UNIVERSIDAD DEL BÍO-BÍO
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

Aceleración Hardware de un Algoritmo de Aprendizaje Activo para Clasificadores de Imágenes

Angelo Roberto Barbieri Figueroa

CONCEPCIÓN - CHILE
JULIO, 2024

Resumen

Índice general

Resumen **3**

1. Introducción **1**

2. Estado del arte y motivaciones **2**
2.1. Visión por computador 2
2.2. Visión por computador en clasificación 3
2.2.1. ¿Qué es una imagen? (Entrada de datos) 4
2.2.2. Preprocesamiento 5
2.2.3. Extracción de características (enfoque tradicional) 6
2.2.4. Clasificación (enfoque tradicional) 7
2.2.5. Redes neuronales convolucionales 8
2.2.5.1. Red neuronal 8
2.2.5.2. Arquitectura de red neuronal convolucional (CNN) 10
2.2.5.3. Arquitecturas Clásicas 14
2.2.6. Transformadores visuales (ViT) 19
2.2.6.1. Arquitectura de transformador visual 19

3. Materiales y métodos **21**
3.1. Conjuntos de datos y preprocesamiento 21
3.1.1. WM-811K 21
3.1.2. Brain Tumor Classification (MRI) 23

4. Conclusiones y trabajo futuro **27**

Referencias **28**

Índice de cuadros

Índice de figuras

2.2.1.Pipeline de visión por computador. Fuente: Adaptado de [1]. . . . 3
2.2.2.Comparación de Imagen en Escala de Grises y en Color. Fuente: Adaptado de [1]. 5
2.2.3.Preprocesamiento para quitar ruido de una imagen. Fuente: Elaboración propia. 6
2.2.4.Red neuronal convolucional. Fuente: Adaptado de [1]. 8
2.2.5.Neurona artificial. Fuente: Adaptado de [1] 9
2.2.6.Red perceptrón multicapa. Fuente: Adaptado de [1] 10
2.2.7.Funciones de activación. Fuente: Adaptado de [2]. 12
2.2.8.Arquitectura de LeNet. Fuente: Adaptado de [3, 1]. 14
2.2.9.Arquitectura AlexNet. Fuente: Adaptado de [4, 1]. 15
2.2.10Arquitectura VGGNet16. Fuente: Adaptado de [5, 1]. 16
2.2.11Módulo Inception. Fuente: Adaptado de [6, 1]. 17

3.1.1.Distribución de clases de WM-811K. Fuente: Elaboración propia. . 22
3.1.2.Clases de WM-811K. Fuente: Elaboración propia. 23
3.1.3.Distribución de clases de Brain Tumor. Fuente: Elaboración propia. 24
3.1.4.Procedimiento para recortar una imagen. Fuente: Elaboración propia. 25
3.1.5.Clases de Brain Tumor. Fuente: Elaboración propia. 26

Capítulo 1

Introducción

Capítulo 2

Estado del arte y motivaciones

2.1. Visión por computador

Es esencial que un sistema de inteligencia artificial (AI) pueda comprender su entorno y, de esta forma, tomar decisiones basadas en su comprensión. La visión por computadora (CV) se define como un área de la inteligencia artificial que se ocupa de la percepción visual, extrayendo e interpretando información relevante desde imágenes digitales, videos u otras entradas visuales. Basado en la visión humana, un sistema de visión consiste en dos componentes principales: un dispositivo de adquisición de datos que captura información (función del ojo humano) y un dispositivo que es capaz de interpretar la información detectada (función del cerebro) [1, 7, 8].

Dispositivos de adquisición de datos

Entre las capacidades sensoriales humanas, la visión es el sentido más avanzado. Sin embargo, mientras que la visión humana está limitada a una estrecha banda de luz visible del espectro electromagnético (EM), los dispositivos de adquisición de imágenes abarcan todo el espectro EM, desde los rayos gamma hasta las ondas de radio. Además, las tecnologías actuales permiten generar representaciones visuales a partir de fuentes que normalmente no se asocian con imágenes. Dado que los sistemas de visión están diseñados para realizar tareas específicas, es crucial seleccionar el dispositivo de detección que mejor se ajuste a las características del problema [1, 9].

Dispositivo interpretador

Como dispositivo interpretador, se utilizan algoritmos de visión por computador, que actúan como el *cerebro* del sistema de visión. Inspirados en el aprendizaje de las neuronas biológicas, donde una señal de salida se envía a otras neuronas conectadas si se activan suficientes señales de entrada, los científicos desarrollaron un cerebro artificial con neuronas artificiales. De este modo nacen las *Redes Neuronales Artificiales* (ANN). Aunque cada neurona realiza una función simple por sí sola, la agrupación de neuronas en capas y la conexión de múltiples capas entre sí forman una red capaz de aprender. El uso de redes con múltiples capas de neuronas se denomina *Aprendizaje Profundo* (DL) [1].

2.2. Visión por computador en clasificación

Clasificación es la tarea de asignar una etiqueta a una imagen desde un conjunto de categorías predefinidas [1]. Como se mencionó, los sistemas de visión se componen de dos principales elementos: dispositivo de adquisición de datos y dispositivo interpretador. El trabajo que realiza el dispositivo interpretador se divide en una serie de pasos como muestra la figura 2.2.1.

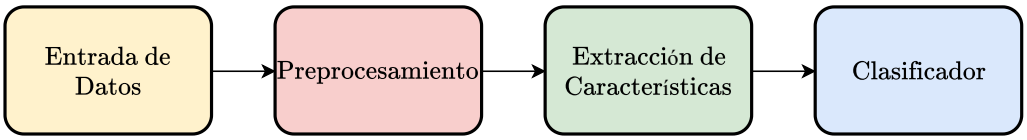


Figura 2.2.1: Pipeline de visión por computador. Fuente: Adaptado de [1].

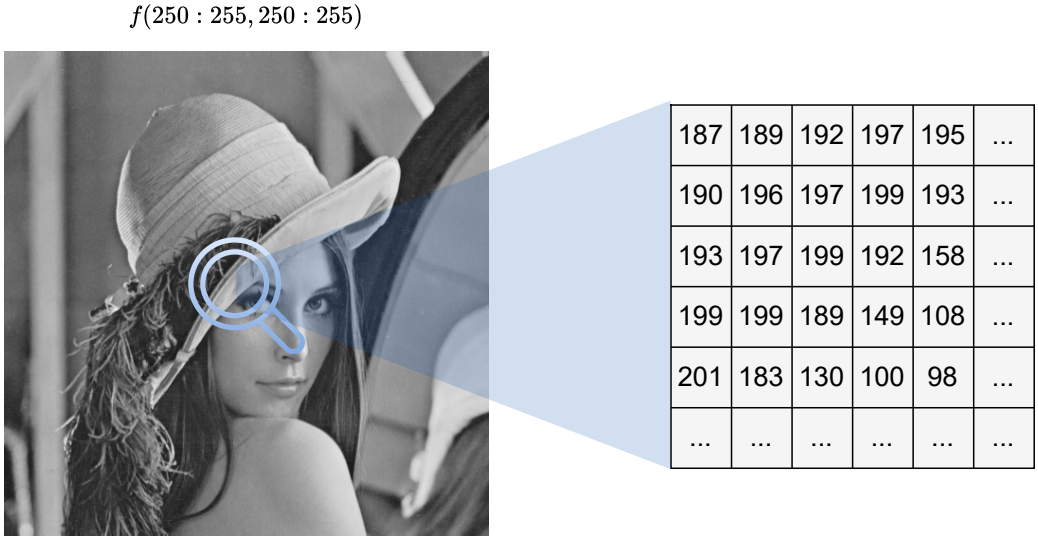
Para llevar a cabo la clasificación de imágenes, existen dos enfoques principales: el tradicional y el basado en Redes Neuronales. El enfoque tradicional utiliza métodos de extracción de características y algoritmos de clasificación convencionales. En contraste, el enfoque basado en redes neuronales emplea técnicas de aprendizaje profundo que integran tanto la extracción de características como la clasificación en un solo proceso.

