 10 K)	98.71	-	Liu et al. [57]
Center Invariant Loss	2017	LeNet	1	Center Invariant Loss	CASIA WebFace (0.45 M, 10 k)	99.12	93.88	Wu et al.[58]
L2-Softmax	2017	ResNet-101	1	L2-Softmax	MS-Celeb 1M (3.7 M, 58 k)	99.78	96.08	Ranjan et al. [59]
Marginal Loss	2017	ResNet-27	1	Marginal Loss	MS-Celeb 1M (4 M, 82 k)	99.48	95.98	Deng et al. [60]
NormFace	2017	ResNet-28	1	Contrastive Loss	CASIA WebFace (494 k, 10 k)	99.19	-	Wang et al. [61]
COCO Loss	2017	ResNet-128	1	COCO Loss	MS-Celeb 1M (3 M, 80 k)	99.86	-	Liu et al. [62]
Von Mises-Fisher	2017	ResNet-27	1	vMF Loss	MS-Celeb-1M (4.61 M, 61.24 K)	99.63	96.46	Hasnat et al. [63]
SphereFace	2018	ResNet-64	1	A-Softmax	CASIA WebFace (494 k, 10 k)	99.42	95.0	Liu et al. [64]
Ring Loss	2018	ResNet-64	1	Ring Loss	MS-Celeb-1M (3.5 M, 31 K)	99.50	-	Zheng et al. [65]

MLR	2018	ResNet-34	1	CCS Loss	MS-Celeb-1M (10 M, 100 K)	99.71	-	Guo and Zhang [66]
Cosface	2018	ResNet-64	1	Large Margin Cosine Loss	CASIA WebFace (494 k, 10 k)	99.73	97.6	Wang et al. [68]
AM-Softmax	2018	ResNet-20	1	AM-Softmax Loss	CASIA WebFace (494 k, 10 k)	99.12	-	Wang et al. [69]
Light-CNN	2018	Light CNN	1	Softmax	MS-Celeb-1M (5 M, 79 K)	99.33	-	Wu et al. [70]
ArcFace	2019 2022Rev.	ResNet-100	1	ArcFace	MS-Celeb-1M (5.8 M, 85 k)	99.83	98.01	Deng et al. [86]
Affinity Loss	2019	ResNet-50	1	Affinity Loss	VGGFace2 (3.31 M, 8 K)	99.65	97.3	Hayat et al. [71]
CLMLE	2019	Resnet-64	1	CLMLE Loss	CASIA WebFace (494 k, 10 k)	99.62	96.5	Huang et al. [**72**]
PDSN	2019	ResNet-50	1	Pairwise Contrastive Loss	CASIA WebFace (494 k, 10 k)	99.20	-	Song et al. [73]
Feature Transfer	2019	Letnet++	1	Softmax	MS-Celeb-1M (4.8 M, 76.5 K)	99.55	-	Yin et al. [75]
ACNN	2019	ResNet-100	1	ArcFace Loss	DeepGlint-MS1M (3.9 M, 86 K)	99.83	-	Ling et al.[78]
Agressive Data Augmentation	2020	GoogLeNet	1	Softmax with center loss	CASIA WebFace (494 k, 10 k)	99.02	96.6	Ben Fredj et al. [76]
MML	2020	Inception ResNet-V1	1	Minimum Margin Loss Loss	VGGFace2 (3.05 M, 8 K)	99.63	-	Wei et al. [67]
IAM	2020	Inception ResNet-V1	1	IAM Loss	CASIA WebFace (494 k, 10 k)	99.12	95.28	Sun et al. [77]
RCM loss	2020	ResNet-18	1	Rotation Consistent Margin Loss	CASIA WebFace (494 k, 10 k)	98.91	94.95	Wu et al. [79]
LMC SDLMC DLMC	2020	Resnet-32	1	LMC loss SDLMC loss DLMC loss	CASIA WebFace (494 k, 10 k)	99.13 99.03 99.07	- - 94.36	Wu and Wu [80]
ACLL	2020	ResNet-100	1	ML MV-Arc-Softmax	CASIA WebFace (494 k, 10 k)	99.80	-	Huang et al. [81]
Group-aware	2020	ResNet-100	1	ArcFace	MSCeleb-1M (3.8M, 85 K)	99.85	97.8	Kim et al. [82]
Dyn-arcFace	2021	SE-LResnetE-IR	1	Dyn-arcFace	CASIA WebFace (494 k, 10 k	99.80	-	Jiao et al. [83]
ElasticFace	2022	ResNet-50	1	ElasticFace-Arc ElasticFace-Cos ElasticFace-Arc+ ElasticFace-Cos+	MS1MV2 (MS-Celeb-1M (5.8 M, 85 k))	99.80 99.82 99.82 99.80	-	Boutros et al. [84]
AO Loss	2022	MTCNN	1	Additive Orthant Loss	CASIA WebFace (494 k, 10 k	99.45	-	Hun et al.[85]
Interface	2022	ResNet - 100	1	Interface	MS1MV2 (MS-Celeb-1M (5.8 M, 85 k))	99.83	-	Sang et al. [87]

4 - Prueba de concepto

4.1 - Introducción

Esta prueba de concepto consiste en realizar un Reconocimiento Facial sobre las imágenes captadas por una webcam en vivo, una muestra del funcionamiento de las redes neuronales convolucionales. Aunque la biometría facial es más amplia que esta prueba, el Reconocimiento Facial reúne gran parte de los conceptos que convergen en la biometría facial.

La prueba seguirá la canalización estándar del Reconocimiento Facial descrito en el artículo, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification" de Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato y Lior Wolf de Y. Taigman, M. Yang, M. Ranzato and L. Wolf [45]

Detección > Alineación > Normalización > Representación > Verificación [45]

En ocasiones, parte de estas canalizaciones durante la implementación se solapan. Para que la prueba sea aún más clara compactaré los pasos en el siguiente esquema.

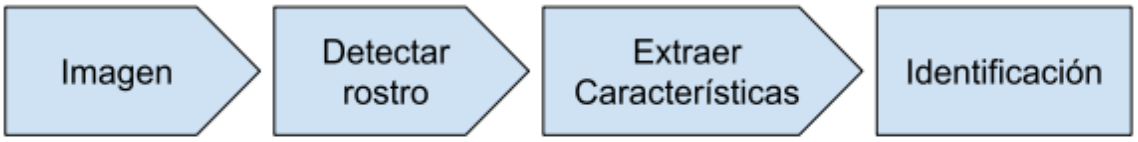


Figura 68. Etapas del Reconocimiento Facial. Fuente: Propia

La prueba se desarrolla sobre un jupyter notebook, que nos permite ir viendo celda a celda cada etapa. En este proceso de reconocimiento, la red neuronal sólo actúa extrayendo las características de nuevas imágenes que le son presentadas (matriculación de perfiles): el vector de representación que se extrae de estas imágenes es el que sirve para comparar y proceder con la identificación de las personas que sean expuesta a la webcam.

El proceso de matriculación de perfiles se realizará colocando en una carpeta denominada "rostrosMatriculados" imágenes de las personas que se quiere identificar.

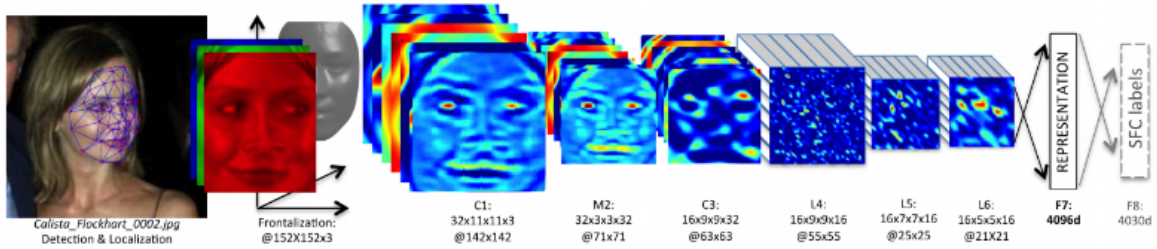
El esquema anterior quedaría por tanto:

- Imagen: Captura Webcam
- Detectar Rostro: Utilización de un detector de rostros de openCV (Haar)
- Extraer Características: extracción del vector de representación de rostros matriculados con el uso de ConvNet DeepFace previamente entrenada
- Identificación: Comparación del vector del rostro que detecta la webcam con el vector de las imágenes matriculadas

El desarrollo de la prueba se realizará en un entorno local Linux con Python como lenguaje de programación. Actualmente existen diferentes alternativas para construir modelos de aprendizaje

profundo con Python, que van desde implementar a bajo nivel una red de neuronas artificiales hasta utilizar librerías que ya las implementan, como Tensorflow, facilitando la creación y configuración de estas redes neuronales.

La red neuronal utilizada en esta prueba de concepto es DeepFace [45], una ConvNet de 9 capas, desarrollada por Taigman et al [45] en 2014 sobre el dataset Social Face Classification - SFC [45]. Casi cualquier red actual tiene muchas más capas, pero he elegido esta por la claridad con la que se puede mostrar el funcionamiento de una ConvNet en el Reconocimiento Facial. Incluso existen diversos frameworks para el Reconocimiento Facial que presentan una API que facilitan enormemente su implementación, pero no permiten ver su funcionamiento interno.
La ConvNet DeepFace utilizada en esta prueba de concepto es una adaptación a Keras del mismo modelo que el original, realizada por Swarup Ghosh[119-122] sobre el dataset VGGFace2 [101,121]




```
#DeepFace
base_model = Sequential()
base_model.add(Convolution2D(32, (11, 11), activation='relu', name='C1', input_shape=(152, 152, 3)))
base_model.add(MaxPooling2D(pool_size=3, strides=2, padding='same', name='M2'))
base_model.add(Convolution2D(16, (9, 9), activation='relu', name='C3'))
base_model.add(LocallyConnected2D(16, (9, 9), activation='relu', name='L4'))
base_model.add(LocallyConnected2D(16, (7, 7), strides=2, activation='relu', name='L5') )
base_model.add(LocallyConnected2D(16, (5, 5), activation='relu', name='L6'))
base_model.add(Flatten(name='F0'))
base_model.add(Dense(4096, activation='relu', name='F7'))
base_model.add(Dropout(rate=0.5, name='D0'))
base_model.add(Dense(8631, activation='softmax', name='F8'))
base_model.summary()
```

Figura 69: Arriba, DeepFace [45]. Abajo, Keras de DeepFace [123]

La figura muestra con una claridad pedagógica la correspondencia de cada capa del modelo DeepFace con cada línea de código.

4.2 - Entorno de desarrollo y herramientas

El entorno de desarrollo se ha realizado de forma local sobre el hardware disponible, con sistema operativo linux y descargando las herramientas de software necesarias.

