UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ciencias

Comparación de algoritmos de machine learning para la detección y clasificación de tumores cerebrales mediante imágenes por resonancia magnética

Presenta:
Alberto Olvera Trejo

Diciembre 2022

1. Pregunta de investigación

La pregunta que se quiere responder por medio de este trabajo es la siguiente:

¿Cual es el mejor algoritmo de machine learning para detectar y clasificar tumores cerebrales en imágenes por resonancia magnética?

2. Objetivos

El objetivo principal es determinar cuál es el mejor algoritmo de redes neuronales convolucionales (CNN) para clasificar tumores cerebrales, usando como métrica el *recall*.

Como objetivo secundario se tiene que con la CNN obtenida se obtenga como salida la imagen de resonancia magnética (IRM) y dentro de ella un rectángulo en donde se encuentra el tumo para de esa manera poder identificarlo de mejor manera

3. Antecedentes

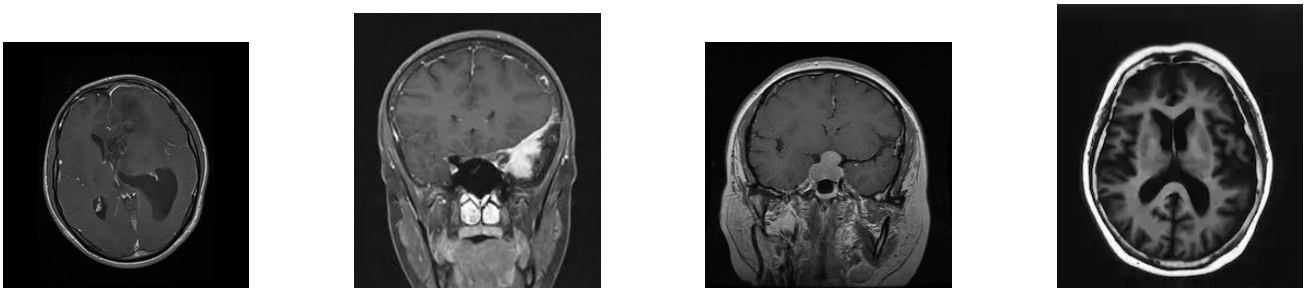
Un tumor cerebral es un conjunto de células las cuales se han multiplicado de manera descontrolada, formando una masa anormal en cualquier parte del cerebro. Pueden cau sar múltiples daños como por ejemplo la pérdida de la memoria, convoluciones, cambio de personalidad, problemas al hablar, entre otros [1]. Debido a su naturaleza y a su ubicación, los tumores cerebrales representan un riesgo tanto para la detección como para el diagnóstico, el cual debe de ser lo más temprano posible. Tomando en cuenta lo anterior y el hecho de que al año se diagnostican 300 mil casos de tumores cerebrales en el mundo [2], se deben de crear nuevas técnicas de detección y clasificación que tengan una precisión alta y una fácil implementación.

Actualmente una de las técnicas más usadas (la cual es considerada como la mejor) para la detección de tumores cerebrales es la imagen por resonancia magnética (IRM) ya que si hay un tumor encefálico, dicha imagen casi siempre lo va a mostrar, además de que también sirven para poder darse una idea de qué tipo de tumor es [3].

Una vez detectado el tumo se puede clasificar como benigno o maligno, aunque también se puede clasificar como primario o secundario. Los primarios son aquellos que se originan en las células dentro del cerebro (o las que están próximas a él) y pueden ser tanto benignos como malignos. Los secundarios son metástasis, es decir que se originaron en otra parte del cuerpo y se diseminaron al cerebro [4].

Los tumores primarios más frecuentes son gliomas, los cuales se cree que crecen en las células gliales o

las precursoras gliales y pueden ser o no agresivo, y los meningiomas que son aquellos que comienza en las meninges y la mayoría de las veces no es canceroso, aunque puede causar síntomas graves [5]. Otro tipo de tumor son los pituitarios los cuales son en su mayoría benignos y se originan en la glándula pituitaria que está ubicada en la base del cerebro, detrás de la nariz. En la figura 1 se muestra una IRM de cada tipo de tumor y una extra de cuando no hay tumor.



(a) Glioma (b) Meningioma (c) Pituitario (d) No tumor

Figura 1: IRM de los tres tipos de tumor y no tumor [6]

El machine learning es un subárea de la inteligencia artificial, el cual es un campo de la ciencia que tiene como objetivo hacer que las computadoras puedan razonar, pensar y crear como lo haría un ser humano, todo ello con una gran cantidad de datos que sirven como experiencia [7]. Dentro del machine learning está el subconjunto del deep learning, el cual consiste en una serie de algoritmos inspirados en las redes neuronales [8]. Dichos algoritmos pueden aprender patrones muy complejos siempre y cuando se tenga una gran cantidad de datos. Debido a su gran capacidad para resolver problemas, las redes neuronales están siendo aplicadas en todas las industrias, en particular en la medicina.