2.2.1. ¿Qué es una imagen? (Entrada de datos)

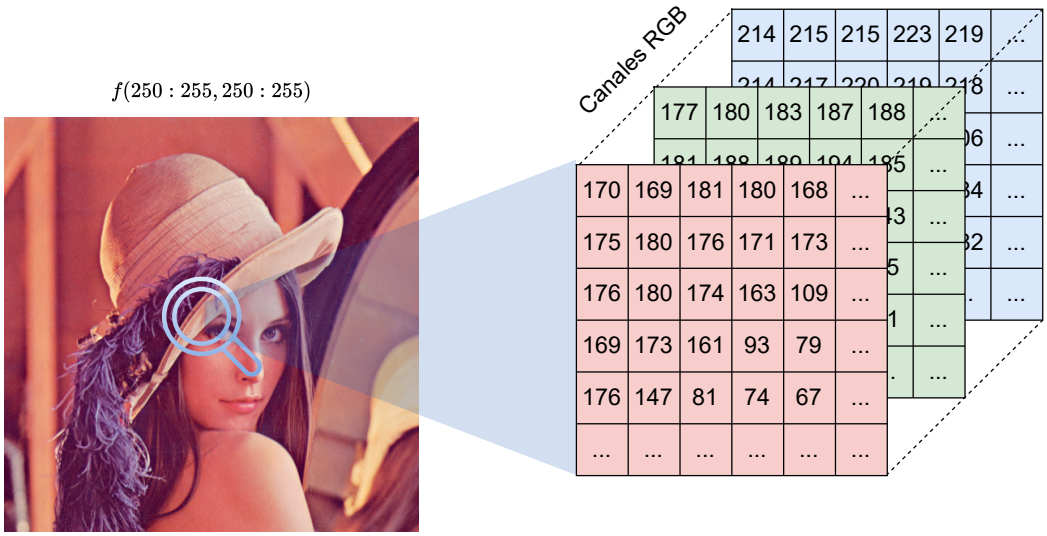
Una imagen puede ser definida como una función bidimensional $f(x, y)$, donde x e y son las coordenadas espaciales en un plano, y la amplitud de f para cualquier par de coordenadas (x, y) se denomina intensidad o nivel de gris en ese punto. Es importante notar que una imagen se compone de un número finito de elementos llamados píxeles, cada uno con una posición en el plano xy y un valor específico. En una imagen en escala de grises, el valor de intensidad de un píxel abarca un rango de 0 (negro) a 255 (blanco) usando 8 bits [9, 1].

En el caso de imágenes a color, cada píxel tiene un valor de intensidad para cada canal de color. Por ejemplo, en el sistema RGB, cada píxel se representa mediante su intensidad en el canal rojo, intensidad en el canal verde e intensidad en el canal azul. Esta representación se extiende a otros sistemas de color como HSV [1].

Dada la composición de una imagen digital, en un computador estas se tratan como matrices de píxeles. En una imagen en escala de grises, cada píxel determina el valor de intensidad de un solo color, lo que se puede representar mediante una matriz 2D. En contraste, en las imágenes a color, como en el sistema RGB, se utilizan tres matrices, una para cada canal: una matriz para la intensidad del color rojo, otra para la intensidad del color verde y una tercera para la intensidad del color azul. Por lo tanto, las imágenes a color se tratan como matrices 3D, donde la profundidad es tres [1].



(a) Imagen en escala de grises



(b) Imagen en color

Figura 2.2.2: Comparación de Imagen en Escala de Grises y en Color. Fuente: Adaptado de [1].

2.2.2. Preprocesamiento

En esta etapa, se realizan operaciones sobre las imágenes como redimensionamiento, normalización, aumento de datos, eliminación de ruido, corrección de iluminación, segmentación, transformaciones geométricas y filtrado de bordes. El objetivo es mejorar la calidad de la información, de modo que se facilite el análisis y procesamiento computacional en los pasos posteriores.

(a) Imagen con ruido **(b)** Imagen filtrada

Figura 2.2.3: Preprocesamiento para quitar ruido de una imagen. Fuente: Elaboración propia.

2.2.3. Extracción de características (enfoque tradicional)

El principal objetivo de la extracción características es obtener la información más relevante de los datos originales y representar esta información en un espacio dimensional reducido, donde el conjunto de características extraídas se denomina vector de características [10]. En el enfoque tradicional existen distintos métodos. La forma más primitiva consiste en considerar los píxeles individuales como características. En 1999, David Lowe presentó un método llamado características de escala invariante (SIFT) que transforma una imagen en múltiples vectores de características invariantes a transformaciones geométricas de la imagen, tales como traslación, cambios de escala y rotación. Además, estos vectores son parcialmente invariantes a los cambios de iluminación y adecuados para imágenes de 3 canales [11]. Una técnica similar a SIFT, pero que utiliza aproximaciones para acelerar el cálculo, reducir el tiempo de procesamiento y es más robusta, llamada características robusta aceleradas (SURF), fue presentada por Herbet Bay, Tinne Tuytelaars y Luc Van Gool en 2006 [12]. Patrón Binario Local (LBP) es una técnica que compara el valor de intensidad de un píxel central (umbral) con todos los píxeles que lo rodean (vecinos) y asigna un número binario que representa la textura de ese vecindario. Luego, se construye un histograma para analizar la distribución de texturas en la imagen. Esta técnica,

utilizada en la clasificación de texturas y reconocimiento facial, entre otras aplicaciones, fue introducida en 1994 por Timo Ojala, Matti Pietikäinen y David Harwood [13]. En 2005, Navneet Dalal y Bill Triggs utilizaron un método de histogramas de direcciones de gradientes (HOG) para la detección de peatones. Este método construye histogramas de direcciones de los gradientes dentro de celdas (subdivisiones de la imagen). Luego, forma bloques a partir de la unión de múltiples celdas; estos bloques son normalizados y concatenados para formar el vector de características, mejorando así la invariancia a la iluminación y el contraste [14]. En 2001, con la intención de reconocer rostros en tiempo real, Paul Viola y Michael Jones presentaron el método de características de haar. Este método utiliza características basadas en la suma de píxeles dentro de áreas rectangulares. Para calcular estas sumas de manera eficiente, introdujeron el concepto de imagen integral, una representación de la imagen que permite obtener rápidamente la suma de píxeles en cualquier rectángulo [15]. La técnica de Bolsa de Palabras Visuales (BoVW) fue introducida en 2004 por Gabriella Csurka y sus colaboradores [16]. Esta técnica extrae vectores de características de zonas específicas de una imagen y luego agrupa estos vectores utilizando técnicas de clustering para crear las "palabras visuales". Finalmente, cada imagen puede ser representada por un histograma de "palabras visuales".

2.2.4. Clasificación (enfoque tradicional)

Del mismo modo, existen varios algoritmos de clasificación, entre los cuales se destacan algunos por su relevancia histórica. Existe una colección de clasificadores basados en el teorema de Bayes, conocidos como clasificadores Naive Bayes. Los primeros desarrollos de estos clasificadores se originan en trabajos estadísticos a finales de la década de 1960. El principio detrás de estos clasificadores es que asumen independencia condicional entre las características. Esto significa que cada característica contribuye de manera independiente a la probabilidad de una clase específica, sin tener en cuenta las posibles relaciones o interdependencias entre ellas [**?**]. En 1967, Thomas Cover y Peter Hart introdujeron el algoritmo de k-vecinos más cercanos (k-NN). Este método etiqueta una determinada instancia en base a la mayoría de etiquetas que poseen sus k-vecinos más cercanos [17]. Otro tipo de algoritmos utilizados en clasificación son los árboles de decisión (DT).

Estos se basan en una estructura de árbol y funcionan como un flujo de preguntas. Cada pregunta representa un nodo de decisión que divide el conjunto de datos en subconjuntos más pequeños hasta llegar a los nodos terminales, donde se obtiene una decisión final (etiqueta). Entre los trabajos más influyentes se encuentran los algoritmos propuestos por John Ross Quinlan [18, 19]. En 2001, Leo Breiman introdujo los bosques aleatorios (RF), que son un conjunto de árboles de decisión entrenados con diferentes subconjuntos del conjunto de datos; la decisión final se toma en función de la decisión de muchos árboles [20]. Uno de los trabajos más importantes fue presentado en 1995 por Corinna Cortes y Vladimir Vapnik, máquinas de vectores de soporte (SVM), este algoritmo encuentra el hiperplano que separa las diferentes clases en el espacio de características de manera que maximiza la distancia entre las muestras más cercanas de cada clase. Es conocido por su eficacia en problemas de clasificación binaria [21].

2.2.5. Redes neuronales convolucionales

Esta sección explora los componentes básicos de CNNs, lo cuál es fundamental para entender su funcionamiento y decisiones tomadas en el diseño de arquitecturas expuestas más adelante.

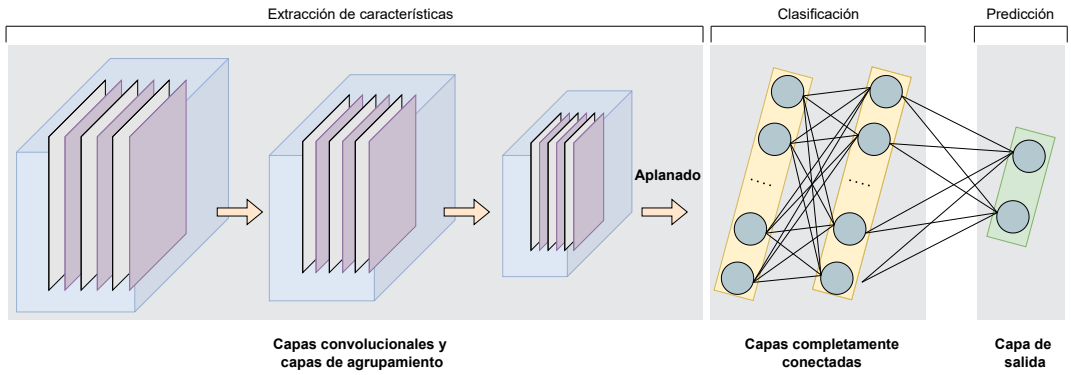


Figura 2.2.4: Red neuronal convolucional. Fuente: Adaptado de [1].