- Hardware
  - Lenovo ThinkStation S20
    - Procesador Xeon W3690 de 6 núcleos
    - RAM 16GB DDR3
    - GPU Nvidia RTX2060 6GB

```
(base) manpla@HALTS:~$ nvidia-smi
Sat Dec  3 11:38:14 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.85.02    Driver Version: 510.85.02    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA GeForce ...  Off  | 00000000:02:00.0  On |                  N/A |
| 24%   29C    P8    16W / 160W |   4874MiB /  6144MiB |      2%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
```

Figura 70: Captura pantalla Driver Nvidia RTX2060. Fuente: Propia

- WebCam con conexión USB:
  - Webcam USB Sony PlayStation 3

Figura 71: Webcam PS3 utilizada. Fuente: Propia

Software: Entorno y Herramientas

- **S.O. BackBox 8**: Distribución de linux basada en Ubuntu
  - `(base) manpla@HALTS:~$ lsb_release -a`
    `Distributor ID: Ubuntu`
    `Description: BackBox Linux 8 amd64 (Cubic 2022-11-15 20:46)`
    `Release:22.04`
    `Codename:jammy`

- **Python**: Lenguaje de programación interpretado multiplataforma de código abierto. Fue creado a finales de los años ochenta por Guido van Rossum en el Centro para las Matemáticas y la Informática de Países Bajos. Python ofrece una gran cantidad de librerías de código abierto y frameworks que facilitan el desarrollo de aplicaciones. Es uno de los lenguajes más utilizados en deep learning por ser muy versátil y contar con un gran número de librerías especializadas: TensorFlow, Keras, NumPy, entre otras.
  - `(base) manpla@HALTS:~$ python -V`
    `Python 3.9.12`

- **Conda**: Gestor de paquetes y entorno para Python
  - `(base) manpla@HALTS:~$ conda -V`
    `conda 22.9.0`

- **Pip**: Gestor de paquetes y entorno para Python.
  - `(base) manpla@HALTS:~$ pip -V`
    `pip 21.2.4 from /home/manpla/miniconda3/lib/python3.9/site-packages/pip (python 3.9)`

- **Jupyter Notebook**: Jupyter Notebook es un entorno interactivo basado en la web para crear documentos, que alterna código ejecutable con texto plano y admite el formateo de texto mediante lenguaje de marcado HTML.
    - `(base) manpla@HALTS:~$ jupyter notebook -V`
      `6.5.2`

- **Numpy**: Es una librería para crear vectores y matrices multidimensionales. Cuenta además con una colección de funciones matemáticas de alto nivel.

- **Matplotlib**: Es una librería para generación de gráficos. La API está diseñada para recordar a la de MATLAB. Su creador fue John D. Hunter en 2003.

- **Os**: Esta librería permite manipular rutas y crear archivos o directorios del S.O. base. Dispone de múltiples funciones para interactuar con el sistema operativo.

- **Pillow:** es una librería que proporciona funciones para el tratamiento de imágenes en distintos formatos.

- **OpenCV**: Es una librería de visión artificial originalmente desarrollada por Intel. Está desarrollada en C++ pero incluye conectores para otros lenguajes, como Python, Javascript, Octave, entre otros muchos.

- **TensorFlow**: Es una librería desarrollada por Google para el aprendizaje automático. Permite construir y entrenar redes neuronales. TensorFlow fue originalmente desarrollado por el equipo de Google Brain para uso interno antes de ser publicado bajo la licencia de código abierto Apache 2.0 el 9 de noviembre de 2015. TensorFlow es compatible de forma nativa con múltiples lenguajes de programación como C++, Haskell, Java, Go, Rust y Python. Tiene una arquitectura flexible, versátil y permite un desarrollo rápido de computación para CPU/GPU/TPU con el mismo código, sin tener que especificar. TensorFlow determinará qué partes del cálculo se deben mover a la GPU/TPU. Esto optimiza el proceso de aprendizaje de las redes neuronales.

- **Keras**: Keras es una API de alto nivel escrito en Python desarrollado por François Chollet[4] para facilitar la construcción de modelos de deep learning sobre TensorFlow y otras librerías Theano o CNTK. Keras simplifica la generación de modelos mediante abstracciones intuitivas, modularidad y extensibilidad. Desde 2017, TensorFlow incluye a Keras en el núcleo de la librería de TensorFlow 2.0. Aunque Keras presenta bastantes ventajas para el uso de TensorFlow, existen ciertos casos en los que Keras no es recomendable. Con TensorFlow es posible entrenar modelos de estructura compleja que no pueden ser implementados con Keras, así como en aquellos casos en los que sea necesario aprovechar la máxima capacidad de cómputo del equipo.

```
Terminal - manpla@HALTS: ~
File  Edit  View  Terminal  Tabs  Help
(base) manpla@HALTS:~$ python
Python 3.9.12 (main, Apr  5 2022, 06:56:58)
[GCC 7.5.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow
2022-12-05 11:12:01.553598: I tensorflow/stream_executor/platform/default
/dso_loader.cc:49] Successfully opened dynamic library libcudart.so.10.1
>>> tensorflow.__version__
'2.4.1'
>>> from tensorflow import keras
>>> keras.__version__
'2.4.0'
>>>
```

Figura 72: Captura pantalla. Versiones de tensorflow y keras. Fuente: [Propia]

4.3 - Prueba de Concepto.

4.3.1 - Reconocimiento facial Webcam con ConvNet

Para aislar el trabajo a realizar de esta prueba se ha creado un entorno (environment) denominado UOC donde cargar todas las librerías necesarias.

```
(base) manpla@HALTS:~$ conda create -n UOC
Collecting package metadata (current_repodata.json): done
Solving environment: done
## Package Plan ##
environment location: /home/manpla/miniconda3/envs/UOC
```

```
Terminal - manpla@HALTS: ~
File  Edit  View  Terminal  Tabs  Help
(base) manpla@HALTS:~$ conda activate UOC
(UOC) manpla@HALTS:~$ jupyter notebook
```

Figura 73: Captura de pantalla. Entorno env UOC. Fuente: [Propia]

Mediante los gestores conda y/o pip se completa el entorno recién creado con las librerías necesarias descritas anteriormente para que pueda correr la red neuronal.
Debido a que el procesador Xeon W3690 es de 2011 no cuenta con las instrucciones AVX que son usadas en las versiones más recientes de Tensorflow. La versión de Tensorflow utilizada es 2.4.1. La versión de Keras que tiene integrada es 2.4.0
El entorno cuenta con el apoyo de una gráfica Nvidia RTX2060 de 6GB con soporte CUDA que agiliza el proceso.

4.3.2 - VGGFace2 Dataset

El conjunto de datos utilizado en esta prueba es VGGFace2 [101,122]. Se trata de un dataset con múltiples imágenes por individuo, con distintas poses, edades, iluminación, entre otras. Aproximadamente 3.3 millones de rostros de unas 9000 identidades distintas. Esto hace una media aproximada de 362 muestras por cada individuo

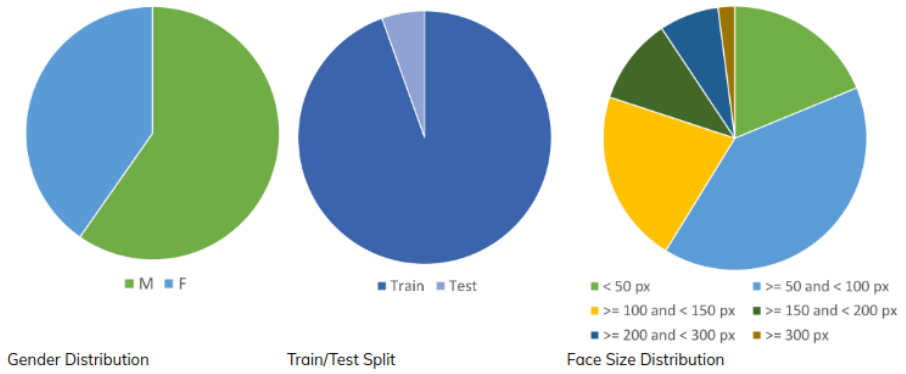


Figura 74: Distribuciones del dataset VGGFace2. Fuente: [101]

VGGFace y VGGFace2 son una serie de modelos desarrollados por el equipo de Visual Geometry Group (VGG) de la Universidad de Oxford para el Reconocimiento Facial y conjuntos de datos de visión por computadora.

4.3.3 - DeepFace

DeepFace [45, 121] es una red neuronal convolucional de 9 capas.

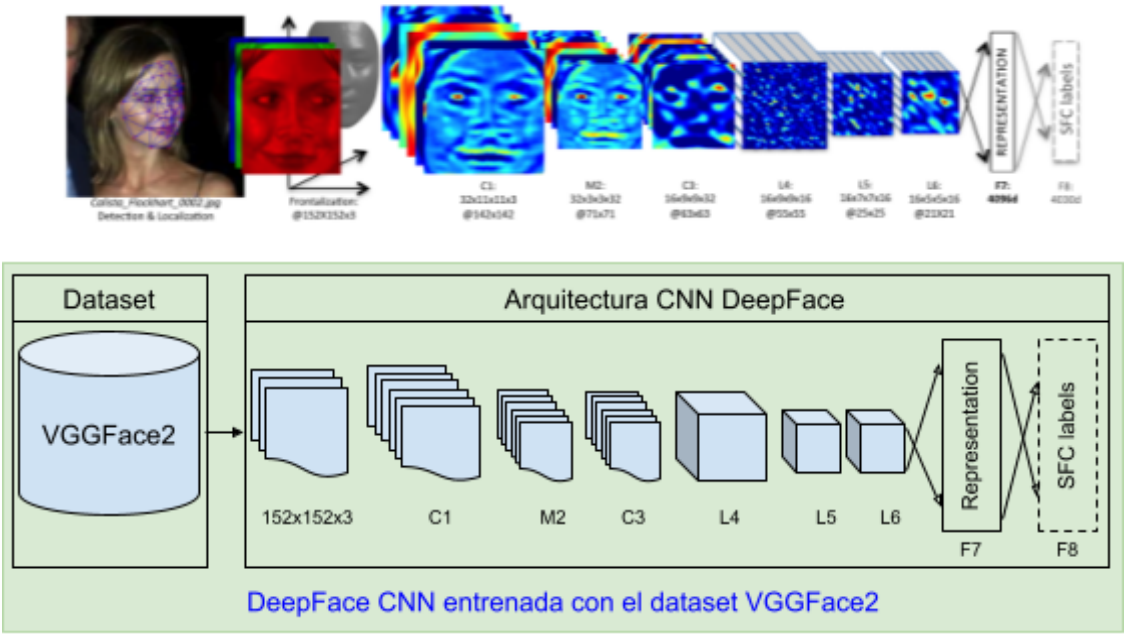


Figura 75: Modelo DeepFace de 9 capas CNN-9. Fuentes: Arriba [45] Abajo [Propia]

Breve resumen de la arquitectura que se muestra en la figura:

- La red se alimenta con una imagen facial alineada de 3 canales RGB de tamaño 152 por 152 píxeles,
- esta imagen alineada alimenta a una capa convolucional (C1) con 32 filtros de tamaño 11x11x3 (denotamos esto por 32x11x11x3@152x152).