Dentro de los campos más importantes de la inteligencia artificial está la visión por computadora o visión artificial, el cual tiene como objetivo que una computadora obtenga información relevante de imágenes digitales, vídeos y algunas otras entradas visuales [9]. Para poder llevar a cabo dicha tarea se han creado diferentes algoritmos de entre los cuales destacan las Redes Neuronales Convolucionales (CNN) que son inspirados en la forma en como el cerebro procesa las imágenes. Se llaman convolucionales debido a que emplean las *convoluciones* matemáticas que sirven de manera similar a que si se le aplicara un filtro a alguna imagen (por ejemplo el filtro de blanco y negro, el de blurr, resaltar orillas, etc.). Las diferentes arquitecturas de las CNN han permitido resolver tareas sumamente complejas como por ejemplo la clasificación de imágenes, reconocimiento de arritmias peligrosas, detección de rostros, de códigos postales, entre otros. [10],[11]. En particular han servido demasiado para el área de la salud como por ejemplo:

- Descubrimiento de fármacos. Una red neuronal puede sacar a la luz asociaciones de moléculas previamente desconocidas y así predecir sus propiedades en fármacos [12]
- Detectar signos prematuros de un posible cáncer de piel mediante la aplicación para smartphones SkinVision [12]
- Medicina personalizada para, por ejemplo, saber qué fármacos son los mejores para un determinado paciente [12]
- Chatbots para salud mental [13]

En la imagen 2 se observa como es que una CNN procesa todos los datos, por ejemplo en la primer capa aprende a reconocer las orillas, los lugares en donde hay lineas verticales, horizontales, con cierta pendiente, etc., mientras que en la segunda capa junta los datos de la primera y empieza a reconocer patrones como lo son los ojos, labios, nariz, entre otros; y finalmente en la tercer capa junta dichos patrones para poder identificar rostros humanos.

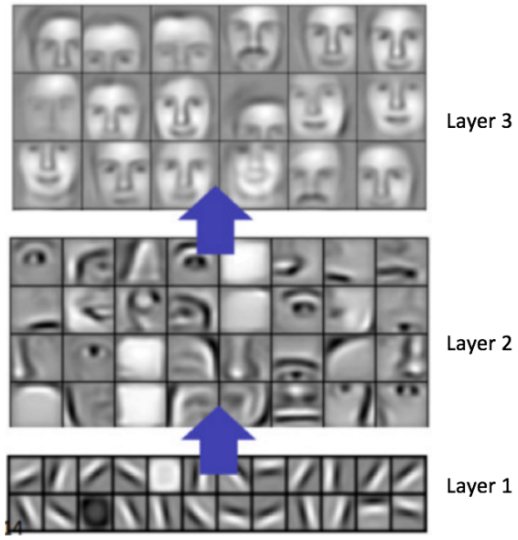


Figura 2: Visualización de filtros de una CNN [14]

Una técnica muy usada para el reconocimiento de imágenes es el aprendizaje por transferencia, el cual consiste en tomar una CNN ya entrenada para alguna otra tarea (por ejemplo clasificar diferentes razas perros) y en lugar de entrenar toda la red únicamente se modifican las últimas capas (aquellas que intervienen meramente en la clasificación). Se ha demostrado que dicha técnica tiene buenos resultados cuando el conjunto de datos de entrenamiento es muy chico (al rededor de 100 o 200 imágenes), además de que permite ahorrar tiempo de creación y entrenamiento de nuevos modelos [15] [16]. Cuando se trata de muestras grandes se ha visto que las redes usadas en aprendizaje por transferencia tienden a estar sesgadas, por lo que su rendimiento es peor que cuando se tienen pocos datos [17] [18].

Hay algunas CNN que son famosas ya sea porque alguna empresa de gran prestigio las entreno con todo su poder de computo y después las ponen a disposición de todos para que las usen. Algunos ejemplo de dichas arquitecturas se muestran en la tabla 1

4. Trabajos relacionados

Actualmente ya existen programas para hacer clasificación y detección de tumores cerebrales pero cada uno de ellos tiene sus desventajas, las cuales se muestran en la tabla 3

Nombre	Características	Logros
AlexNet	Tiene capas convolucionales consecutivas	Ganó la competencia ImageNet en 2012 con un error del 17 %
GoogLeNet	Logró reducir el número de parámetros (10 veces menos que los de AlexNet) usando el *principio de dispersión* de la fisiología Mejoro el accuracy de la clasificación de tumores	Ganó el reto ILSVRC en 2014 con un error por debajo del 7. Fue desarrollada por Google % Dataset pequeño
VGGNet	Es un modelo simple que intercala 2 o 3 capas convolucionales con una capa de pooling hasta llegar a las 16 o 19 capas convolucionales	Fue finalista en la competencia ILSVRC en 2014 y fue desarrollada en la universidad de Oxford
ResNet	Red muy profunda (152 capas) pero con menos parámetros entre una capa y la siguiente y evitando que todas las neuronas estén completamente conectadas con las de la siguiente capa	Ganó el reto ILSVRC en 2015 con un error por debajo del 3.6 %

Tabla 1: Arquitecturas de CNN usadas para aprendizaje por transferencia [19]

5. Justificación

Al reunir los resultados y al replicar los algoritmos más importantes se podrán identificar los componentes y las características que son más relevantes al querer hacer un programa como el ya mencionado y de esa manera poder crear un algoritmo que contenga las ventajas del resto.