2.2.5.1. Red neuronal

Neurona

Las neuronas son la unidad elemental del sistema nervioso, estas reciben, procesan y

envían información hacia otras neuronas en forma de señales químicas o eléctricas. Inspiradas en el funcionamiento de las neuronas biológicas, se han creado las neuronas artificiales, también conocidas como perceptrones. Al igual que las neuronas biológicas, las neuronas artificiales procesan matemáticamente múltiples entradas para producir una respuesta que puede ser transmitida. La salida de una neurona se define por:

$$f\left(b + \sum_{i=1}^{n} (x_i \cdot w_i)\right) \tag{2.2.1}$$

Donde $f(.)$ es la función de activación (se detallará más adelante), x_i representa la señal de entrada, n es el número total de señales, w_i es el peso asignado a la entrada x_i, y b es el sesgo.

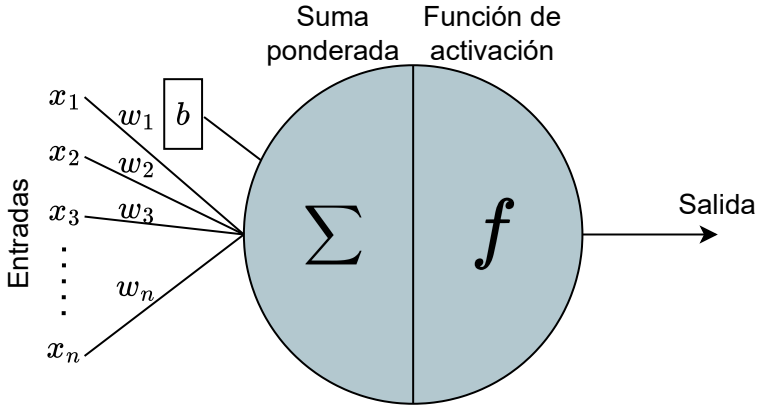


Figura 2.2.5: Neurona artificial. Fuente: Adaptado de [1]

Perceptrón multicapa (MLP)

MLP se constituye de una capa de entrada, una o más capas ocultas y una capa de salida. Cada neurona en una determinada capa, se encuentra conectada a todas las neuronas de la capa anterior y a todas las neuronas de la capa siguiente, por lo tanto, las salidas de las neuronas de una capa, son las entradas a las neuronas de la siguiente capa, permitiendo la transferencia de información entre capas a través de la red. Las capas anteriores en una red neuronal aprenden características generales de los datos, mientras que las capas posteriores se especializan en aprender características cada vez más específicas y detalladas [1, 2].

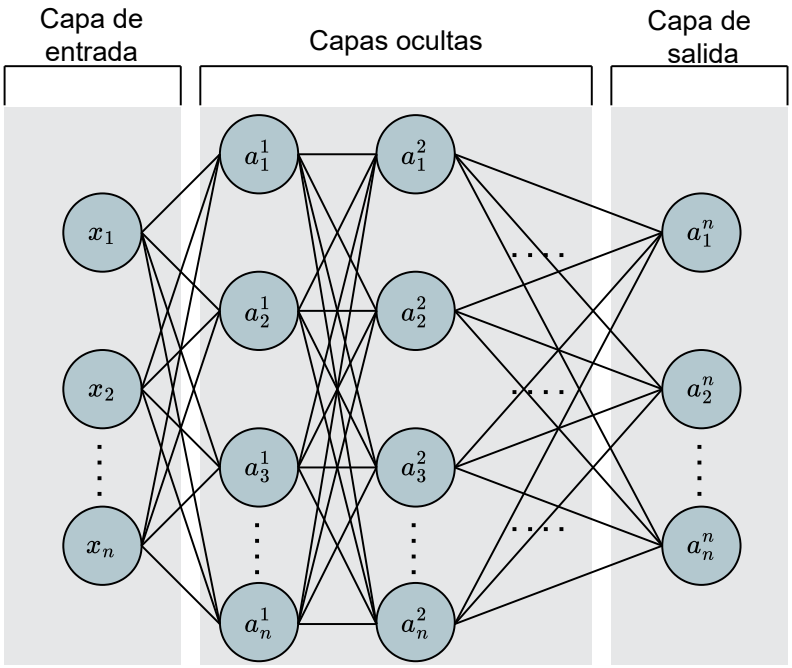


Figura 2.2.6: Red perceptrón multicapa. Fuente: Adaptado de [1]

El proceso de aprendizaje en un MLP se basa en dos fases principales: feedforward y backpropagation. En la fase de feedforward, la información se propaga desde la capa de entrada hasta la capa de salida, generando predicciones basadas en las características aprendidas. En la fase de backpropagation, se calcula el error entre las predicciones y las respuestas reales, y este error se utiliza para ajustar los pesos de la red mediante un algoritmo de optimización. La actualización para un determinado peso está dado por:

$$w_{\text{nuevo}} = w_{\text{antiguo}} - \alpha \cdot \frac{\partial E}{\partial w}, \tag{2.2.2}$$

donde w_{nuevo} es el valor actualizado del peso, w_{antiguo} es el valor del peso antes de la actualización, α es la tasa de aprendizaje y $\frac{\partial E}{\partial w}$ es el gradiente de la función error E con respecto al peso w [1].

2.2.5.2. Arquitectura de red neuronal convolucional (CNN)

Capa convolucional

Una capa convolucional es un componente fundamental en las CNNs. Esta capa está compuesta por varios filtros convolucionales, cada uno de los cuales contiene pesos, generalmente con dimensiones de 3×3 (9 pesos), 5×5 (25 pesos) o 7×7 (49 pesos). El número de canales de cada filtro debe ser igual al número de canales de la imagen de entrada. Estos filtros recorren la imagen en pequeñas regiones, píxel por píxel, extrayendo características relevantes y generando mapas de características. Las capas anteriores extraen características generales de las imágenes, como líneas, bordes y texturas. Por otro lado, las capas posteriores se enfocan en extraer información más específica y compleja, como formas y patrones detallados [1, 2].

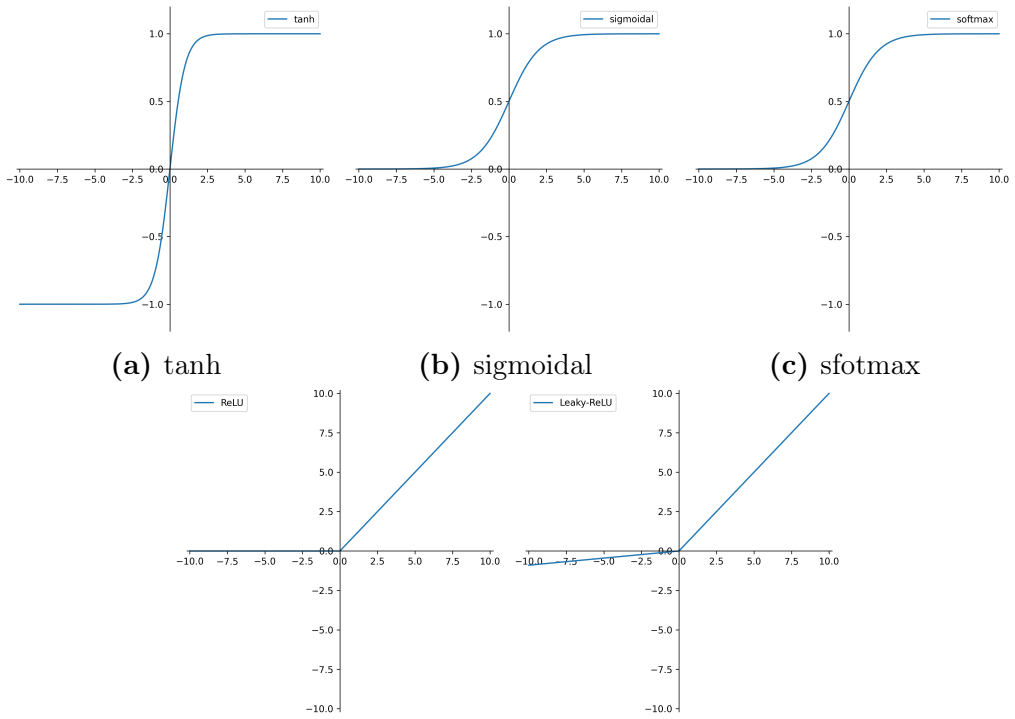
Capa de agrupamiento

Una capa de pooling es otro componente importante en las CNNs. Su función es reducir las dimensiones espaciales (ancho y alto) de los mapas de características, disminuyendo así la cantidad de parámetros y el costo computacional, mientras se retienen las características más relevantes. Existen varios tipos de pooling, siendo los más comunes el max pooling (agrupamiento máximo) y el average pooling (agrupamiento promedio) [1, 2].