- Los 32 mapas de entidades resultantes alimentan una capa de agrupación Max-Pooling (M2) que extrae el máximo sobre 3x3 vecindarios espaciales con una zancada de 2, para cada uno de los 32 canales.
- A esto le sigue otra capa convolucional (C3) que tiene 16 filtros de tamaño 9x9x16.
  El propósito de estas tres capas es extraer características de bajo nivel, como bordes simples y textura. Las capas de agrupación max-pool M2 hacen que la salida de las redes de convolución sea más robusta para las regiones locales. El hecho que se alineen las imágenes previas a la entrada facilitan la extracción de características faciales. El propósito de estas tres capas es extraer características de bajo nivel, como bordes simples y textura.

Antes de seguir con la funcionalidad del resto de las capas, hay que indicar que alinear los rostros previamente a ser procesados, mejora notablemente la extracción de características, porque sitúa los mismos elementos (ojos, nariz, boca, entre otras.) en posiciones similares para todas las imágenes del dataset. Esto ocurre para todas las redes convolucionales, sea cual sea su arquitectura. Como ejemplo, vamos a alinear dos imágenes cualquiera y superponerlas con una semitransparencia para mostrar, cómo una vez alineadas, los rasgos faciales quedan prácticamente en las mismas posiciones.

Figura 76. De izquierda a derecha, Maria Valverde y Vanesa Romero. Fuente: [118]

Las distancias entre los ojos y las distancias “T” que forman ojos-nariz (nariz-boca) son características comunes a casi todos los rostros. Sin embargo, aun siendo imágenes frontales, el hecho de que estén ligeramente inclinadas hace que la extracción de estas características puedan diferir enormemente.

Figura 77. Rotación para alinear los rostros. Fuente: [118]

La alineación de los rostros mejora la extracción de estas características y son más robustas.
Para mostrar visualmente como mejora la comparación superponemos ambas imágenes con transparencia parcial para visualizar las posiciones más relevantes: ojos, nariz, boca.

Figura 78: Imágenes superpuestas. Izquierda sin alinear. Derecha alineadas. Fuente: Propia

En general, las capas de agrupación max-pooling hacen que la salida de las ConvNet sea más sólida para las transferencias de información de regiones locales.
Las regiones locales son espacios de la imagen de entrada con información relevante, como pueda ser, por ejemplo, la distancia entre los ojos. Al tratarse de un recuadro fijo (152x152) la entrada, la región que ocupa la posición de los ojos es muy parecida en todas las imágenes de entrada (si están alineadas previamente). Por tanto, estas regiones locales se pueden conectar a capas locales que extraen más información de estas zonas concretas.
Cuando estas capas iniciales se aplican a imágenes faciales alineadas, hacen que la red sea más robusta frente a pequeños errores de registro. Sin embargo, varios niveles de agrupación harían que la red perdiera información sobre la posición precisa de la estructura facial detallada y las microtexturas. Esta es la razón de que solo se aplique la capa max-pooling a la primera capa convolucional. Se podría decir que estas primeras capas (C1-M2-C3) actúan como una etapa de preprocesamiento adaptativo de front-end. Estas capas simplemente amplían la entrada en un conjunto de simples características locales (región local con características específicas del recuadro 152x152 inicial).

- Las capas posteriores (L4, L5 y L6) están conectadas localmente (a cada región local de características que ha localizado). Dado que las diferentes regiones de una imagen alineada tienen diferentes estadísticas locales, actúan como un banco de filtros en cada ubicación en el mapa de entidades. Por ejemplo, las áreas entre los ojos y las cejas exhiben una apariencia muy diferente y tienen una capacidad de discriminación mucho mayor en comparación con las áreas entre la nariz y la boca, u otras. Se utiliza esta localización de regiones para realizar las conexiones de las capas locales, aprovechando el hecho de que las imágenes de entrada están alineadas.

- Para terminar, las dos capas finales (F7 y F8) son fully_layer, están completamente conectadas: cada unidad de salida está conectada a todas las entradas. Estas capas capturan las correlaciones entre las características capturadas en partes distantes de las imágenes de la cara, como la posición-la forma de los ojos y la posición-la forma de la boca.
  La salida de la primera capa completamente conectada (F7) extrae el vector de características de representación de cara sin procesar.
  F8 , la capa de pérdida Softmax, se representa por puntos discontinuos porque contiene la identificación de los rostros del dataset, VGGFace2, por tanto hay que descartarla para usar el entrenamiento de la red en otros reconocimientos.

Hay otros detalles de esta red, como la frontalización inicial, por ejemplo, que no aportan más información para la explicación de esta prueba y entrar en estos detalles alargarían la explicación.

4.4 - Jupyter Notebook. RF Webcam

4.4.1 - Importación Librerías

Inicialmente se importan las librerías que se van a utilizar, que fundamentalmente es la API Keras para modelar la ConvNet y OpenCV para la manipulación de imágenes en vivo.

4.4.2 - Filtro de detección de rostros

Como filtro de detección de rostros se utiliza el filtro HAAR que provee la librería OpenCV

```
#Deteccion de rostros con OpenCV haarcascade
face_cascade = cv2.CascadeClassifier('/home/manpla/miniconda3/envs/UOC/lib/python3.10/'
                                     'site-packages/cv2/data/haarcascade_frontalface_default.xml')
```

Figura 79. Captura Jupyter Notebook. Fuente: [Propia]

4.4.3 - DeepFace con Keras

La arquitectura de DeepFace se realiza con Keras

```
#DeepFace
base_model = Sequential()
base_model.add(Convolution2D(32, (11, 11), activation='relu', name='C1', input_shape=(152, 152, 3)))
base_model.add(MaxPooling2D(pool_size=3, strides=2, padding='same', name='M2'))
base_model.add(Convolution2D(16, (9, 9), activation='relu', name='C3'))
base_model.add(LocallyConnected2D(16, (9, 9), activation='relu', name='L4'))
base_model.add(LocallyConnected2D(16, (7, 7), strides=2, activation='relu', name='L5') )
base_model.add(LocallyConnected2D(16, (5, 5), activation='relu', name='L6'))
base_model.add(Flatten(name='F0'))
base_model.add(Dense(4096, activation='relu', name='F7'))
base_model.add(Dropout(rate=0.5, name='D0'))
base_model.add(Dense(8631, activation='softmax', name='F8'))
base_model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 C1 (Conv2D)                 (None, 142, 142, 32)      11648

 M2 (MaxPooling2D)           (None, 71, 71, 32)        0

 C3 (Conv2D)                 (None, 63, 63, 16)        41488

 L4 (LocallyConnected2D)     (None, 55, 55, 16)        62774800

 L5 (LocallyConnected2D)     (None, 25, 25, 16)        7850000

 L6 (LocallyConnected2D)     (None, 21, 21, 16)        2829456

 F0 (Flatten)                (None, 7056)              0

 F7 (Dense)                  (None, 4096)              28905472

 D0 (Dropout)                (None, 4096)              0

 F8 (Dense)                  (None, 8631)              35361207

=================================================================
Total params: 137,774,071
Trainable params: 137,774,071
Non-trainable params: 0
```

Figura 80. Captura de pantalla. DeepFace en Keras. Fuente: [Propia]

El entrenamiento de una red neuronal es un proceso complejo, arduo y que lleva un tiempo de pruebas y afinamiento.

Pasar el dataset VGGFace2 con millones de imágenes por el entorno local descrito para realizar esta prueba supondría semanas de entrenamiento. Por esto se opta por cargar los pesos ya entrenados **"VGGFace2_DeepFace_weights_val-0.9034.h5"** [122]

Dataset	# Identities	# Images
VGGFace2-Train	8631	3.3M
VGGFace2-Train*	8631	2.3M

*The VGGFace2 dataset (publicly available training set) was used for training such that 3 images from each of the classes were left for the hold-out dev-validation set. The aforementioned alignment technique accounted for removal of a lot of images contained in the original VGGFace2 dataset as HOG face detector could not detect faces that lacked frontal pose.

Figura 81. Pesos para DeepFace. Dataset VGGFace2. Fuente: [119]

4.4.4 - Carga de pesos

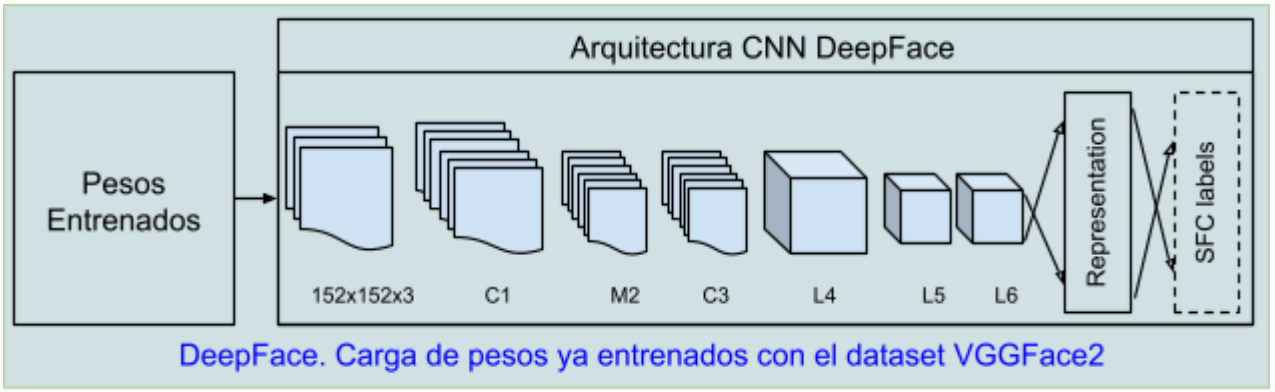


Figura 82. Esquema de carga de pesos a DeepFace. Fuente: [Propia]

Una vez definido el modelo Keras se cargan los pesos de la red entrenada. Denótese que la carga se realiza sobre la red completa, con todas sus capas.