6. Materiales y modelos

6.1. Dataset

El conjunto de las 7023 imágenes con las cual se va a trabajar fue obtenido de la página Brain Tumor MRI Dataset. Dichas imágenes contienen IRM sin tumor, con glioma, meningioma y con tumores pituitarios; para cada paciente se tienen 3 imágenes las cuales corresponden a los cortes axial, coronal y sagital. Se hizo una partición de las imágenes en tres conjuntos los cuales son los siguiente:

- Entrenamiento. - Es el conjunto con el cual se van a ir modificando los pesos de la red neuronal para que vaya mejorando la clasificación
- Validación.- Se usa para modificar los hiperparámetros (como por ejemplo la tasa de aprendizaje, numero de neuronas, numero de capas, etc.).

Referencia	Técnica	Accuray	Ventajas	Desventajas
[20]	Red neuronal VGG-16 con 16 capas	98 %	Mejoro el accuracy de la clasificación de tumores	Dataset pequeño
[21]	Red neuronal convolucional (CNN) para 3 clases	98 %	Si hay pocos datos, es mejor usar aprendizaje por transferencia	Los casos de meningioma fueron clasificados incorrectamente y hubo overfitting
[22]	SVM y k vecinos (k-NN)	97.25 %	Se extendió la región en donde se puede localizar el tumor	El accuracy es bajo
[23]	CNN para multi clasificación	99.33 %	Los doctores se apoyan en dicho algoritmo para validar la primera evaluación	-
[23]	Clasificador multi clase	90.27 %	Tiempo de ejecución corto. Ayuda a los doctores a hacer una mejor decisión	Se necesita incrementar el accuracy
[24]	CNN de 22 capas	96.56 %	Es un modelo relativamente simple a comparación de otros	El conjunto de datos era pequeño y el accuracy es bajo
[24]	CNN simple para multi clasificación	84.10 %	Es el modelo más simple para una CNN	El accuracy es el más bajo
[25]	Modificación de ResNet-50	97.2 %	Se usó aprendizaje por transferencia, por lo que el tiempo de entrenamiento es menor	-

Tabla 2: Tabla de algoritmos actuales

- Test. - Cuando se están comparando varios modelos y se han terminado de entrenar, se usa el conjunto de test para medir su rendimiento, pues son ndatos que nunca antes se habían visto y es una manera de calificar que tan bien generalizan los datos.

La distribución del número de imágenes en cada conjunto es la siguiente:

Tipo de tumor	Total de imágenes	Entrenamiento	Validación	Test
Sin tumor	2000	1436	159	405
Glioma	1621	1189	132	300
Meningioma	1645	1206	133	306
Pituitario	1757	1312	145	300
Total	7023	5143	569	1311

Tabla 3: Distribución de las imagenes

6.2. Modelos

Antes de describir los modelos a emplear, se tiene que dar una explicación de cómo funcionan las redes neuronales, en particular las redes convolucionales.

6.2.1. Funcionamiento de una neurona artificial y una red neuronal multicapa

En la figura 3 se muestra el modelo de una neurona física y una artificial. Lo que hace una neurona física es recibir, por medio de las dendritas, impulsos eléctricos de sus neuronas vecinas y si la suma de todos ellos es mayor que un cierto umbral, entonces se dice que la neurona se activa y manda un impulso eléctrico a sus vecinas.

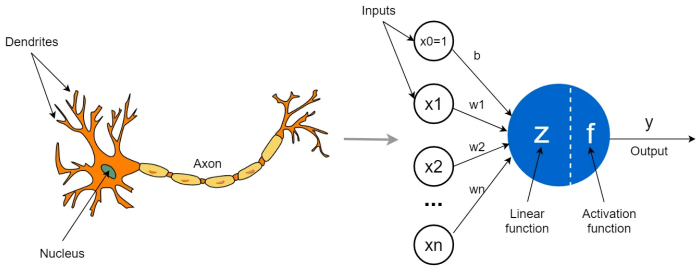


Figura 3: Nerona real vs neurona artificial

Por otro lado, una neurona artificial hace algo similar. Se tiene las entradas (inputs que son vectores), donde cada x_i es un real. Lo que se hace posteriormente es efectuar la operación 1

$$z = bx_0 + \sum_{i=1}^{n} w_i x_i, \tag{1}$$

donde al número b se le conoce como *bias* y a los reales w_i como pesos. Lo anterior descrito modela la suma de los impulsos eléctricos de las neuronas vecinas, por lo que el siguiente paso es determinar si dicha suma supera el umbral y decidir si la neurona se activa o no y para poder modelarlo se usa una *función de activación* denotada por f, la cual es no linear. La operación efectuada es la expresada en

$$y = g(z) = g\left(bx_0 + \sum_{i=1}^{n} w_i x_i\right) \tag{2}$$

Algunos ejemplos de *funciones de activación* son los descritos en la tabla 4.

El resultado que da la neurona artificial es y, el cual transmite hacia todas sus neuronas vecinas. Los valores de los pesos w_i son los que se van a ir modificando durante cada etapa del entrenamiento para que el algoritmo vaya aumentando su desempeño.