Función de activación

La suma ponderada realizada por un perceptrón es una operación lineal que relaciona las entradas con la salida. Para abordar la necesidad de clasificar datos que no son linealmente separables, se introducen las funciones de activación, también conocidas como funciones de activación no lineales. Estas funciones añaden no linealidad a la red y mejoran su desempeño. Existen distintas funciones de activación, entre ellas se encuentran: *tanh*, que ajusta todos los valores entre -1 y 1; *sigmoidal*, que ajusta todos los valores a una probabilidad entre 0 y 1; *softmax*, una generalización de la función sigmoidal que se utiliza cuando hay dos o más clases; *ReLU*, que actúa como una función identidad para valores mayores a cero y es 0 para cualquier entrada menor o igual a cero; y *Leaky ReLU*, que a diferencia de ReLU introduce una pendiente aproximadamente de 0,01 para valores negativos. Debido a la rapidez de la función ReLU en el entrenamiento con descenso de gradiente, es la más utilizada en las capas ocultas de las redes

neuronales, mientras que para problemas multiclase se suele utilizar la función softmax en la capa de salida, ya que proporciona una distribución de probabilidad sobre las diferentes clases [1, 22, 3, 23, 24].



(a) tanh **(b)** sigmoidal **(c)** sfotmax

(d) ReLU **(e)** Leeaky ReLU

Figura 2.2.7: Funciones de activación. Fuente: Adaptado de [2].

Capa completamente conectada (FC)

Las capas completamente conectadas (FC), también conocidas como capas densas, se ubican después de las capas encargadas de la extracción de características. Estas capas están conformadas por neuronas que están completamente conectadas a todas las neuronas de la capa anterior y a todas las neuronas de la capa posterior. Las capas completamente conectadas utilizan las características extraídas por las capas anteriores para realizar la clasificación de los datos. En otras palabras, actúan como el clasificador de la red [1, 2].

Técnicas de regularización

Problemas como el sobreajuste y el subajuste están directamente relacionados con la complejidad de los datos de entrada en comparación con la complejidad de

la red neuronal. Para reducir el riesgo de sobreajuste y mejorar la capacidad de generalización del modelo, se utilizan técnicas de regularización. Dropout implica apagar aleatoriamente una fracción de las neuronas durante el entrenamiento, lo que evita que el modelo dependa demasiado de patrones específicos del conjunto de entrenamiento. La regularización L2 añade una penalización proporcional al cuadrado de los valores de los pesos a la función de pérdida, ayudando a controlar la magnitud de los pesos. Además, la normalización por lotes ajusta y escala las salidas de las neuronas en cada capa para mantener activaciones en un rango más estable, acelerando el entrenamiento y mejorando la estabilidad del modelo [25, 1, 2].

Función de Error

La función de error proporciona una medida del desempeño de la red basada en las predicciones generadas en la capa de salida. Entre las más utilizadas en problemas de clasificación se encuentran: el Error Cuadrático Medio (MSE), que evalúa la diferencia promedio entre las predicciones y los valores verdaderos, y la Entropía Cruzada, que mide la disimilitud entre la distribución de probabilidad predicha y la distribución real de las clases [1, 2].

Optimizador

Un mejor desempeño de la red está directamente relacionado con la reducción del error. Una vez definida la función de error, el objetivo es encontrar los pesos (es decir, los parámetros de la función de error) que minimicen dicho error, convirtiendo este proceso en un problema de optimización. El método de optimización más utilizado en redes neuronales es el descenso del gradiente. Este método busca minimizar la función de error actualizando los parámetros en la dirección opuesta al gradiente de la función. El tamaño de los pasos hacia el mínimo local está determinado por la tasa de aprendizaje α. Algunos algoritmos de descenso del gradiente son: Descenso del Gradiente Estocástico (SGD), Momentum, RMSprop, Adam y Adagrad [1, 2, 26, 27, 28].

2.2.5.3. Arquitecturas Clásicas

LeNet

Lecun y colaboradores propusieron la arquitectura LeNet-5 en 1998, cuyo objetivo era clasificar imágenes de caracteres escritos a mano **??**. Esta red cuenta con 5 capas de pesos (capas entrenables): 3 capas convolucionales (6 filtros de 5x5, 16 filtros de 5x5 y 120 filtros de 5x5), intercaladas con capas de agrupamiento promedio, y dos capas completamente conectadas (una con 84 neuronas y otra de salida con 10 neuronas). En las capas ocultas se utiliza la función de activación sigmoidal, mientras que en la capa de salida se emplea softmax.

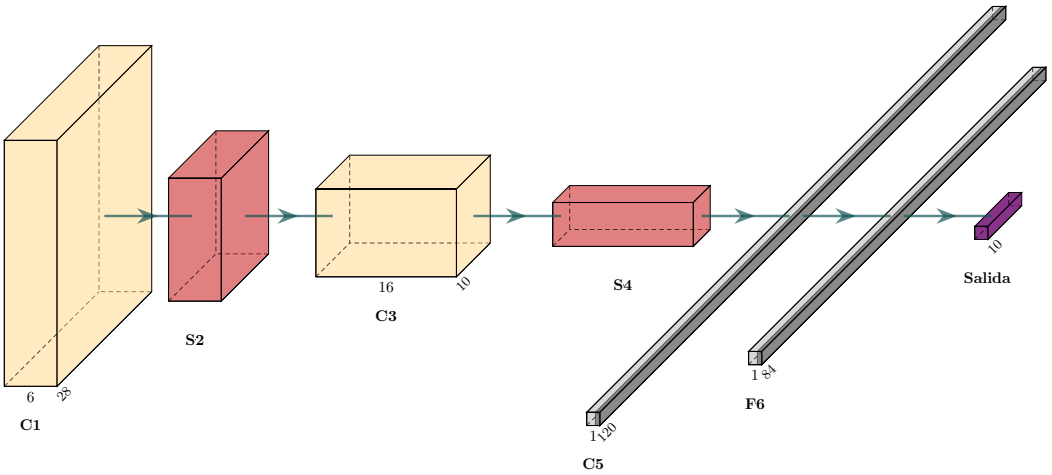


Figura 2.2.8: Arquitectura de LeNet. Fuente: Adaptado de [3, 1].

Cabe destacar que esta arquitectura fue pionera en su época y sirvió de inspiración para numerosos trabajos posteriores. A día de hoy con sus 61,706 parámetros se considera una red relativamente sencilla y menos efectiva para abordar problemas más complejos.

AlexNet

La principal motivación detrás de AlexNet fue desarrollar una red capaz de superar el desempeño en problemas más complejos [4]. Krizhevsky y sus colaboradores diseñaron una red entrenada con 1.2 millones de imágenes de alta resolución, provenientes de 1,000 clases del conjunto de datos ImageNet. La red ganó de manera destacada la competencia de clasificación de imágenes ILSVRC

en 2012. La arquitectura consiste en 8 capas de pesos: 5 capas convolucionales (96 filtros de 11x11, 256 filtros de 5x5, 384 filtros de 3x3, 384 filtros de 3x3 y 256 filtros de 3x3), seguidas de capas de agrupamiento máximo (max pooling) después de las primeras dos y la última capa del extractor de características. Además, incluye 3 capas completamente conectadas (las dos primeras con 4096 neuronas cada una y la capa de salida con 1000 neuronas). La función de activación utilizada en las capas ocultas es ReLU, mientras que para la capa de salida se emplea softmax. Para prevenir el sobreajuste, se aplicó dropout en las capas completamente conectadas de 4096 neuronas con una probabilidad de 0.5. AlexNet aumenta considerablemente el número de parámetros a 62 millones.

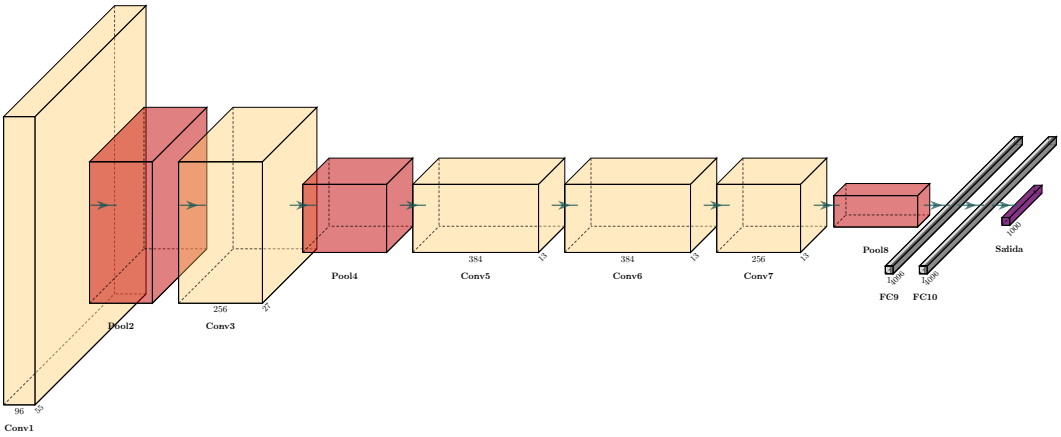


Figura 2.2.9: Arquitectura AlexNet. Fuente: Adaptado de [4, 1].