Se cargan los pesos de la red entrenada y las últimas capas de la red convolucional son eliminadas (drop). La función de Keras model.summary() muestra esquemáticamente la nueva estructura de la red. Si no se eliminase la última capa sólo se detectarían los modelos entrenados del dataset VGGFace2

```
# Carga de los pesos entrenados https://github.com/swghosh/DeepFace/releases/tag/weights-vggface2-2d-aligned
base_model.load_weights("/home/manpla/UOCFace/VGGFace2_DeepFace_weights_val-0.9034.h5")

# Eliminacion de las ultimas capas F8 y D0. Dejamos como salida la capa de representación F7
model = Model(inputs=base_model.layers[0].input, outputs=base_model.layers[-3].output)
model.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 C1_input (InputLayer)       [(None, 152, 152, 3)]     0

 C1 (Conv2D)                 (None, 142, 142, 32)      11648

 M2 (MaxPooling2D)           (None, 71, 71, 32)        0

 C3 (Conv2D)                 (None, 63, 63, 16)        41488

 L4 (LocallyConnected2D)     (None, 55, 55, 16)        62774800

 L5 (LocallyConnected2D)     (None, 25, 25, 16)        7850000

 L6 (LocallyConnected2D)     (None, 21, 21, 16)        2829456

 F0 (Flatten)                (None, 7056)              0

 F7 (Dense)                  (None, 4096)              28905472

=================================================================
Total params: 102,412,864
Trainable params: 102,412,864
Non-trainable params: 0
_________________________________________________________________
```

Figura 83. Captura Notebook de carga de pesos a DeepFace. Fuente: [Propia]

4.4.5 - Matriculación de Rostros

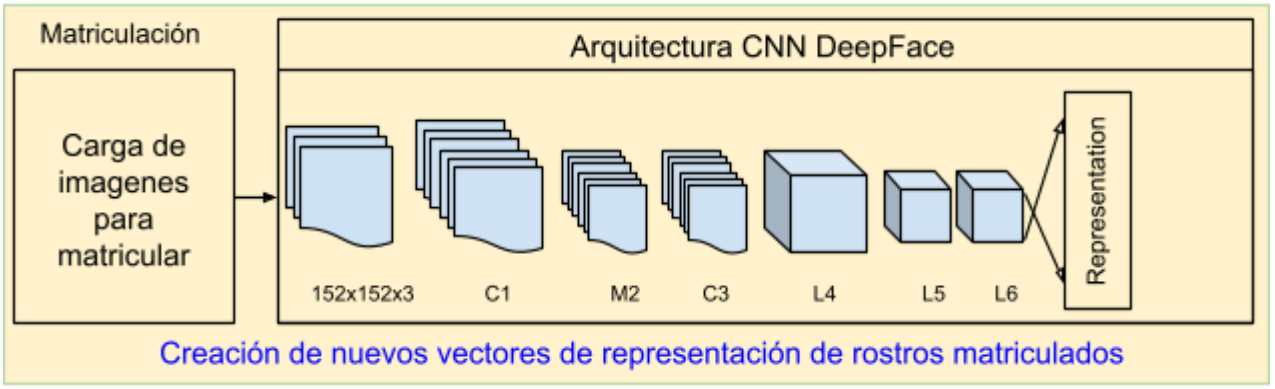


Figura 84. Esquema de carga de imágenes para matricular a DeepFace. Fuente: [Propia]

Una vez eliminada la última capa F8 la red está lista para aceptar (matricular) nuevos rostros. Para no complicar el código, esta carga se realizará colocando las imágenes que se quiere matricular en una carpeta específica “rostrosMatriculados” para la posterior identificación

```
MANUEL
2022-12-06 10:47:45.904470: I tensorflow/stream_executor/cuda/cuda_dnn.cc:384] Loaded cuDNN version 8401
1/1 [==============================] - 57s 57s/step
[0.       0.       0.       ... 0.       5.275256 3.751422]
EMMA
1/1 [==============================] - 0s 119ms/step
[0.        0.        0.        ... 0.        5.521675  1.5546776]
MICHAEL
1/1 [==============================] - 0s 108ms/step
[0.        0.        0.        ... 0.        7.4774456 3.9740827]
```

Figura 85. Captura Notebook del proceso de matriculación con DeepFace. Fuente: [Propia]

4.4.6 - Reconocimiento WEBCAM

Mediante una WebCam (PS3) se captan, en tiempo real, los rostros que sean expuestos a ella.

Se usa la librería OpenCV para capturar la imagen en vivo. Mediante el filtro HAAR se detectan los rostros que aparecen en la imagen y se marcan mediante un recuadro de detección.
El reconocimiento de la imagen captada se consigue al realizar una comparación vectorial, mediante la distancia euclídea, entre el rostro detectado con la imagen registrada previamente (matriculada). Cuanto menor sea la distancia mayor posibilidad existe de que se trate de la misma persona. Denótese que en imágenes fijas, si se comparan dos imágenes exactamente iguales, la distancia euclídea sería cero al tener exactamente el mismo vector de representación.
Se muestra el reconocimiento de tras alcanzar el umbral de reconocimiento, mostrando la imagen que se registró para la matriculación así como el nombre del rostro identificado.
La aplicación sigue el esquema que se expone a continuación

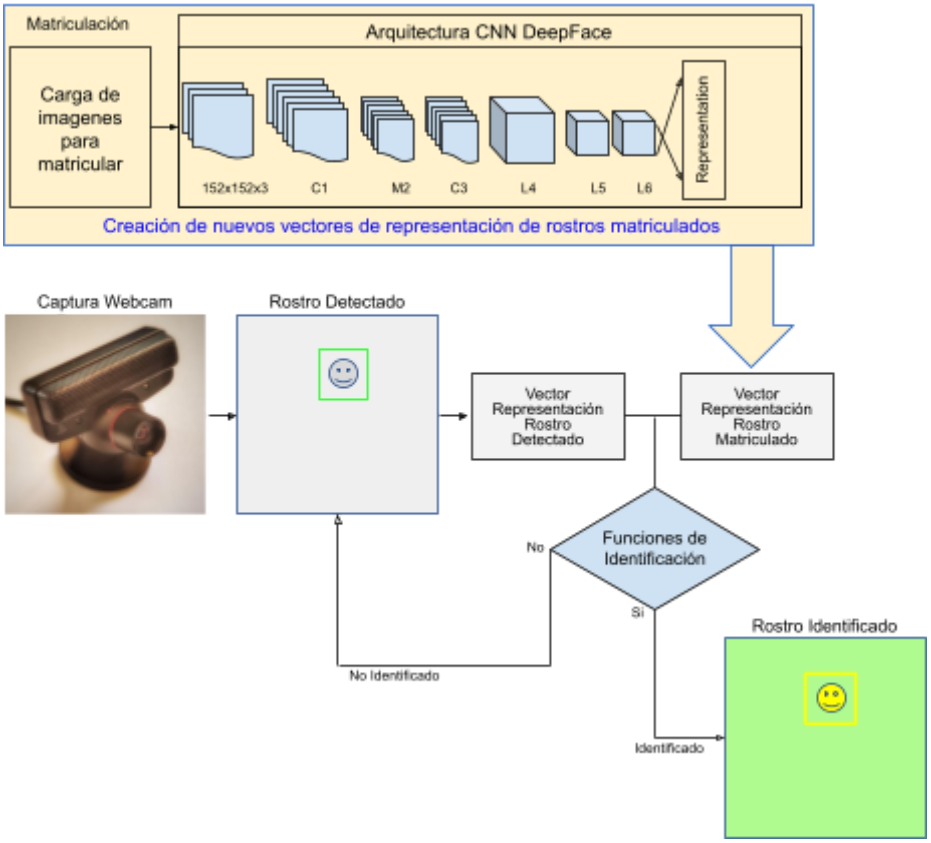


Figura 86. Diagrama del funcionamiento del reconocimiento facial con DeepFace. Fuente: [Propia]

4.4.7 - Prueba de reconocimiento facial en tiempo real

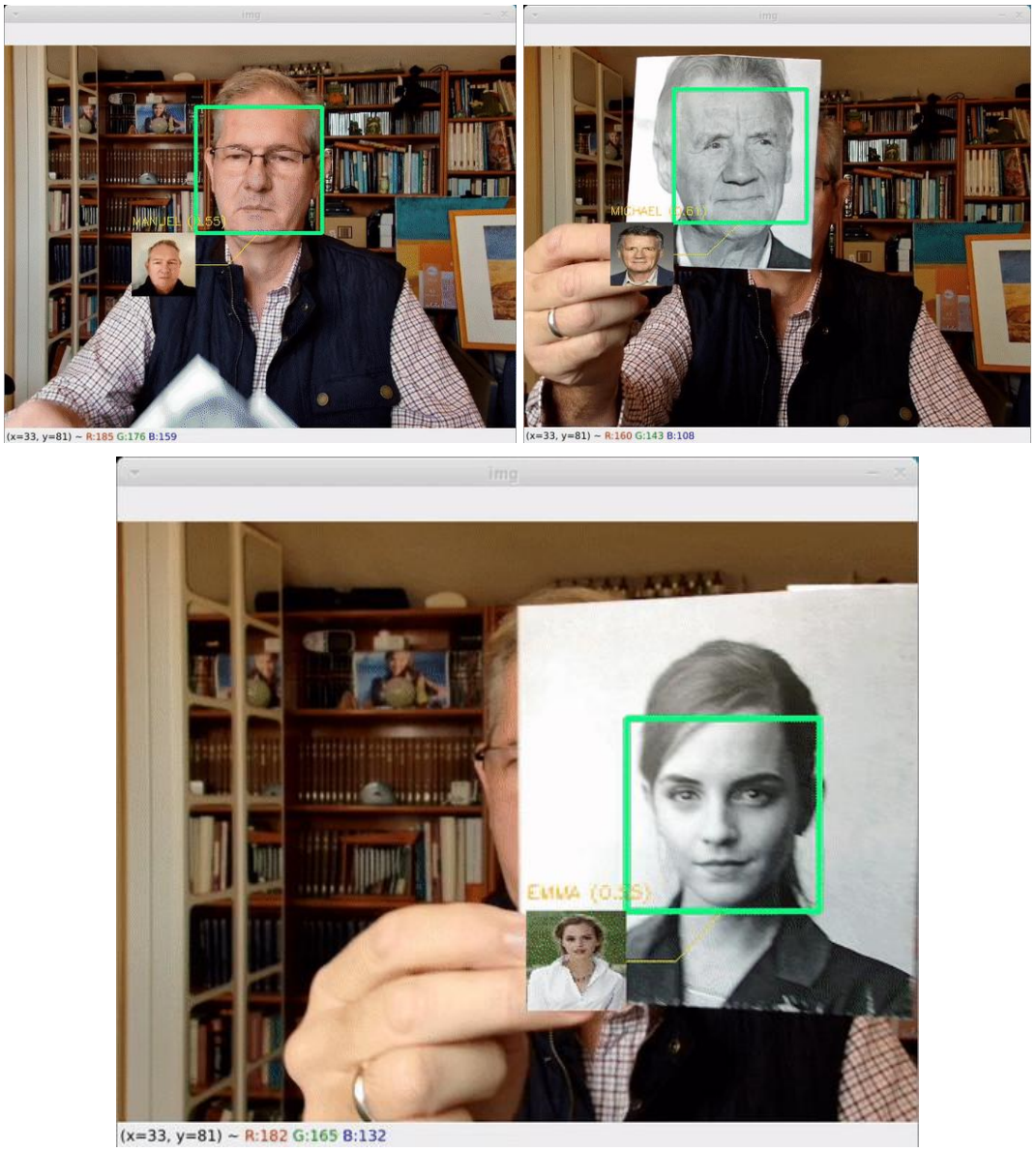


Figura 87: Capturas WEB. Manuel, Michael Palin y Emma Watson. Fuente: [Propia]

Los enlaces a publicaciones, herramientas y repositorios utilizados para la elaboración de esta POC se encuentran en la Bibliografía al final de este documento.