Nombre	Formula
Sigmoide o Logística	$f(z) = \dfrac{1}{1+e^z}$
Tangente hiperbólica	$f(z) = \dfrac{e^z - e^{-z}}{e^z + e^z}$
Rectified Linear Unit (ReLU)	$f(z) = \text{máx}\{0, z\}$

Tabla 4: Ejemplos de funciones de activación

Una vez explicado el funcionamiento de una sola neurona, es intuitivo que una red neuronal es el conjunto de múltiples neuronas, tal y como lo muestra la imagen 4 Cada circulo representa una neurona y

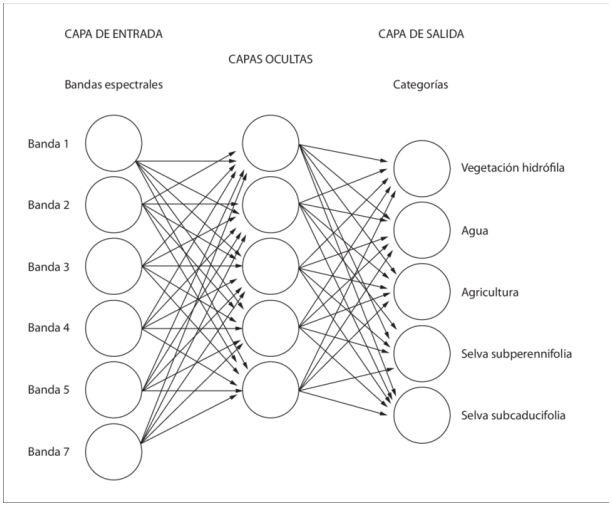


Figura 4: Red Neuronal [26]

cada flecha representa una conexión entre dos neuronas. Cada columna se le llama *capa*, en particular se le llama *capa de entrada* a aquella donde están los inputs. Análogamente, se le conoce como *capa de salida* en donde están todos los outputs y por último están las capas ocultas.

Cabe aclarar que se pueden agregar múltiples capas ocultas o también se puede aumentar el número de neuronas en una sola capa; además no es necesario que todas las capas tengan el mismo número de neuronas, puede que la primera capa oculta tenga 512 neuronas mientras que la segunda 256, la tercera 128 y la última 3. Entre más capas tenga una red se dice que es más profunda (de ahí el nombre de *deep learning*). El modelo recién descrito es conocido como red neuronal multicapa.

6.2.2. Red Neuronal Convolucional

Las redes neuronales usadas para el procesamiento de imágenes son conocidas como Redes Neuronales Convolucionales debido a que en lugar de capas de múltiples neuronas usan capas convolucionales. La idea intuitiva es que se aplica un filtro (conocido como *kernel*) a una imagen. Tanto el kernel como la imagen se representan en la computadora como matrices. Un ejemplo de ambos objetos está en la imagen 5

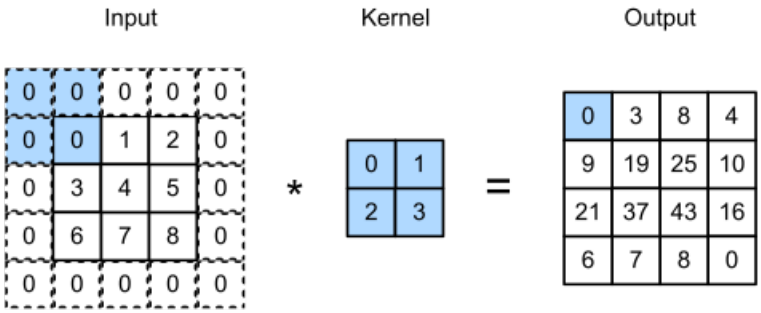


Figura 5: Kernel aplicado a una imagen [27]

La operación descrita está marcada en azul y consiste en superponer la matriz del *kernel* con la de la imagen *(input)* y hacer una multiplicación entrada por entrada y posteriormente sumar dichos valores. El resultado de ir moviendo el *kernel* a través de toda la imagen es otra matriz. La operación de convolución discreta está descrita en la ecuación 3

$$C(h,d) = (k * f)(h,d) = \sum_{i}^{j} k(h-i, d-j) f(i,j), \tag{3}$$

donde k es una imagen (matriz) y f es el kernel. Las principales razones para usar dicha operación son las siguientes:

- En problemas de visión de usa la convolución para verificar la presencia de patrones en imágenes
- Es invariante a la posición que ocupa una característica en la imagen
- No es tan sensible a pequeñas distorsiones
- En 1962 Hubel y Wiesel encontraron que la conexiones en la corteza visual del cerebro de los gatos ejecuta este tipo de operación [28].

Otro elemento importante en las CNN son las capas de Pooling la cual tiene como objetivo reducir las dimensiones de las imágenes y a la vez preservar la información más relevante. Lo anterior se logra aplicando diferentes funciones como por ejemplo el máximo, mínimo o el promedio de potencias aunque la más común es la primera. La capa de pooling necesita como parámetro el tamaño del campo receptivo F y el tamaño de salto S. El primero de ellos hace referencia a una matriz de FxF la cual se va a reducir a un único valor, por ejemplo tomando el valor más grande de dicha matriz; mientras que el tamaño de salto es cada cuanto se va a aplicar dicha operación. Un ejemplo de max pooling es el mostrado en la imagen 6

Finalmente, para los problemas de clasificación se hace una conexión entre la última capa convolucional y una capa completamente conectada, como la mostrada en la imagen 4 y dicha capa es conocida como *flatten layer*. Un ejemplo de una CNN con todos los elementos mencionados se muestra en