VGGNet

Karen Simonyan y Andrew Zisserman crearon en 2014 una arquitectura aún más profunda que AlexNet, la red VGGNet [5]. Esta arquitectura facilita algunas decisiones de diseño, como la elección del tamaño de los filtros en las capas convolucionales, utilizando pequeños filtros de 3×3 en cada capa. La idea detrás de esto es que múltiples pequeños filtros sucesivos añaden más profundidad y, por lo tanto, aumentan la capacidad de la red para aprender características más complejas.

VGGNet agrupa varias capas convolucionales idénticas que mantienen las mismas dimensiones de la entrada y luego agrega una capa de agrupamiento máximo que reduce las dimensiones a la mitad. Existen distintas configuraciones de

VGGNet que difieren en la organización de las capas convolucionales. Una configuración común es VGG16, que se conforma de un primer bloque con dos capas convolucionales de 64 filtros de 3×3, un segundo bloque con dos capas convolucionales de 128 filtros de 3×3, un tercer bloque con tres capas convolucionales de 256 filtros de 3×3, un cuarto bloque con tres capas convolucionales de 512 filtros de 3×3 y un quinto bloque con tres capas convolucionales de 512 filtros de 3×3. Luego de los bloques, hay tres capas completamente conectadas, iguales en todas las configuraciones: las dos primeras de 4096 neuronas cada una y la capa de salida de 1000 neuronas. En las capas ocultas se utiliza ReLU como función de activación y en la salida softmax. Dropout y regularización L2 se emplean para evitar el sobreajuste. Esta configuración posee 138 millones de parámetros.

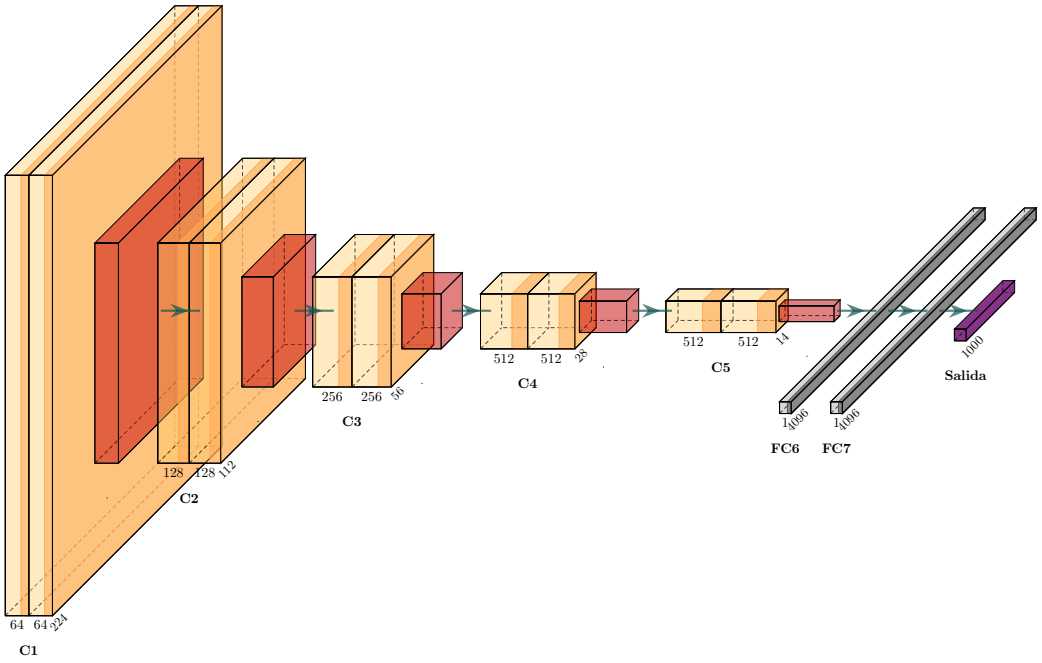


Figura 2.2.10: Arquitectura VGGNet16. Fuente: Adaptado de [5, 1].

Inception (GoogLeNet)

Inception fue presentada en 2014 por un grupo de investigadores de Google [6]. Esta arquitectura permite construir redes mucho más profundas con un menor número de parámetros (12 veces menos) en comparación con VGGNet e introduce

un componente clave llamado módulo Inception. Mientras que las arquitecturas previas incluían capas convolucionales con distintos tamaños de filtros, seguidas de capas de agrupamiento intercaladas, el módulo Inception agrupa estas decisiones en un solo módulo. De esta forma, la salida de un módulo inception está dada por la concatenación de la salida de 4 ramas: una con una capa convolucional de 1×1; otra con una capa convolucional de 3×3; una tercera con una capa convolucional de 5×5; y una cuarta que aplica una capa de agrupamiento máximo de 3×3. Además, dado el costo computacional que agregan los filtros de mayor tamaño (como 5×5), se añaden capas de reducción de dimensiones en cada una de las ramas, es decir, capas convolucionales de 1×1 con un bajo número de filtros.

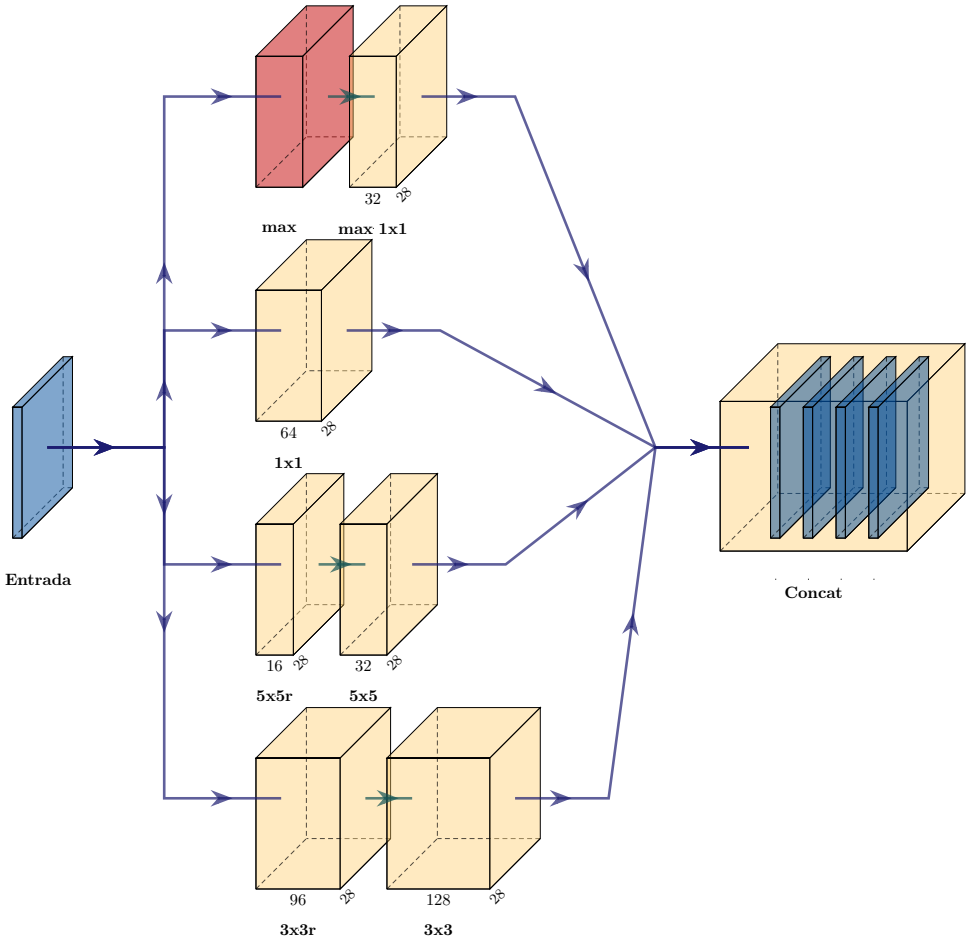


Figura 2.2.11: Módulo Inception. Fuente: Adaptado de [6, 1].

Los creadores participaron en la competencia ILSVRC en 2014 con una versión de Inception llamada GoogLeNet. Esta arquitectura se compone de tres partes principales. La parte A comienza con una capa convolucional de 64 filtros de

7×7, seguida de una capa de agrupamiento máximo de 3×3. Luego, se aplica una capa convolucional con 64 filtros de 1×1 (para reducción de dimensiones), seguida de una capa convolucional con 192 filtros de 3×3, y finaliza con una capa de agrupamiento máximo de 3×3. La parte B está formada por nueve módulos Inception organizados en tres bloques principales, con 2, 5 y 2 módulos Inception, respectivamente. Cada bloque está seguido por una capa de agrupamiento: los primeros dos bloques por una capa de agrupamiento máximo de 3×3 y el último bloque por una capa de agrupamiento promedio de 7×7. Finalmente, la parte C de la arquitectura incluye una capa completamente conectada de 1000 neuronas con dropout, seguida por la capa de salida con 1000 neuronas. En las capas ocultas se utiliza la función de activación ReLU, mientras que en la capa de salida se emplea softmax para la clasificación.