5 - Conclusiones

Con un rendimiento en el dataset de referencia LFW por encima del 99% desde el 2014 prácticamente en todos los estudios presentados para el reconocimiento facial con ConvNet, con alguna excepción que no baja del 97%, se podría decir que nos encontramos ante una técnica muy madura, con resultados excelentes que rozan el 100% de efectividad. Sin embargo, no son pocos los retos a los que aún se enfrenta el Reconocimiento Facial con Deep Learning. Los datasets públicos aún son reducidos, poco miles de identidades, además de tener fuertes desviaciones en el equilibrio de muestras. Constantemente se realizan aportaciones para solventar esta situación, por ejemplo en 2022 WebFace260M [116] 260 millones de rostros con 4 millones de identidades (260M, 4M). Casi ningún conjunto de datos actual tiene una representación equilibrada de las distintas razas humanas, así como un equilibrio en los rangos etarios de las identidades. El equilibrio en número de imagen por identidad también es muy dispar y constantemente se recurren a técnicas para equilibrar este desajuste, como el aumento de datos. En los últimos años se están usando las redes antagónicas generativas GAN [112] para incrementar los datos faltantes. Como indicaba Cao et al. [101] en 2014 es necesario incrementar sustantivamente la cantidad de individuos y las muestras por cada individuo, no solo cuantitativamente sino también cualitativamente, para obtener mejores resultados en el aprendizaje profundo, en definitiva la cantidad y calidad del conjunto de entrenamiento influye notablemente en en los resultados.

En los primeros años tras Alexnet hubo una tendencia al incremento en el número de redes y número de capas en las ConvNet [45-51]. Sin embargo, tras la presentación de GoogLeNet [52], el número de redes se redujo a una y las capas oscilaban entre 9 a 100 de media. Es evidente que aunque hoy en día la potencia computacional es mayor que en 2012, el tener muchas redes y muchas capas supone un gasto computacional innecesario a la vista de resultados con redes mucho más livianas, como Deng et al. con una ResNet-100 obtuvo 99.83% de precisión, o Hasnat et al. [63] con una ResNet-27 un 99.63%, frente a Sun et al. [46] con la serie DeepId donde el número de redes oscilaban entre 25 a 60 redes CNN-9 con resultados del 97.45% al 99.53%

Del estudio del Estado del Arte se deduce claramente la importancia de la Función de Pérdida para la correcta extracción, separabilidad y distriminacion de las características faciales. Fundamentalmente se tratan de variaciones de Softmax donde se busca las diferencias de distancias, bien euclidiana o del coseno. La normalización también ha jugado un papel importante en el tratamiento de las características faciales agilizando más la estructura de la red.

En 2020 tuvimos el infortunio de vivir una pandemia global. Esto hizo más patente el problema de la oclusión facial en el reconocimiento facial, al tapar por imperativo sanitario nariz y boca. Los sistemas ya en funcionamiento se encontraron con esta dificultad adicional. Esto nos lleva a otro de los desafíos aún por resolver en la biometría facial, las variaciones dentro de una misma identidad ya sea por iluminación, variaciones de pose, oclusiones, escalado, entre otras. Actualmente hay muchos estudios para la obtención del reconocimiento facial con solo parte de la detección facial [104].

La prueba POC ha consistido en mostrar como una red convolucional extrae el vector de características de unas imágenes que se ha dado para la matriculación y posterior identificación con una WEBCam. La red elegida ha sido DeepFace de Taigman et al. [45] en una adaptación con Keras [119-123] sobre un dataset VGGNet2. La carga de pesos ya entrenados se ha descargado de

un repositorio público [122] y cargados sobre la ConvNet. La carga de la imágenes matriculadas son distintas a la mostradas en la webcam. La red identifica que las características faciales mostradas a través de la webcam son las mismas que las matriculadas, dando un recuadro con la identificación. En la propia imagen webcam aparecen la imagen matriculada en un pequeño recuadro y el rostro que identifica. Denótese que la oclusión parcial de mi rostro provoca que no se realice la detección/reconocimiento.

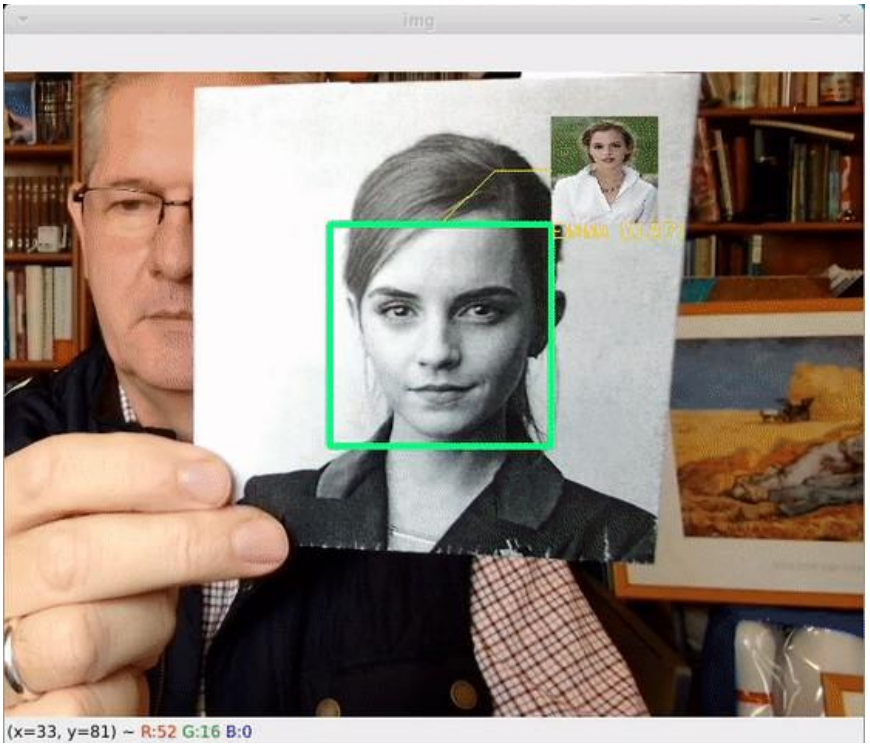


Figura 93: Captura WEB. Emma Watson. Fuente: Propia

Es obvio el tremendo impacto social que tiene el reconocimiento facial por su uso actual y el potencial futuro. Hay que tener en cuenta que actualmente hasta el más básico de los smartphones cuenta con un reconocimiento facial. El registro de datos faciales tiene tal potencial económico, político, social, entre otras. que ha abierto diversos debates a nivel internacional. Empresas como Meta [17], entre otras, han tenido que realizar un borrado masivo de datos biométricos por la presión mediática y gubernamental. Muchos gobiernos han realizado o están realizando revisiones de sus legislaciones para abordar adecuadamente la garantía de los derechos individuales frente al uso de tales datos.

6 - Líneas futuras

Aunque las ConvNet han demostrado una eficacia que roza el 100%, hay aspectos en el reconocimiento en general que estas redes no abordan correctamente.

El siguiente enlace es un video donde G.J. Hinton, describe CapsNets - Capsule Networks, un nuevo tipo de red neuronal para abordar las deficiencias de las ConvNet.

Geoffrey Hinton talks "What is wrong with convolutional neural nets ?"

Las ConvNet no codifican las relaciones de posición y orientación entre las características y, por lo tanto, pueden errar fácilmente. Capsule Networks [105] ha sido concebida por Hinton et al. como una arquitectura más robusta capaz de almacenar información de poses y relaciones espaciales para reconocer objetos con fuertes variaciones en su perspectiva visual (planta, alzado o perfil de un mismo objeto, por ejemplo). Intenta imitar el procedimiento visual del cerebro.

La idea que subyace tras CapsNet [105] es que la representación de los objetos en el cerebro no depende de la perspectiva con la que recibimos información visual. El cerebro extrae una representación jerárquica de nuestro entorno e intenta hacerla corresponder con patrones y relaciones previamente aprendidos y almacenados en el cerebro.

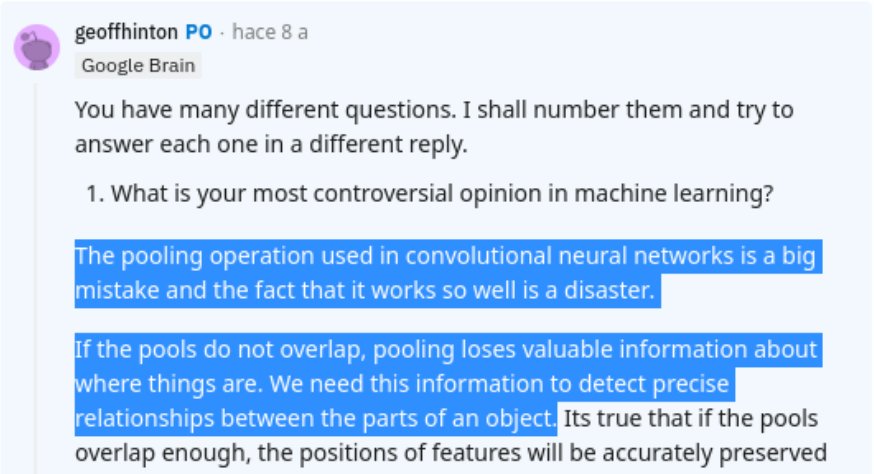


Figura 94. Comentario de Hinton. Fuente: Reddit [111]

Hinton: *"... La operación de agrupación utilizada en las redes neuronales convolucionales es un gran error y el hecho de que funcione tan bien es un desastre. Si las agrupaciones no se superponen, la agrupación pierde información valiosa sobre dónde están las cosas. Necesitamos esta información para detectar relaciones precisas entre las partes de un objeto…. "*

" …se puede eliminar la agrupación máxima [126] y seguir obteniendo buenos resultados con las ConvNet tradicionales, pero aún no resuelven el problema clave: la representación de datos internos de ***una red neuronal convolucional no tiene en cuenta las jerarquías espaciales importantes entre objetos simples y complejos****…."*