Se puede observar la imagen *(input)* a la cual se le aplican $n1$ filtros con un *kernel* de 5×5. Luego de ello hay una capa de Max pooling de 2×2 la cual hace que todos los filtros de la capa anterior sean reducidos de dimensión pero se siguen teniendo todos ellos, es decir, los $n1$. La siguiente capa es

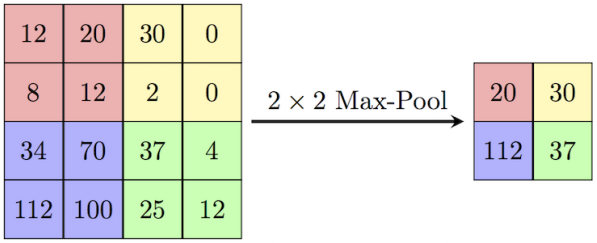


Figura 6: Max Pooling de 2×2 [29]

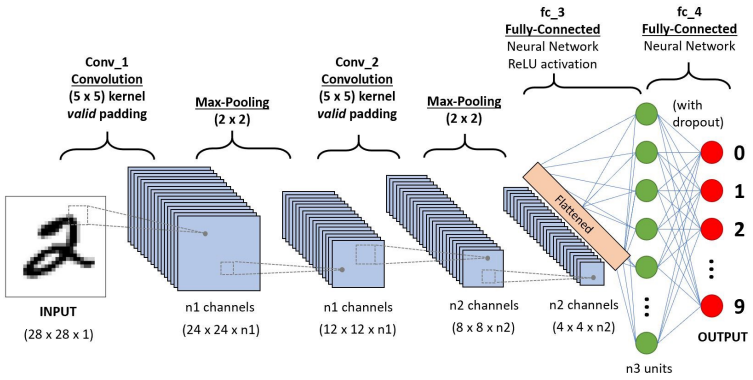


Figura 7: Arquitectura de una CNN [30]

otra convolución de 5×5 de $n2$ filtros para posteriormente procesarlos todos en otro Max pooling de $2\times$ y por último aplicar una capa $flatten$ y pasar a una red neuronal completamente conectada de $n3$ neuronas. La última capa es la de clasificación, en particular se tienen 10 neuronas de salida ya que se trata de una red para clasificar los 10 dígitos.

7. Resultados

Se entrenaron diferentes CNN hechas desde cero y también usando aprendizaje. El optimizador empleado fue *Adam* con tasa de aprendizaje $\alpha \in [0.01, 0.001, 0.0001, 0.00001]$; el criterio para hacer más chico el valor de α es cuando la red ya no mejoraba su aprendizaje. Las arquitecturas empleadas fueron las siguientes:

7.1. CNN VGG16

En la tabla 5 se muestran todas las especificaciones de la red VGG16 que se empleó para aprendizaje por transferencia. Dicha red tiene un total de $139,144,368$ parámetros, de los cuales se entrenaron unicamente $4,883,824$. Se empleó

Layer	Output Shape	Param #

Conv2d-1	[1, 64, 224, 224]	1,792
ReLU-2	[1, 64, 224, 224]	0
Conv2d-3	[1, 64, 224, 224]	36,928
ReLU-4	[1, 64, 224, 224]	0
MaxPool2d-5	[1, 64, 112, 112]	0
Conv2d-6	[1, 128, 112, 112]	73,856
ReLU-7	[1, 128, 112, 112]	0
Conv2d-8	[1, 128, 112, 112]	147,584
ReLU-9	[1, 128, 112, 112]	0
MaxPool2d-10	[1, 128, 56, 56]	0
Conv2d-11	[1, 256, 56, 56]	295, 168
ReLU-12	[1, 256, 56, 56]	0
Conv2d-13	[1, 256, 56, 56]	590,080
ReLU-14	[1, 256, 56, 56]	0
Conv2d-15	[1, 256, 56, 56]	590, 080
ReLU-16	[1, 256, 56, 56]	0
MaxPool2d-17	[1, 256, 28, 28]	0
Conv2d-18	[1, 512, 28, 28]	1,180,160
ReLU-19	[1, 512, 28, 28]	0
Conv2d-20	[1, 512, 28, 28]	2,359,808
ReLU-21	[1, 512, 28, 28]	0
Conv2d-22	[1, 512, 28, 28]	2,359,808
ReLU-23	[1, 512, 28, 28]	0
MaxPool2d-24	[1, 512, 14, 14]	0
Conv2d-25	[1, 512, 14, 14]	2,359,808
ReLU-26	[1, 512, 14, 14]	0
Conv2d-27	[1, 512, 14, 14]	2,359,808
ReLU-28	[1, 512, 14, 14]	0
Conv2d-29	[1, 512, 14, 14]	2,359,808
ReLU-30	[1, 512, 14, 14]	0
MaxPool2d-31	[1, 512, 7, 7]	0
AdaptiveAvgPool2d-32	[1, 512, 7, 7]	0

Linear-33	[1, 4096]	102,764,544
ReLU-34	[1, 4096]	0
Dropout-35	[1, 4096]	0
Linear-36	[1, 4096]	102,764,544
Dropout-37	[1, 4096]	0
ReLU-38	[1, 4096]	0
Linear-39	[1, 1024]	16,781,312
ReLU-40	[1, 1024]	0
Dropout-41	[1, 1024]	0
Linear-42	[1, 512]	524,800
ReLU-43	[1, 512]	0
Dropout-44	[1, 512]	0
Linear-45	[1, 256]	131,328
ReLU-46	[1, 256]	0
Dropout-47	[1, 256]	0
Linear-48	[1, 124]	31,868
ReLU-49	[1, 124]	0
Dropout-50	[1, 124]	0
Linear-51	[1, 4]	500
LogSoftmax-52	[1, 4]	0