Red Neuronal Residual (ResNet)

La red neuronal residual fue desarrollada en 2015 por un equipo de Microsoft Research [29]. El diseño de redes muy profundas ofrece una mayor capacidad de aprendizaje, pero también presenta dos problemas principales: el sobreajuste, que ResNet mitiga mediante un extenso uso de normalización por lotes, y el desvanecimiento del gradiente, que ocurre cuando el gradiente utilizado para actualizar los pesos se vuelve muy pequeño, impidiendo el aprendizaje efectivo en las capas anteriores. Para abordar este problema, ResNet incorpora un módulo denominado módulo residual con una conexión de salto. Esta red logró resultados destacados en la competencia ILSVRC 2015.

Un módulo residual se compone de dos ramas: una de ellas incluye tres capas convolucionales con filtros de 1×1, 3×3 y 1×1, respectivamente. Cada una de estas capas utiliza normalización por lotes y la función de activación ReLU. La otra rama es una conexión de salto que suma directamente la entrada del módulo a la salida de la primera rama antes de aplicar la función de activación en la última capa convolucional. Cuando las dimensiones de la entrada y la salida de la rama convolucional no coinciden, se utiliza una conexión de salto con reducción, lograda mediante una capa convolucional de 1×1 en la rama de la conexión de salto, que ajusta las dimensiones de la entrada para que coincidan con las de la salida del bloque residual, permitiendo así la suma entre la entrada y la salida.

Al igual que VGGNet, ResNet tiene distintas configuraciones que varían en la profundidad: ResNet-18, ResNet-34, ResNet-50, ResNet-101 y ResNet-152. En el caso de ResNet-50, esta tiene una primera parte que se compone de una capa convolucional con 64 filtros de 7×7 con función de activación ReLU, seguida de una capa de agrupamiento máximo de 3×3. Luego, la red se divide en cuatro bloques de módulos residuales. El primer bloque contiene tres módulos con capas convolucionales de 64, 64, y 256 filtros respectivamente; el segundo bloque contiene cuatro módulos con capas convolucionales de 128, 128, y 512 filtros respectivamente; el tercer bloque incluye seis módulos con capas convolucionales de 256, 256, y 1024 filtros respectivamente; y el cuarto bloque tiene tres módulos con capas convolucionales de 512, 512, y 2048 filtros respectivamente. Finalmente, la red incluye una capa de agrupamiento promedio y una capa completamente conectada de salida con 1000 neuronas y función de activación softmax.

2.2.6. Transformadores visuales (ViT)

Los transformadores visuales fueron introducidos en 2021 por Alexey Dosovitskiy y sus colaboradores [30]. Estos modelos, inspirados en la exitosa investigación de Ashish Vaswani y su equipo en 2017 [31], que se centró en el procesamiento del lenguaje natural (NLP), aprovechan un mecanismo de atención para capturar relaciones entre parches dentro de las imágenes.

2.2.6.1. Arquitectura de transformador visual

Esta sección describe las partes y procesos esenciales que conforman un ViT.

Embebido de parches

En una etapa inicial, las imágenes son subdivididas en parches de igual dimensión. Si las dimensiones de una imagen son (H, W, C), donde H es la altura, W es el ancho, y C es el número de canales, y las dimensiones de cada parche son (P, P), entonces el número de parches (subdivisiones de la imagen) está dado por $N = H \times W/P^2$. Luego, cada parche es aplanado (flattened) para formar un vector de características de dimensiones $(1, P^2 \times C)$, llamado token. Estos tokens se ajustan mediante una proyección lineal entrenable a una dimensión D, de modo que cada token tiene dimensiones $(1, D)$. Junto a los token, es concatenado un

token de predicción, este token obtiene información del resto de tokens y es utilizado para realizar la predicción final. Además, se añade (suma) información relativa a la posición de los tokens dentro de las imágenes. Finalmente, una secuencia de tokens es pasada hacia el transformador [30, 31, 32, 33, 34].

Atención

Es el mecanismo que busca relaciones entre las distintas subdivisiones (parches) de las imágenes. A través de este proceso, el modelo aprende a comprender el contexto global de la imagen evaluando la importancia relativa de cada parche en función de los demás parches presentes. El cálculo de la atención se define por la siguiente fórmula:

$$attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{2.2.3}$$

donde Q, K y V son las proyecciones lineales de las consultas, claves y valores, respectivamente, y d_k es la dimensión de las claves.

Además, *atención múltiple* permite ejecutar k operaciones de atención en paralelo denominadas *cabezas*, donde cada cabeza utiliza proyecciones lineales independientes, generalmente con dimensiones D/k. Las salidas de todas las cabezas se concatenan y proyectan a la dimensión original D. De esta manera, el modelo puede captar aspectos diferentes en las relaciones de las entradas. Esta operación es realizada por la *capa de atención múltiple* [35, 30, 31, 32, 33, 34].

Codificador

Un codificador, es una capa que se compone de dos subcapas: atención múltiple y perceptrón multicapa. A la salida de ambas subcapas, se añade normalización por lotes y una conexión de salto (conexión residual). Un transformador visual incluye varios codificadores [30, 31, 32, 33, 34].

Predicción

Una vez que una determinada entrada pasa a través todas las capas de codificación, el token de predicción es separado de los demás token y pasado a un clasificador compuesto de una o más capas de densas. Este se encarga de generar la predicción final del modelo [30, 31, 32, 33, 34].

Capítulo 3

Materiales y métodos

En este capitulo se exponen los conjuntos de datos utilizados, así como aspectos claves de algunos trabajos previos que realizaron clasificación con ellos.

3.1. Conjuntos de datos y preprocesamiento

Como conjunto de datos se utilizaron colecciones de imágenes pertenecientes a distintas clases, cada uno de estos conjuntos se enfoca en resolver problemas de áreas particulares, siendo cada área diferente de las que abordan los otros conjuntos.

3.1.1. WM-811K

Este conjunto de datos contiene mapas de *wafers*, es decir, imágenes que representan el estado de los discos de silicio utilizados en la fabricación de semiconductores. En total posee $811,457$ imágenes, los datos etiquetados pertenecen a 9 clases. Con la intención de aumentar el porcentaje de chips que son fabricados con éxito en un disco de silicio, es fundamental identificar los tipos de fallas que ocurren en producción, Acorde al análisis exploratorio de los datos, $25,519$ imágenes se distribuyen entres las 8 clases que corresponden a tipos de fallas, $147,431$ son de la clase reservada para discos sin fallas (*none*) y $638,507$ no tienen etiqueta.

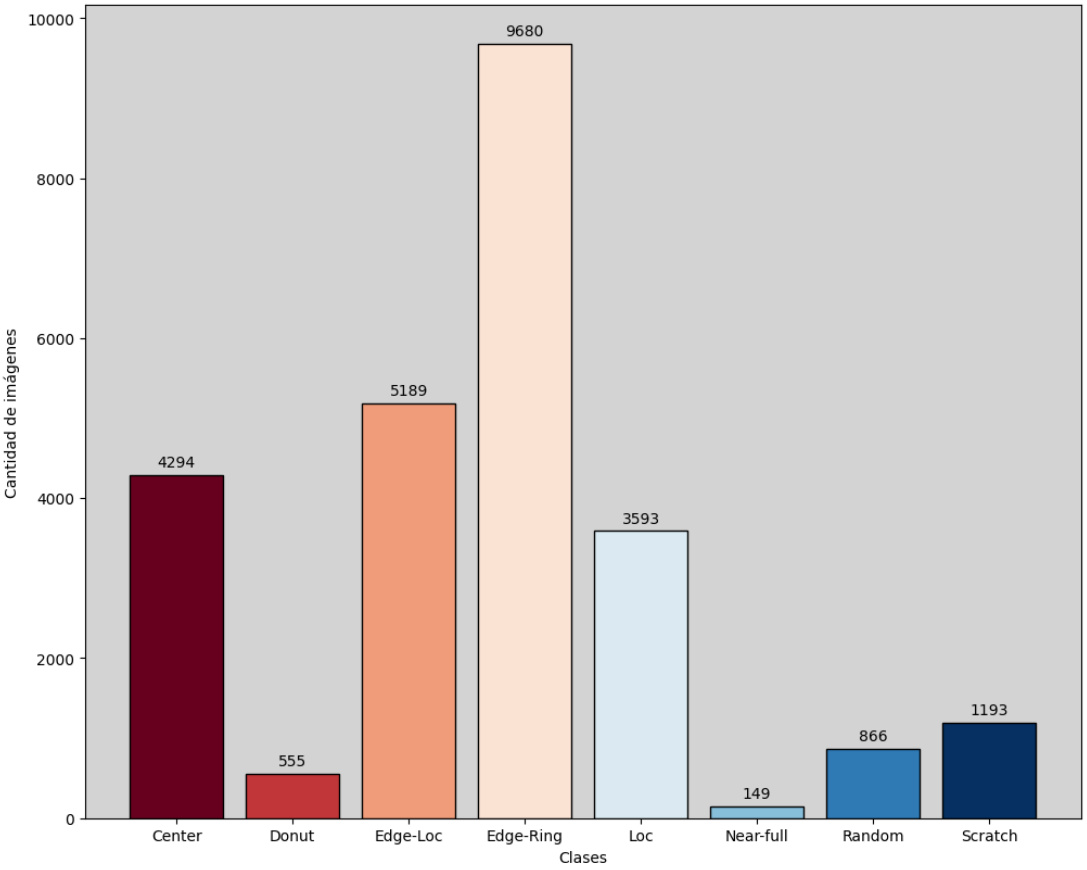


Figura 3.1.1: Distribución de clases de WM-811K. Fuente: Elaboración propia.