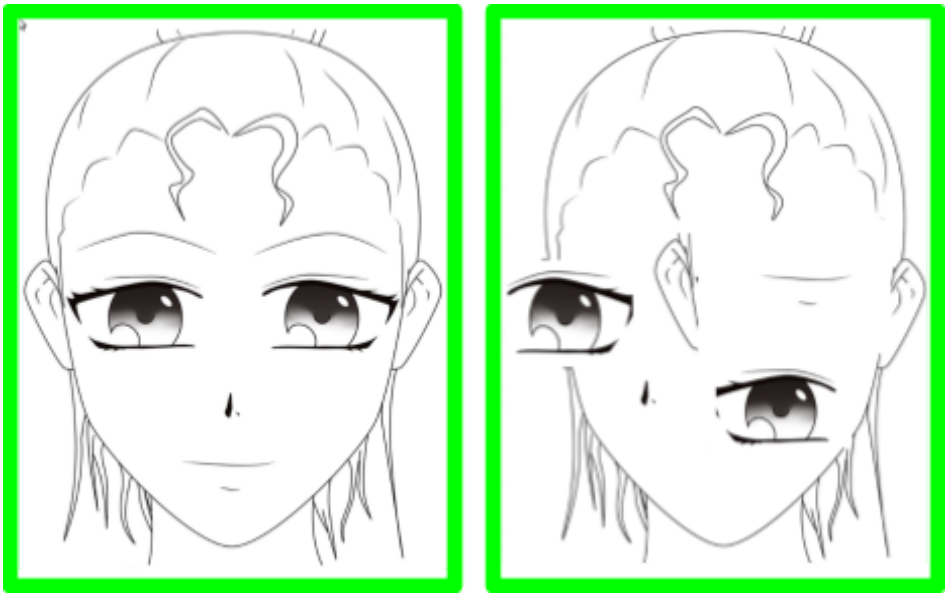


Figura 95. Posiciones Faciales Relativas. Fuente: [106]

El hecho de representar los elementos que constituyen un rostro en el mismo espacio contenedor no garantiza que estén bien colocados. Las redes de cápsulas buscan incorporar información relativa a las relaciones espaciales entre los componentes y sus orientaciones.

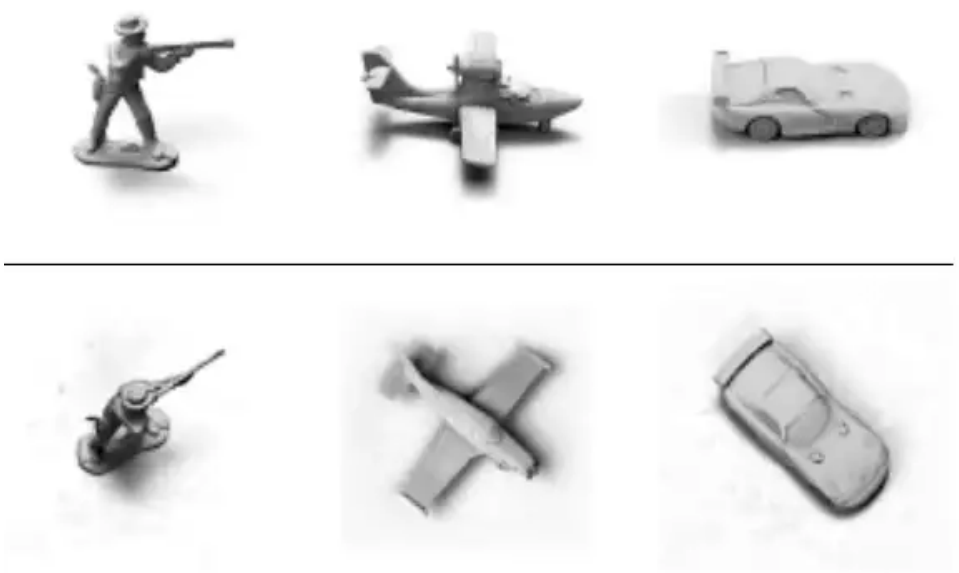


Figura 96: Figura con distintas vistas. Fuente: [110]

La red de cápsulas identifica mejor que otros modelos neuronales que las imágenes en las filas superior e inferior pertenecen a las mismas clases [108-110]

Figura 97: Diferentes vistas de la Estatua de la Libertad. Fuente: [110]

Una red convolucional actual sería incapaz de reconocer estas 5 imágenes de la Estatua de la Libertad como una identidad única, frente a CapsNet que busca cubrir esta deficiencia.

Ya existen algunos estudios de reconocimiento facial con CapsNet, como el realizado por Chui et al. [107] aunque aún con bajos resultados 93,7% sobre LFW en comparativa con las actuales ConvNet. Por otro lado, en su desempeño principal, sobre rostros no vistos previamente, rotados o desenfocados, iguala o supera el rendimiento de las ConvNet de última generación. Los autores nos sugieren que, comparando resultados previos con los experimentados con CapsNet, pueden concluir que las CapsNet pueden superar a las ConvNet más profundas en datos transformados no vistos debido a sus propiedades de equivarianza únicas.

Las Redes Neuronales Encapsuladas CapsNet, junto a las Redes Generativas Antagónicas GAN pueden ser una línea para mejorar los excelentes resultados actuales en la biometría facial con ConvNet.

7 - Bibliografía

[1] Deep Learning - Yoshua Bengio, Ian Goodfellow, Aaron Courville - Deep Learning - MIT Press - 2016 (Oct-2022)

[2] Charu C. Aggarwal - Neural Networks and Deep Learning - SPRINGER - 2018 - ISBN 9783319944623

[3] Josh Patterson & Adam Gibson - Deep Learning. A PRACTITIONER'S APPROACH - O'REILLY - 2018 - ISBN 9781491914250

[4] François Chollet - Deep Learning with Python - MANNING - 2017 - ISBN 9781617294433

[5]·https://hipertextual.com/2022/05/ia-reconocimiento-facial-pone-fotos-privadas-servicio-cualquiera-pimeyes (Oct-2022)

[6] https://en.wikipedia.org/wiki/Woody_Bledsoe (Oct-2022)

[7]·https://www.rand.org/pubs/research_memoranda/RM4122.html (Oct-2022)

[8] Sirovich, Lawrence & Kirby, M. (1987). Low-Dimensional Procedure for the Characterization of Human Faces. Journal of the Optical Society of America. A, Optics and image science. 4. 519-24.10.1364/JOSAA.4.000519 (Oct-2022)

[9] Karamizadeh, Sasan & Abdullah, Shahidan & Manaf, Azizah & Zamani, Mazdak & Hooman, Alireza. (2013). An Overview of Principal Component Analysis. Journal of Signal and Information Processing. 10.4236/jsip.2013.43B031. (Oct-2022)

[10] Matthew Turk, Alex Pentland; Eigenfaces for Recognition. *J Cogn Neurosci* 1991; 3 (1): 71–86 (Oct-2022)

[11] https://es.wikipedia.org/wiki/Warren_McCulloch (Oct-2022)

[12] https://es.wikipedia.org/wiki/Walter_Pitts (Oct-2022)

[13] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. 2009. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09). Association for Computing Machinery, New York, NY, USA, 609–616. https://doi.org/10.1145/1553374.1553453 (Oct-2022)

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. https://doi.org/10.1145/3065386 (Oct-2022)

[15] https://openai.com/blog/ai-and-compute/ (Oct-2022)

[16]·https://www.eff.org/deeplinks/2021/07/clearviews-face-surveillance-still-has-no-first-amendment-defense (Oct-2022)

[17]·https://www.xataka.com/privacidad/meta-borrara-datos-reconocimiento-facial-mil-millones-personas-usados-facebook-golpes-efectos-zuckerberg (Oct-2022)

[18]·https://towardsdatascience.com/using-linkedin-profile-pictures-for-facial-recognition-8be709e8fac (Oct-2022)

[19]·https://www.xataka.com/privacidad/test-distopico-corea-sur-10-820-camaras-reconocimiento-facial-para-rastrear-infectados-covid-19 (Oct-2022)

[20]·https://www.eldiario.es/tecnologia/mercadona-paga-multa-2-5-millones-proteccion-datos-sistema-reconocimiento-facial_1_8161764.html (Oct-2022)

[21]·https://bandaancha.eu/articulos/asi-abis-sistema-reconocimiento-facial-9930 (Oct-2022)

[22]·https://www.elespanol.com/omicrono/tecnologia/20200806/nuevo-problema-seguridad-fronteras-enganan-reconocimiento-facial/510699972_0.html (Oct-2022)

[23]·https://www.vozpopuli.com/economia-y-finanzas/reconocimiento-facial-causas-pendientes_0_1375363234.html (Oct-2022)

[24]·https://arstechnica.com/tech-policy/2020/06/police-arrested-wrong-man-based-on-facial-recognition-fail-aclu-says/?utm_brand=arstechnica&utm_source=twitter&utm_social-type=owned&utm_medium=social (Oct-2022)

[25]·https://www.europapress.es/portaltic/sector/noticia-ue-plantea-prohibir-anos-reconocimiento-facial-lugares-publicos-analizar-riesgos-20200117171823.html (Oct-2022)

[26]·https://es.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining#/media/Archivo:CRISP-DM_Process_Diagram.png (Oct-2022)

[27] Practical Convolutional Neural Networks: Implement advanced deep learning models using Python.Sewak, Mohit; Karim, Md. Rezaul; Pujari, Pradeep 2018 Packt https://www.packtpub.com/product/practical-convolutional-neural-networks/9781788392303

[28] G. Montavon, G.B. Orr et al. - Neural Networks: Tricks of the Trade.2ed. ED.Springer 1998 (Oct-2022)

[29]·https://ikhlestov.github.io/pages/machine-learning/convolutional-layers/ (Oct-2022)

[30]·https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/#:~:text=Max%20pooling%20is%20a%20pooling,of%20the%20previous%20feature%20map (Oct-2022)

[31] https://github.com/vdumoulin/conv_arithmetic (Oct-2022)

[33]·http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf (Oct-2022)

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. https://doi.org/10.1145/3065386

https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf (Oct-2022)

[35] Simonyan, Karen and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." CoRR abs/1409.1556 (2014)
https://arxiv.org/abs/1409.1556 (Oct-2022)

[36] Szegedy, Christian et al. "Going deeper with convolutions." *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014): 1-9
https://arxiv.org/abs/1409.4842 (Oct-2022)

[37] https://pytorch.org/hub/pytorch_vision_googlenet (Oct-2022)

[38] He, Kaiming et al. "Deep Residual Learning for Image Recognition." *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015): 770-778.
https://arxiv.org/abs/1512.03385 (Oct-2022)

[39] Girshick, Ross B. et al. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation." *2014 IEEE Conference on Computer Vision and Pattern Recognition* (2013): 580-587.
https://arxiv.org/abs/1311.2524 (Oct-2022)