Tabla 5: Capas y parámetros VGG16 transfer learning

7.2. CNN's simples

El primer modelo, el cual llamaremos CNN1 chica, está conformado por las siguientes capas:

Layer	Output Shape	Param #
Conv2d-1	[1, 16, 224, 224]	448
ReLU-2	[1, 16, 224, 224]	0
MaxPool2d-3	[1, 16, 112, 112]	0
Conv2d-4	[1, 32, 112, 112]	4,640
ReLU-5	[1, 32, 112, 112]	0
MaxPool2d-6	[1, 32, 56, 56]	0

Linear-7	[1, 128]	128, 854, 184
ReLU-8	[1, 128]	0
Linear-9	[1, 4]	516

El total de parámetros es de $12,850,788$ y todos ellos son entrenables

Por otro lado, se creó otra CNN desde cero pero con un mayor número de capas la cual está conformada de la siguiente manera:

Layer	Output Shape	Param #
Conv2d-1	[1, 128, 224, 224]	3,584
ReLU-2	[1, 128, 224, 224]	0
MaxPool2d-3	[1, 128, 112, 112]	0
Conv2d-4	[1, 64, 112, 112]	73,792
ReLU-5	[1, 64, 112, 112]	0
MaxPool2d-6	[1, 64, 56, 56]	0
Conv2d-7	[1, 32, 56, 56]	18,464
ReLU-8	[1, 32, 56, 56]	0
MaxPool2d-9	[1, 32, 28, 28]	0
Conv2d-10	[1, 16, 28, 28]	4,624
ReLU-11	[1, 16, 28, 28]	0
MaxPool2d-12	[1, 16, 14, 14]	0
Conv2d-13	[1, 8, 14, 14]	1,160
ReLU-14	[1, 8, 14, 14]	0
MaxPool2d-15	[1, 8, 7, 7]	0
Linear-16	[1, 128]	50,304
ReLU-17	[1, 128]	0
Dropout-18	[1, 128]	0
Linear-19	[1, 4]	516

El número de parámetros totales es 152,444 y todos ellos son entrenables.

Finalmente, una última CNN simple fue entrenada pero con la característica de que tiene redes convolucionales apiladas a diferencia de las otras dos redes en las cuales había una capa de MaxPool entre

cada capa covolucional. Su arquitectura es la siguiente:

Layer	Output Shape	Param #
Conv2d-1	[1, 128, 224, 224]	3,584
Conv2d-2	[1, 128, 224, 224]	147,584
ReLU-3	[1, 128, 224, 224]	0
MaxPool2d-4	[1, 128, 112, 112]	0
Conv2d-5	[1, 64, 112, 112]	73,792
ReLU-6	[1, 64, 112, 112]	0
Conv2d-7	[1, 32, 112, 112]	18,464
ReLU-8	[1, 32, 112, 112]	0
MaxPool2d-9	[1, 32, 56, 56]	0
Conv2d-10	[1, 16, 56, 56]	4,624
Conv2d-11	[1, 16, 56, 56]	2,320
ReLU-12	[1, 16, 56, 56]	0
MaxPool2d-13	[1, 16, 28, 28]	0
Conv2d-14	[1, 8, 28, 28]	1,160
ReLU-15	[1, 8, 28, 28]	0
MaxPool2d-16	[1, 8, 14, 14]	0
Linear-17	[1, 512]	803,328
ReLU-18	[1, 512]	0
Dropout-19	[1, 512]	0
Linear-20	[1, 256]	131,328
ReLU-21	[1, 256]	0
Dropout-22	[1, 256]	0
Linear-24	[1, 4]	1,028

Ésta última red tiene un total de $1,187,212$ parámetros y todos son entrenables.

7.3. AlexNet

Se usó aprendizaje por transferencia empleando la red pre-entrenada AlexNet. La arquitectura de dicha red está en la siguiente tabla:

Layer	Output Shape	Param #
Conv2d-1	[1, 64, 55, 55]	23,296
ReLU-2	[1, 64, 55, 55]	0
MaxPool2d-3	[1, 64, 27, 27]	0
Conv2d-4	[1, 192, 27, 27]	307,392
ReLU-5	[1, 192, 27, 27]	0
MaxPool2d-6	[1, 192, 13, 13]	0
Conv2d-7	[1, 384, 13, 13]	663,936
ReLU-8	[1, 384, 13, 13]	0
Conv2d-9	[1, 256, 13, 13]	884,992
ReLU-10	[1, 256, 13, 13]	0
Conv2d-11	[1, 256, 13, 13]	590,080
ReLU-12	[1, 256, 13, 13]	0
MaxPool2d-13	[1, 256, 6, 6]	0
AdaptiveAvgPool2d-14	[1, 256, 6, 6]	0
Dropout-15	[1, 9216]	0
Linear-16	[1, 4096]	37,752,832
ReLU-17	[1, 4096]	0
Dropout-18	[1, 4096]	0
Linear-19	[1, 4096]	16,781,312
ReLU-20	[1, 4096]	0
Linear-21	[1, 512]	2,097,664
ReLU-22	[1, 512]	0
Dropout-23	[1, 512]	0
Linear-24	[1, 256]	131,328
ReLU-25	[1, 256]	0
Dropout-26	[1, 256]	0
Linear-27	[1, 128]	32,896
ReLU-28	[1, 128]	0
Dropout-29	[1, 128]	0
Linear-30	[1, 64]	8,256