Las imágenes fueron preprocesadas, igualando las dimensiones de todas ellas. El tamaño escogido del redimensionamiento se basó en la media aritmética de las dimensiones en x e y de las imágenes etiquetadas en algún tipo de falla, similar a lo realizado en [**?**], donde emplearon la media ponderada. Todas las imágenes tienen un solo canal.

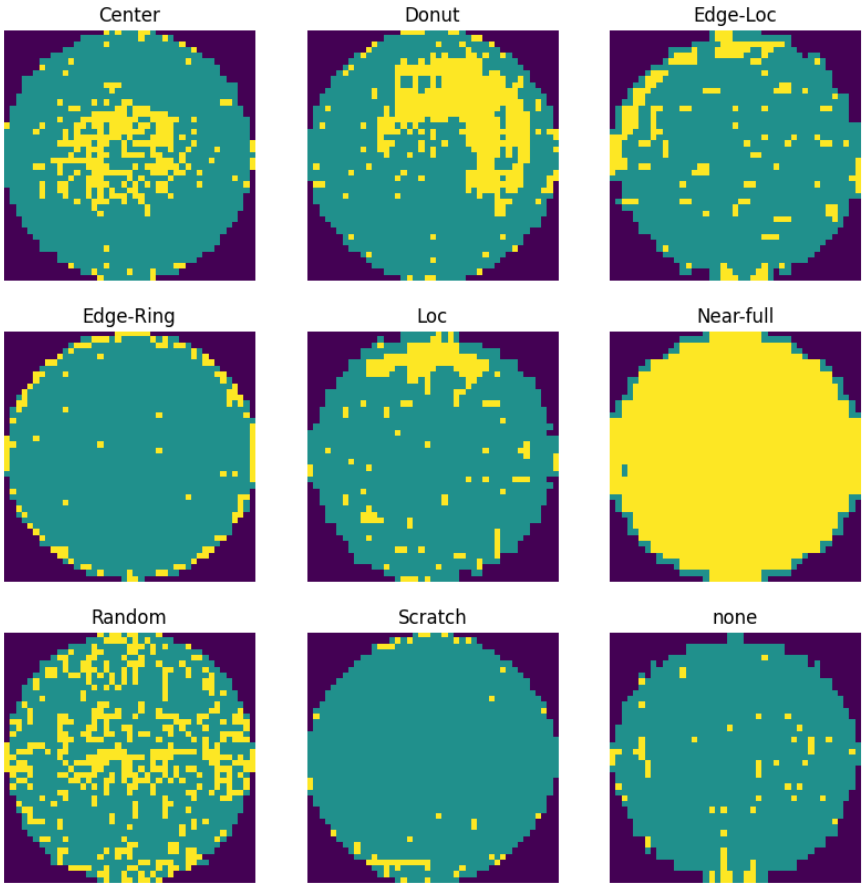


Figura 3.1.2: Clases de WM-811K. Fuente: Elaboración propia.

3.1.2. Brain Tumor Classification (MRI)

Este conjunto de datos reúne imágenes de resonancias magnéticas, que se ordenan en clases de tumores cerebrales: pituitario, glioma, meningioma y no tumor. Contiene un total de $3,264$ imágenes distribuidas como muestra la Figura **??**.

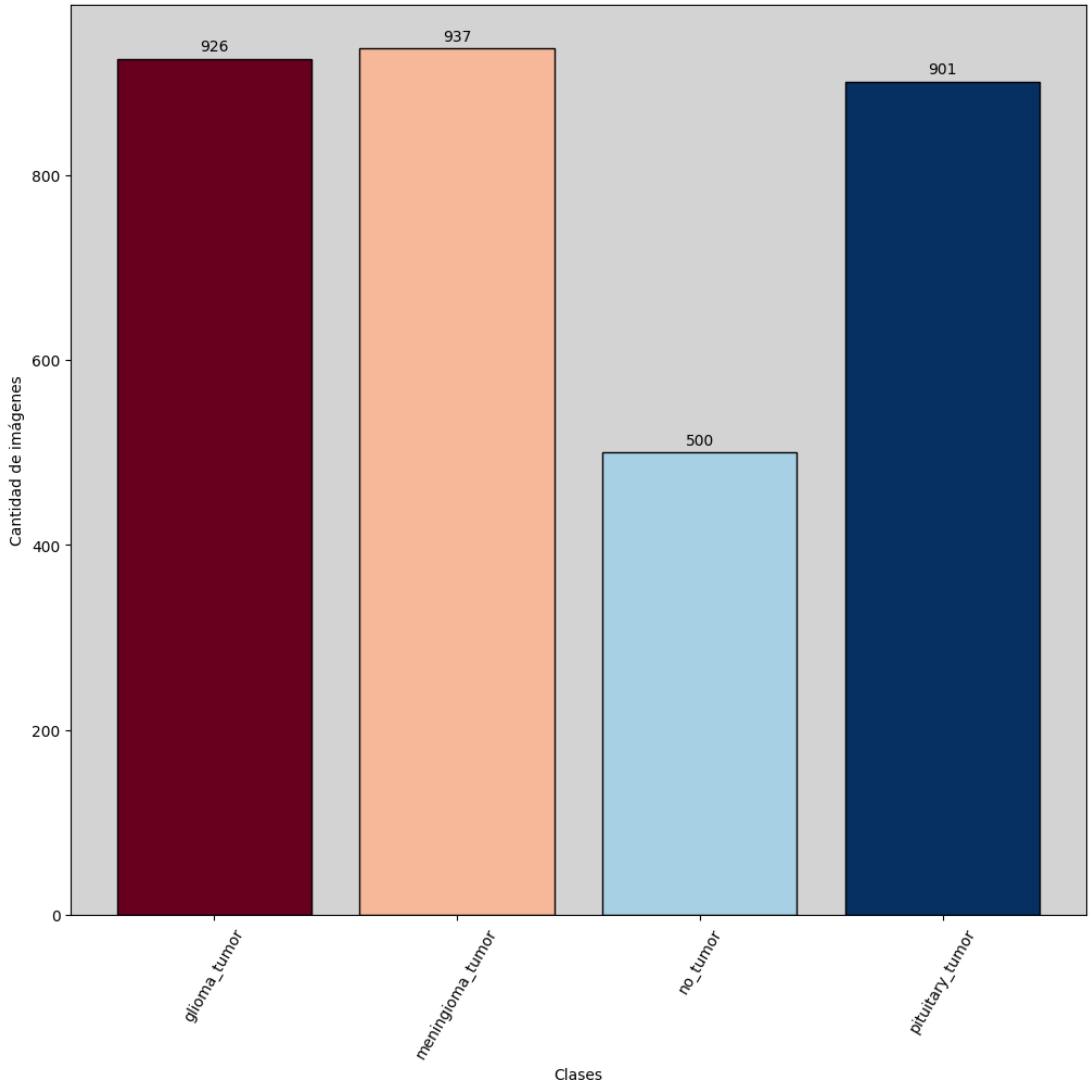
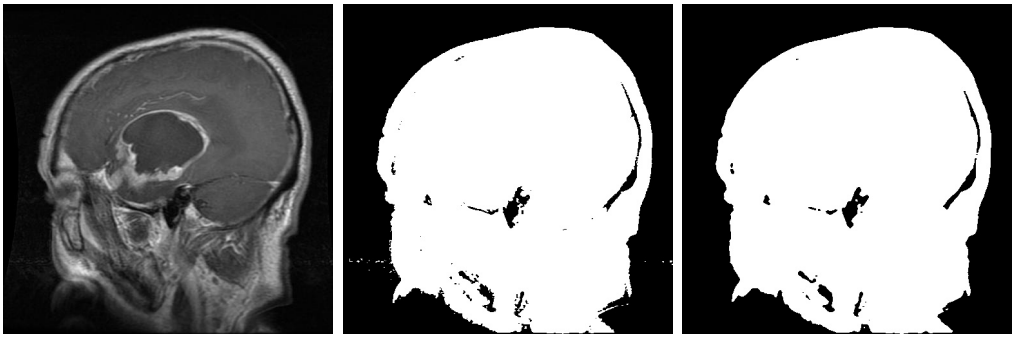
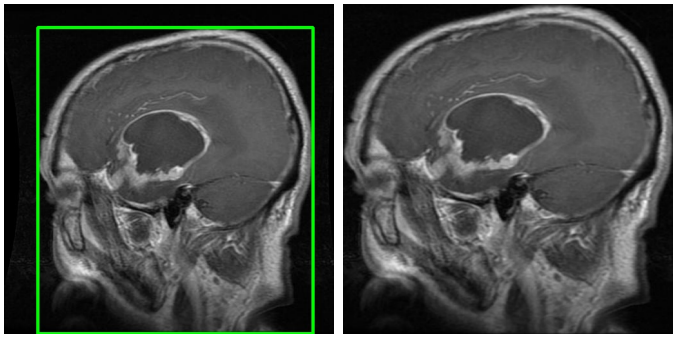


Figura 3.1.3: Distribución de clases de Brain Tumor. Fuente: Elaboración propia.