[40] Moschoglou, Stylianos et al. "AgeDB: The First Manually Collected, In-the-Wild Age Database." 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2017): 1997-2005.
https://ibug.doc.ic.ac.uk/media/uploads/documents/agedb.pdf (Oct-2022)

[41] Zheng, Tianyue et al. "Cross-Age LFW: A Database for Studying Cross-Age Face Recognition in Unconstrained Environments." ArXiv abs/1708.08197 (2017)
https://arxiv.org/abs/1708.08197 (Oct-2022)

[42] Zheng, Tianyue and Weihong Deng. "Cross-Pose LFW : A Database for Studying Cross-Pose Face Recognition in Unconstrained Environments." (2018).
http://www.whdeng.cn/CPLFW/Cross-Pose-LFW.pdf (Oct-2022)

[43] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller.Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. University of Massachusetts, Amherst, Technical Report 07-49, October, 2007.
http://vis-www.cs.umass.edu/lfw/lfw.pdf (Oct-2022)
http://vis-www.cs.umass.edu/lfw/ (Oct-2022)

[44] Lior Wolf, Tal Hassner and Itay Maoz
Face Recognition in Unconstrained Videos with Matched Background Similarity.IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2011.
https://www.cs.tau.ac.il/~wolf/ytfaces/WolfHassnerMaoz_CVPR11.pdf (Oct-2022)
https://www.cs.tau.ac.il/~wolf/ytfaces/index.html#reference (Oct-2022)

[45] Y. Taigman, M. Yang, M. Ranzato and L. Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification, " 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1701-1708, doi: 10.1109/CVPR.2014.220.
https://research.facebook.com/publications/deepface-closing-the-gap-to-human-level-performance-in-face-verification/
https://www.cs.toronto.edu/~ranzato/publications/taigman_supp_material_cvpr2014.pdf
https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14
(Oct-2022)

[46] Y. Sun, X. Wang and X. Tang, "Deep Learning Face Representation from Predicting 10,000 Classes, " 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1891-1898, doi: 10.1109/CVPR.2014.244.
http://mmlab.ie.cuhk.edu.hk/pdf/YiSun_CVPR14.pdf (Oct-2022)

[47] Sun, Y., Chen, Y., Wang, X., & Tang, X. (2014). Deep Learning Face Representation by Joint Identification-Verification. NIPS.
https://arxiv.org/abs/1406.4773 (Oct-2022)

[48] Sun, Y., Wang, X., & Tang, X. (2015). Deeply learned face representations are sparse, selective, and robust. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2892-2900.
https://arxiv.org/abs/1412.1265 (Oct-2022)

[49] APA: Sun, Y., Liang, D., Wang, X., & Tang, X. (2015). DeepID3: Face Recognition with Very Deep Neural Networks. ArXiv, abs/1502.00873.
https://arxiv.org/abs/1502.00873 (Oct-2022)

[50] Taigman, Y., Yang, M., Ranzato, M., & Wolf, L. (2015). Web-scale training for face identification. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2746-2754.
https://arxiv.org/abs/1406.5266 (Oct-2022)

[51] Liu, J., Deng, Y., Bai, T., & Huang, C. (2015). Targeting Ultimate Accuracy: Face Recognition via Deep Embedding. ArXiv, abs/1506.07310.
https://arxiv.org/abs/1506.07310 (Oct-2022)

[52] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 815-823.
https://arxiv.org/abs/1503.03832 (Oct-2022)

[53] Parkhi, O.M., Vedaldi, A., Zisserman, A., Deep Face Recognition. British Machine Vision Conference, 2015
https://www.robots.ox.ac.uk/~vgg/publications/2015/Parkhi15/parkhi15.pdf (Oct-2022)

[54] Masi, I., Tran, A.T., Hassner, T., Leksut, J.T., & Medioni, G.G. (2016). Do We Really Need to Collect Millions of Faces for Effective Face Recognition? ArXiv, abs/1603.07057.
https://arxiv.org/abs/1603.07057 (Oct-2022)

[55] Zhang, X., Fang, Z., Wen, Y., Li, Z., & Qiao, Y. (2017). Range Loss for Deep Face Recognition with Long-Tailed Training Data. 2017 IEEE International Conference on Computer Vision (ICCV), 5419-5428.
https://arxiv.org/abs/1611.08976 (Oct-2022)

[56] Wen, Y.; Zhang, K.; Li, Z.; Qiao, Y. A discriminative feature learning approach for deep face recognition. In

Proceedings of the 14th European Conference on Computer Vision (ECCV), Amsterdam, 2016
https://kpzhang93.github.io/papers/eccv2016.pdf (Oct-2022)

[57] Liu, W., Wen, Y., Yu, Z., & Yang, M. (2016). Large-Margin Softmax Loss for Convolutional Neural Networks. ArXiv, abs/1612.02295.
https://arxiv.org/abs/1612.02295 (Oct-2022)

[58] Wu, Y., Liu, H., Li, J.Y., & Fu, Y.R. (2017). Deep Face Recognition with Center Invariant Loss. Proceedings of the on Thematic Workshops of ACM Multimedia 2017.
https://www.semanticscholar.org/paper/Deep-Face-Recognition-with-Center-Invariant-Loss-Wu-Liu/feea73095b1be0cbae1ad7af8ba2c4fb6f316d35 (Oct-2022)

[59] Ranjan, R., Castillo, C.D., & Chellappa, R. (2017). L2-constrained Softmax Loss for Discriminative Face Verification. ArXiv, abs/1703.09507.
https://arxiv.org/abs/1703.09507 (Oct-2022)

[60] J. Deng, Y. Zhou and S. Zafeiriou, "Marginal Loss for Deep Face Recognition, " 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017, pp. 2006-2014, doi: 10.1109/CVPRW.2017.251
https://openaccess.thecvf.com/content_cvpr_2017_workshops/w33/papers/Deng_Marginal_Loss_for_CVPR_2017_paper.pdf (Oct-2022)

[61] Wang, F., Xiang, X., Cheng, J., & Yuille, A.L. (2017). NormFace: L2 Hypersphere Embedding for Face Verification. Proceedings of the 25th ACM international conference on Multimedia.
https://arxiv.org/abs/1704.06369 (Oct-2022)

[62] Liu, Y., Li, H., & Wang, X. (2017). Rethinking Feature Discrimination and Polymerization for Large-scale Recognition. ArXiv, abs/1710.00870.
https://arxiv.org/abs/1710.00870 (Oct-2022)

[63] Hasnat, M.A., Bohné, J., Milgram, J., Gentric, S., & Chen, L. (2017). von Mises-Fisher Mixture Model-based Deep learning: Application to Face Verification. ArXiv, abs/1706.04264.
https://arxiv.org/abs/1706.04264 (Oct-2022)

[64] Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B., & Song, L. (2017). SphereFace: Deep Hypersphere Embedding for Face Recognition. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6738-6746.
https://arxiv.org/abs/1704.08063 (Oct-2022)

[65] Zheng, Y., Pal, D.K., & Savvides, M. (2018). Ring Loss: Convex Feature Normalization for Face Recognition. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5089-5097.
https://arxiv.org/abs/1803.00130 (Oct-2022)

[66] Guo, Y., & Zhang, L. (2017). One-shot Face Recognition by Promoting Underrepresented Classes. ArXiv, abs/1707.05574.
https://arxiv.org/abs/1707.05574 (Oct-2022)

[67] Wei, X., Wang, H., Scotney, B.W., & Wan, H. (2020). Minimum Margin Loss for Deep Face Recognition. Pattern Recognit., 97.
https://arxiv.org/abs/1805.06741 (Oct-2022)

[68] Wang, H., Wang, Y., Zhou, Z., Ji, X., Li, Z., Gong, D., Zhou, J., & Liu, W. (2018). CosFace: Large Margin Cosine Loss for Deep Face Recognition. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5265-5274.
https://arxiv.org/abs/1801.09414 (Oct-2022)

[69] Wang, F., Cheng, J., Liu, W., & Liu, H. (2018). Additive Margin Softmax for Face Verification. IEEE Signal Processing Letters, 25, 926-930.
https://arxiv.org/abs/1801.05599 (Oct-2022)

[70] Wu, X., He, R., Sun, Z., & Tan, T. (2018). A Light CNN for Deep Face Representation With Noisy Labels. IEEE Transactions on Information Forensics and Security, 13, 2884-2896.
https://arxiv.org/abs/1511.02683 (Oct-2022)

[71] Hayat, M., Khan, S.H., Zamir, W., Shen, J., & Shao, L. (2019). Max-margin Class Imbalanced Learning with Gaussian Affinity. ArXiv, abs/1901.07711.
https://arxiv.org/abs/1901.07711 (Oct-2022)

[72] Huang, C., Li, Y., Loy, C.C., & Tang, X. (2020). Deep Imbalanced Learning for Face Recognition and Attribute Prediction. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42, 2781-2794
https://arxiv.org/abs/1806.00194 (Oct-2022)

[73] Song, L., Gong, D., Li, Z., Liu, C., & Liu, W. (2019). Occlusion Robust Face Recognition Based on Mask Learning With Pairwise Differential Siamese Network. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 773-782.
https://arxiv.org/abs/1908.06290 (Oct-2022)

[74] Kang, B., Kim, Y., Jun, B., & Kim, D. (2019). Attentional Feature-Pair Relation Networks for Accurate Face Recognition. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 5471-5480.
https://arxiv.org/abs/1908.06255 (Oct-2022)

[75] Yin, X., Yu, X., Sohn, K., Liu, X., & Chandraker, M. (2018). Feature Transfer Learning for Deep Face Recognition with Long-Tail Data. ArXiv, abs/1803.09014.
https://arxiv.org/abs/1803.09014 (Oct-2022)

[76] Ben Fredj, H., Bouguezzi, S. & Souani, C. Face recognition in unconstrained environment with CNN. Vis Comput 37, 217–226 (2021).
https://doi.org/10.1007/s00371-020-01794-9 (Oct-2022)

[77] Wenming, Yang & Sun, Jingna & Gao, Riqiang & Xue, Jing-Hao & Liao, Qingmin. (2019). Inter-class angular margin loss for face recognition. Signal Processing: Image Communication. 80.115636.10.1016/j.image.2019.115636.
https://www.researchgate.net/publication/335805835_Inter-class_angular_margin_loss_for_face_recognition (Oct-2022)

[78] Ling, H., Wu, J., Huang, J. et al. Attention-based convolutional neural network for deep face recognition. Multimed Tools Appl 79, 5595–5616 (2020).
https://doi.org/10.1007/s11042-019-08422-2 (Oct-2022)

[79] Y. Wu et al., "Rotation Consistent Margin Loss for Efficient Low-Bit Face Recognition, " 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition

(CVPR), 2020, pp. 6865-6875, doi: 10.1109/CVPR42600.2020.00690. https://openaccess.thecvf.com/content_CVPR_2020/papers/Wu_Rotation_Consistent_Margin_Loss_for_Efficient_Low-Bit_Face_Recognition_CVPR_2020_paper.pdf (Oct-2022)

[80] B. Wu and H. Wu, "Angular Discriminative Deep Feature Learning for Face Verification, " ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 2133-2137, doi: 10.1109/ICASSP40776.2020.9053675. http://huamingwu.cn/PDF/ICASSP2.pdf (Oct-2022)

[81] Huang, Y., Wang, Y., Tai, Y., Liu, X., Shen, P., Li, S., Li, J., & Huang, F. (2020). CurricularFace: Adaptive Curriculum Learning Loss for Deep Face Recognition. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 5900-5909. https://arxiv.org/abs/2004.00288 (Oct-2022)

[82] Kim, Y., Park, W., Roh, M., & Shin, J. (2020). GroupFace: Learning Latent Groups and Constructing Group-Based Representations for Face Recognition. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 5620-5629. https://arxiv.org/abs/2005.10497 (Oct-2022)

[83] Jiao, J., Liu, W., Mo, Y. et al. Dyn-arcFace: dynamic additive angular margin loss for deep face recognition. Multimed Tools Appl 80, 25741–25756 (2021). https://doi.org/10.1007/s11042-021-10865-5 (Oct-2022)

[84] Boutros, F., Damer, N., Kirchbuchner, F., & Kuijper, A. (2022). ElasticFace: Elastic Margin Loss for Deep Face Recognition. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 1577-1586. https://arxiv.org/abs/2109.09416 (Oct-2022)

[85] Seo Y, Yu NY. Additive Orthant Loss for Deep Face Recognition. Applied Sciences. 2022; 12(17):8606. https://doi.org/10.3390/app12178606 (Oct-2022)

[86] Deng, J., Guo, J., & Zafeiriou, S. (2019). ArcFace: Additive Angular Margin Loss for Deep Face Recognition. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 4685-4694. https://arxiv.org/abs/1801.07698 (Nov-2022)

[87] Sang, M., Chen, J., Li, M., Tan, P., Pan, A., Zhao, S., & Yang, Y. (2022). InterFace: Adjustable Angular Margin Inter-class Loss for Deep Face Recognition. ArXiv, abs/2210.02018. (Nov-2022) https://arxiv.org/abs/2210.02018 (Oct-2022)

[88] F. Wang, L. Chen, C. Li, S. Huang, Y. Chen, C. Qian, and C. Change Loy, "The devil of face recognition is in the noise, " in The European Conference on Computer Vision (ECCV), September 2018. https://arxiv.org/abs/1807.11649 (Nov-2022)

[89] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. https://arxiv.org/abs/1409.1556 (Nov-2022)

[90] Meta borrará los datos de reconocimiento facial de más de mil millones de personas usados en Facebook: más golpes de efectos de Zuckerberg https://www.xataka.com/privacidad/meta-borrara-datos-reconocimiento-facial-mil-millones-personas-usados-facebook-golpes-efectos-zuckerberg (Nov-2022)

[91] Fierro, Atoany N., Nakano, Mariko, Yanai, Keiji, & Pérez, Héctor M. (2019). Redes Convolucionales Siamesas y Tripletas para la Recuperación de Imágenes Similares en Contenido. Información tecnológica, 30(6), 243-254. https://dx.doi.org/10.4067/S0718-07642019000600243 (Nov-2016)

[92] P.Jonathon Phillips, Harry Wechsler, Jeffery Huang, Patrick J. Rauss, The FERET database and evaluation procedure for face-recognition algorithms, Image and Vision Computing, Volume 16, Issue 5, 1998, Pages 295-306, ISSN 0262-8856, https://doi.org/10.1016/S0262-8856(97)00070-X (Nov-2002) https://www.sciencedirect.com/science/article/pii/S026288569700070X(Nov-2022)

[93] Phillips, P., Flynn, P., Scruggs, W., Bowyer, K., Chang, J., Hoffman, K., Marques, J., Min, J. and Worek, W. (2005), Overview of the Face Recognition Grand Challenge, IEEE Computer Society International Conference on Computer Vision and Pattern Recognition, san diego, CA https://www.nist.gov/publications/overview-face-recognition-grand-challenge (Nov-2022)

[94] The Deep Learning Workshop.M.Moorcarme et al. 2021 Packt https://www.packtpub.com/product/the-deep-learning-workshop/9781839219856 (Nov-2022)

[95] Imágen VGGNet16. https://www.cs.toronto.edu/~frossard/post/vgg16/ (Nov-2022)

[96] L. Wiskott, N. Krüger, N. Kuiger and C. von der Malsburg, "Face recognition by elastic bunch graph matching, " in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 775-779, July 1997, doi: 10.1109/34.598235. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=598235 (Nov-2022)

[97] Face Authentication Using Adapted Local Binary Pattern Histograms https://link.springer.com/content/pdf/10.1007/11744085_25.pdf (Nov-2022)

[98] Zhou, Erjin et al. "Naive-Deep Face Recognition: Touching the Limit of LFW Benchmark or Not?" ArXiv abs/1501.04690 (2015) https://arxiv.org/abs/1501.04690 (Nov-2022)

[99] Yi, Dong et al. "Learning Face Representation from Scratch." *ArXiv* abs/1411.7923 (2014) https://arxiv.org/abs/1411.7923 (Nov-2022)

[100] Guo, Yandong et al. "MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition." *European Conference on Computer Vision* (2016) https://arxiv.org/abs/1607.08221 (Nov-2022)

[101] Cao, Qiong et al. "VGGFace2: A Dataset for Recognising Faces across Pose and Age." *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)* (2017): 67-74. https://arxiv.org/abs/1710.08092 (Nov-2022)

https://www.robots.ox.ac.uk/~vgg/data/vgg_face2/ (Nov-2022)

[102] Kemelmacher-Shlizerman, Ira et al. "The MegaFace Benchmark: 1 Million Faces for Recognition at Scale." *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015): 4873-4882. https://arxiv.org/abs/1512.00596 (Nov-2022)

[103] Hu, Jie et al. "Squeeze-and-Excitation Networks." *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2017): 7132-7141. https://arxiv.org/abs/1709.01507 (Nov-2022)

[104] Eyiokur, Fevziye Irem et al. "A survey on computer vision based human analysis in the COVID-19 era." *Image and Vision Computing* 130 (2022): 104610 - 104610. https://arxiv.org/abs/2211.03705 (Nov-2022)

[105] Sabour, Sara et al. "Dynamic Routing Between Capsules." *ArXiv* abs/1710.09829 (2017): n. pag. https://arxiv.org/abs/1710.09829 (Nov-2022)

[106] Benjamin Planche, Eliot Andres. Hands-On Deep Learning Architectures with Python. Packt 2019 https://www.packtpub.com/product/hands-on-deep-learning-architectures-with-python/9781788998086 (Dic-2022)

[107] A Chui, A Patnaik, K Ramesh, L Wang "Capsule Networks and Face Recognition" - Lindawangg. github. io, 2019 https://lindawangg.github.io/projects/capsnet.pdf (Dic-2022)

[108] Hinton, G.E., Krizhevsky, A., Wang, S.D. (2011). Transforming Auto-Encoders. In: Honkela, T., Duch, W., Girolami, M., Kaski, S. (eds) Artificial Neural Networks and Machine Learning – ICANN 2011. ICANN 2011. Lecture Notes in Computer Science, vol 6791. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-21735-7_6 (Dic-2022) http://www.cs.toronto.edu/~fritz/absps/transauto6.pdf (Dic-2022)

[109] Geoffrey E Hinton,Sara Sabour, Nicholas Frosst, "Matrix capsules with EM routing" (2017) https://openreview.net/pdf?id=HJWLfGWRb (Dic-2022)

[110] Hui, J. "Understanding Matrix capsules with EM Routing (Based on Hinton's Capsule Networks)" https://jhui.github.io/2017/11/14/Matrix-Capsules-with-EM-routing-Capsule-Network/ (Dic-2022)
[111] Cita G. Hinton. Reddit https://www.reddit.com/r/MachineLearning/comments/2lmo0l/comment/clyj4jv/ (Dic-2022)

[112] Marriott, Richard T. et al. "A 3D GAN for Improved Large-pose Facial Recognition." 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020): 13440-13450. https://arxiv.org/abs/2012.10545 (Dic-2022)

[113] Sengupta, Soumyadip et al. "Frontal to profile face verification in the wild." 2016 IEEE Winter Conference on Applications of Computer Vision (WACV) (2016): 1-9. http://www.cfpw.io/paper.pdf (Dic-2022)

[114] Whitelam, Cameron et al. "IARPA Janus Benchmark-B Face Dataset." *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2017): 592-600. http://biometrics.cse.msu.edu/Publications/Face/Whitelametal_IARPAJanusBenchmark-BFaceDataset_CVPRW17.pdf (Dic-2022)

[115] Maze, Brianna et al. "IARPA Janus Benchmark - C: Face Dataset and Protocol." *2018 International Conference on Biometrics (ICB)* (2018): 158-165. http://biometrics.cse.msu.edu/Publications/Face/Mazeetal_IARPAJanusBenchmarkCFaceDatasetAndProtocol_ICB2018.pdf (Dic-2022)

[116] Zhu, Zheng Hua et al. "WebFace260M: A Benchmark for Million-Scale Deep Face Recognition." IEEE Transactions on Pattern Analysis and Machine Intelligence PP (2022): 1-1. https://arxiv.org/abs/2204.10149 (Dic-2022)

[117] Nech, Aaron and Ira Kemelmacher-Shlizerman. "Level Playing Field for Million Scale Face Recognition." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017): 3406-3415. https://arxiv.org/abs/1705.00393 (Dic-2022)

[118]·https://www.dupalu.com/2015/02/especial-cine-las-12-actrices-mas-guapa.html (Oct-2022)

[119]·https://github.com/swghosh/DeepFace/releases/tag/weights-vggface2-2d-aligned (Nov-2022)

[120]·https://github.com/swghosh/DeepFace/blob/master/deepface-architecture.png (Nov-2022)

[121] https://github.com/swghosh/DeepFace (Nov-2022)

[122] https://github.com/swghosh/DeepFace/releases (Nov-2022)

[123] https://github.com/serengil (Nov-2022)

[124] https://github.com/ox-vgg/vgg_face2 (Nov-2022)

[125]·https://academictorrents.com/details/535113b8395832f09121bc53ac85d7bc8ef6fa5b (Nov-2022)

[126] Springenberg, Jost Tobias et al. "Striving for Simplicity: The All Convolutional Net." *CoRR* abs/1412.6806 (2014) https://arxiv.org/abs/1412.6806 (Dic-2022)