ReLU-31	[1, 64]	0
Dropout-32	[1, 64]	0
Linear-33	[1, 4]	260

El número total de parámetros es $59,274,244$, de los cuales $2,270,404$ fueron entrenados

8. Resultados

En la seccón presente se va a mostrar el desempeño que tuvo cada red neuronal descrita en el apartado anterior. Todas las métricas fueron hechas con el conjunto de imágenes *test*.

En la introducción se mencionó que un factor importante para la selección del mejor algoritmos es aquel que tuviera un menor número de falsos negativos y por esa misma razón se iba a usar la métrica *recall*. Además, también se va a considerar el *accuracy*, el cual mide el número de predicciones que estuvieron correctas. Para poder definir dichas métricas se requiere de la siguiente notación:

- $TP =$ verdaderos positivos
- $TN =$ verdaderos negativos
- $FN =$ falsos negativos
- $FP =$ falsos positivos

Las formulas para el *recall* y *accuracy* son:

$$Recall = \frac{TP}{TP + FN} \qquad Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

El valor de ambas medidas va de 0 a 1, donde un valor más cercano a 1 indicará un mejor rendimiento. Al ver la formula de *recall* observamos que si el valor de los falsos negativos tiende a cero, entonces dicha métrica va a valer 1. Análogamente, si lo falsos negativos y falsos positivos tienden a cero, entonces el *accuracy* tiende a uno.

En la tabla 10 se muestran los resultados

Red	Accuracy	Recall
VGG16	83.22 %	82.21 %
Red Personal Pequeña	97.41 %	97.33 %
Red Personal Grande	96.41 %	96.1 %
AlexNet	85.66 %	84.77 %
Red Personal Convoluciones Juntas	96.5 %	96.52 %

También se presentan las diferentes matrices de confusión para las arquitecturas mencionadas en las imágenes 8

9. Conclusiones

De la tabla 10 notamos que la red neuronal que alcanzó un mayor valor tanto de *accuracy* como de *recall* fue la red persona pequeña, la cual está descrita en la tabla 6, lo cual es sorprendente debido a que de entre las 6 redes entrenadas, la ganadora fue aquella que tiene un menor número de capas y por lo tanto un menor número de parámetros. Además, de entre los 1311 casos del conjunto de *test*, únicamente hubo 4 falsos negativos (Ver la matriz de dicha red y observar la primer columna quitando la primer entrada).

Un factor relevante es que entre más parámetros tenían los algoritmos, mayor tiempo tomaba entrenarlos y es por ello que la red personal pequeña se pudo entrenar durante aproximadamente 60 épocas, mientras que $VGG16$ únicamente durante aproximadamente 40.

Se puede concluir que la red personal pequeña fue la ganadora, lo cual indicia que no siempre es mejor un modelo más complejo para todo.
Como siguientes pasos se sugiere conseguir un equipo que tenga mayor poder de cómputo y volver a entrenar las redes más complejas durante un mayor tiempo para observar si cambia mucho sus métricas de *recall* y *accuracy* o si siguen aproximadamente igual.

Finalmente, si bien no se pudo lograr el valor de 0 falsos negativos, sí se pudo alcanzar un número bajo, y es por ello que se podría utilizar dicho algoritmo como un apoyo para doctores en diagnostico y clasificación de tumores cerebrales.