Originalmente, las imágenes se componen de 3 canales, pero como los colores no aportan información relevante para el problema en cuestión, se transforman a escala de grises (lucen igual). Además, siguiendo la metodología de otro trabajo que utiliza el mismo conjunto de datos, donde se realiza un recorte para incluir en la imagen exclusivamente la zona de interés (quitar márgenes) [**?**], se realiza un recorte basado en un procedimiento que a diferencia del trabajo mencionado, evita la pérdida de información (imágenes inutilizables) que puede ocurrir debido a un mal cálculo del contorno. Luego, se realiza un redimensionamiento para que todas tengan el mismo tamaño.



(a) Original **(b)** Umbralizada **(c)** Filtrada



(d) Contorno **(e)** Recortada

Figura 3.1.4: Procedimiento para recortar una imagen. Fuente: Elaboración propia.

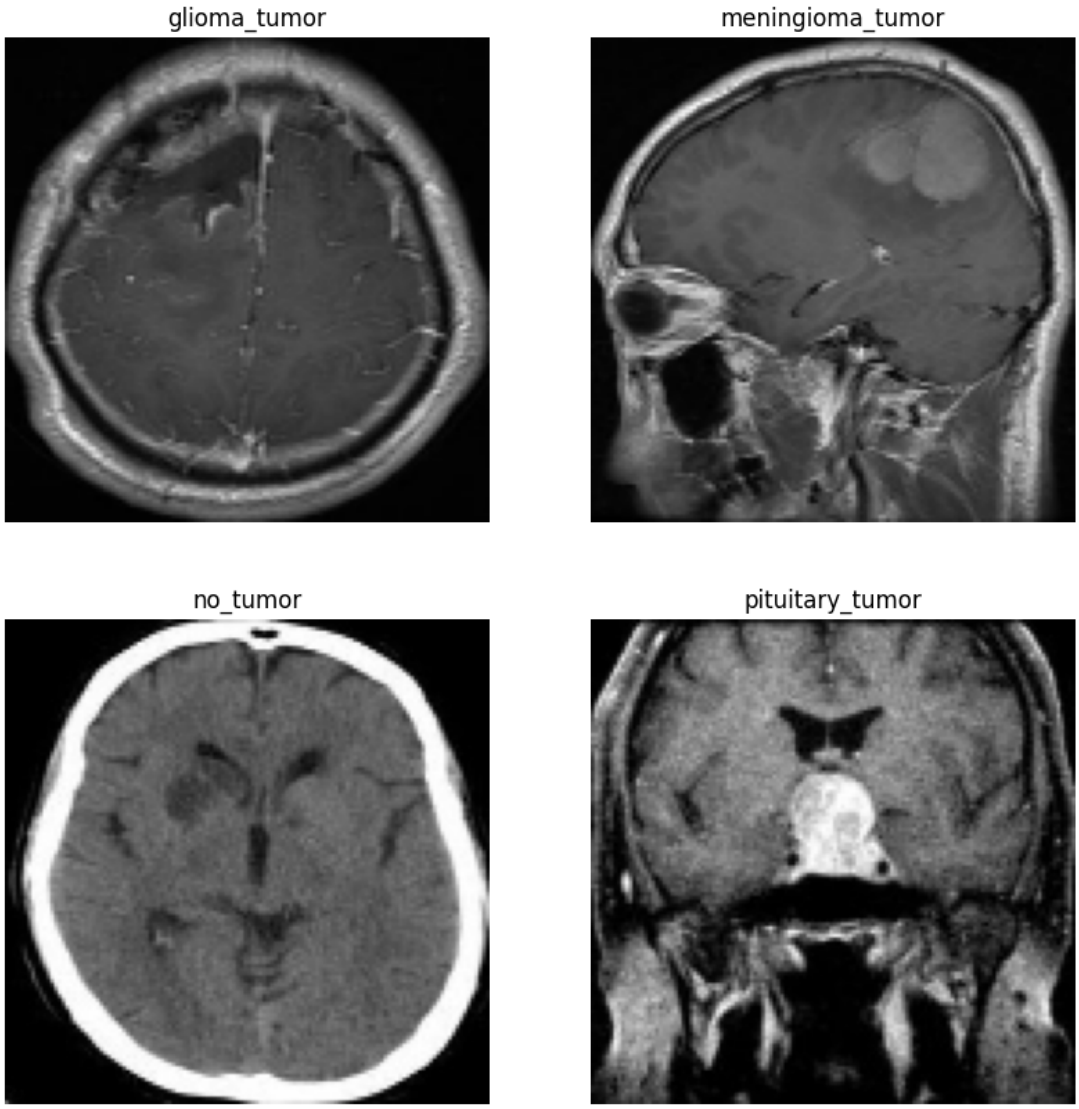


Figura 3.1.5: Clases de Brain Tumor. Fuente: Elaboración propia.

Capítulo 4

Conclusiones y trabajo futuro

Bibliografía

[1] M. Elgendy, O. for Higher Education (Firm), and an O'Reilly Media Company. Safari, *Deep Learning for Vision Systems*.

[2] L. Chen, S. Li, Q. Bai, J. Yang, S. Jiang, and Y. Miao, "Review of image classification algorithms based on convolutional neural networks," *Remote Sensing 2021, Vol. 13, Page 4712*, vol. 13, p. 4712, 11 2021. [Online]. Available: https://www.mdpi.com/2072-4292/13/22/4712/htmhttps://www.mdpi.com/2072-4292/13/22/4712

[3] Y. Lecun, L. E. Bottou, Y. Bengio, and P. H. Abstract|, "Gradient-based learning applied to document recognition," 1998.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84–90, 6 2017.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 9 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[7] G. Boesch, "What is computer vision? the complete tech guide for 2024 - viso.ai," 1 2023. [Online]. Available: https://viso.ai/computer-vision/what-is-computer-vision/

[8] IBM, "What is computer vision? | ibm." [Online]. Available: https://www.ibm.com/topics/computer-vision

[9] R. C. Gonzalez and R. E. R. E. Woods, *Digital image processing*, 4th ed.

[10] G. Kumar and P. K. Bhatia, "A detailed review of feature extraction in image processing systems," *International Conference on Advanced Computing and Communication Technologies, ACCT*, pp. 5–12, 2014.

[11] D. G. Lowe, "Object recognition from local scale-invariant features," 1999.

[12] H. Bay, T. Tuytelaars, and L. V. Gool, "Lncs 3951 - surf: Speeded up robust features," 2006.

[13] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *Proceedings of 12th International Conference on Pattern Recognition*. IEEE Comput. Soc. Press, 1994, pp. 582–585.

[14] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. IEEE, 2005, pp. 886–893.

[15] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001.

[16] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints."

[17] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, pp. 21–27, 1 1967.

[18] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, pp. 81–106, 1986.

[19] by J Ross Quinlan, M. K. Publishers, and S. L. Salzberg, "Programs for machine learning," vol. 16, pp. 235–240, 1994.

[20] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[21] C. Cortes, "Support-vector networks," pp. 273–297, 1995.

[22] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "Activation functions in deep learning: A comprehensive survey and benchmark," *Neurocomputing*, vol. 503, pp. 92–108, 9 2022.

[23] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," 2011.

[24] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," 2013.

[25] Y. Tian and Y. Zhang, "A comprehensive survey on regularization strategies in machine learning," *Information Fusion*, vol. 80, pp. 146–166, 2022. [Online]. Available: https://doi.org/10.1016/j.inffus.2021.11.005

[26] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini--batch gradient descent," 2012.

[27] J. D. JDUCHI and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization * elad hazan," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[28] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization."

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 12 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[30] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *ICLR 2021 - 9th International Conference on Learning Representation*, 10 2021. [Online]. Available: https://arxiv.org/abs/2010.11929v2

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Łukasz Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 2017-December, pp. 5999–6009, 6 2017. [Online]. Available: https://arxiv.org/abs/1706.03762v7

[32] G. Boesch, "Vision transformers (vit) in image recognition: Full guide - viso.ai," 11 2023. [Online]. Available: https://viso.ai/deep-learning/vision-transformer-vit/

[33] S. J. Callis, "Vision transformers, explained. a full walk-through of vision... | by skylar jean callis | towards data science," 1 2024. [Online]. Available: https://towardsdatascience.com/vision-transformers-explained-a9d07147e4c8

[34] H. Hettiarachchi, "what is vision transformers | medium," 2023. [Online]. Available: https://medium.com/@hansahettiarachchi/unveiling-vision-transformers-revolutionizing-computer-vision-beyond-convolution-c410110ef061

[35] S. J. Callis, "Attention for vision transformers, explained | by skylar jean callis | towards data science," 2 2024. [Online]. Available: https://towardsdatascience.com/attention-for-vision-transformers-explained-70f83984c673