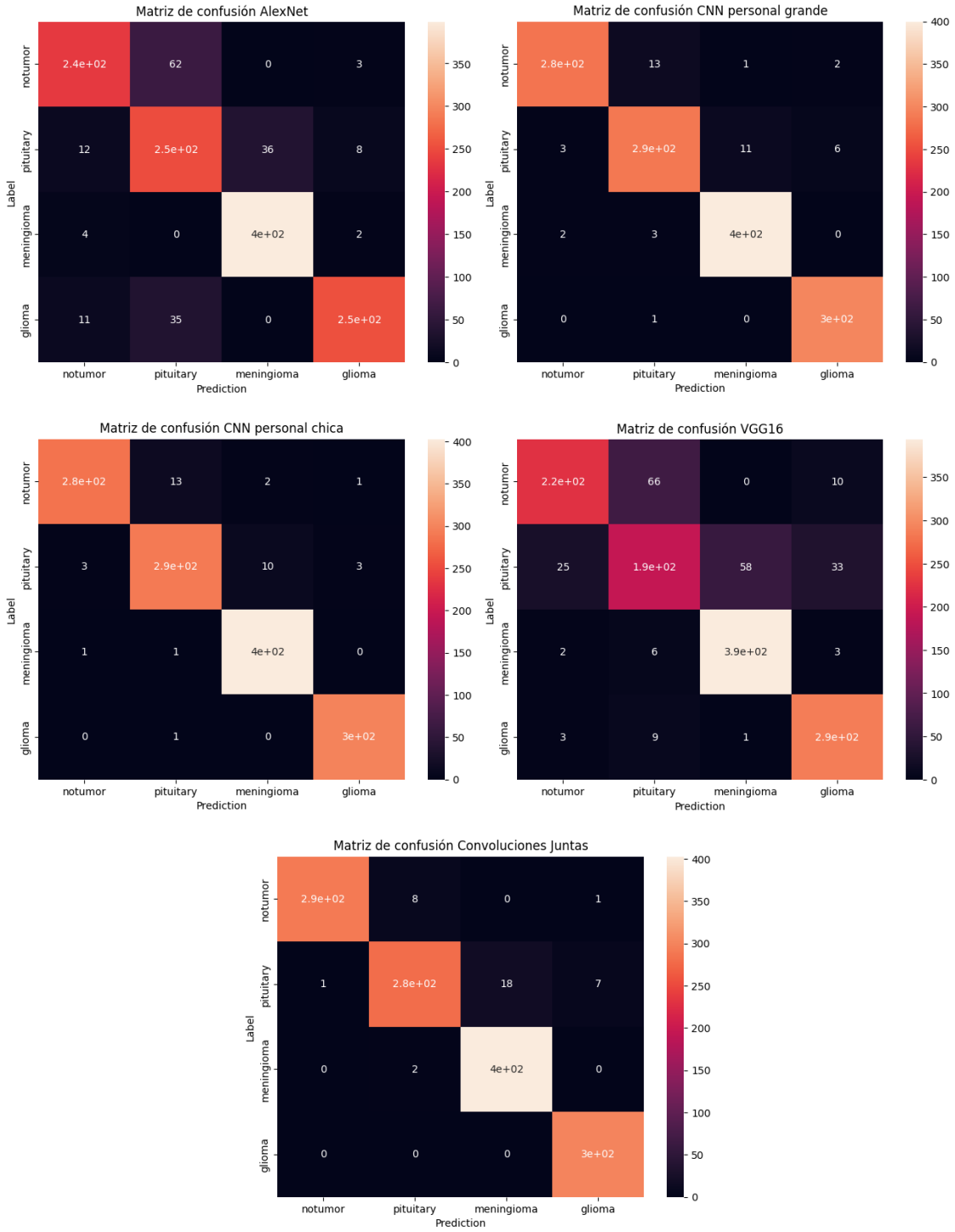


Figura 8: Matrices de confusión de las diferentes redes entrenadas

Referencias

[1] 2023. URL: https://neuronrehab.es/que-tratamos/dano-cerebral-adquirido/tumor-cerebral-tratamiento/.

[2] Aliesha González. *Al Año, diagnostican 300 mil nuevos casos de tumores cerebrales en el mundo.* 2020. URL: https://www.dgcs.unam.mx/boletin/bdboletin/2020_580.html.

[3] URL: https://www.cancer.org/es/cancer/tipos/tumores-de-encefalo-o-de-medula-espinal/deteccion-diagnostico-clasificacion-por-etapas/como-se-diagnostica.html.

[4] Mark H. Bilsky. *Introducción a Los tumores Cerebrales - Enfermedades Cerebrales, medulares Y Nerviosas*. 2023. URL: https://www.msdmanuals.com/es-mx/hogar/enfermedades-cerebrales,-medulares-y-nerviosas/tumores-del-sistema-nervioso/introducci%C3%B3n-a-los-tumores-cerebrales.

[5] 2022. URL: https://www.cancer.net/es/tipos-de-c%C3%A1ncer/tumor-cerebral/introducci%C3%B3n#:~:text=Meningioma.,crece%20en%20el%20tejido%20cerebral..

[6] Jyotismita Chaki y Marcin Wozniak. *Brain Tumor MRI Dataset*. 2023. DOI: 10.21227/1jny-g144. URL: https://dx.doi.org/10.21227/1jny-g144.

[7] 2022. URL: https://cloud.google.com/learn/what-is-artificial-intelligence?hl=es-419.

[8] URL: https://www.ibm.com/mx-es/cloud/deep-learning.

[9] URL: https://www.ibm.com/mx-es/topics/computer-vision.

[10] Natalia Casado Beinat. «Redes neuronales convolucionales y aplicaciones». En: (2022).

[11] Yann LeCun. *Aprendizaje Profundo*. 2020. URL: https://atcold.github.io/pytorch-Deep-Learning/es/week06/06-1/.

[12] Techtalks. *Deep learning en el sector de la biología*. 2023. URL: https://www.atriainnovation.com/deep-learning-en-el-sector-de-la-biologia/#:~:text=Un%20ejemplo%20de%20Deep%20Learning,un%20posible%20c%C3%A1ncer%20de%20piel..

[13] URL: https://research.aimultiple.com/deep-learning-in-healthcare/.

[14] Saad Albawi, Tareq Abed Mohammed y Saad ALZAWI. «Understanding of a Convolutional Neural Network». En: ago. de 2017. DOI: 10.1109/ICEngTechnol.2017.8308186.

[15] Muhammed Talo, Ulas Baran Baloglu, Özal Yıldırım y U Rajendra Acharya. «Application of deep transfer learning for automated brain abnormality classification using MR images». En: *Cognitive Systems Research* 54 (2019), págs. 176-188.

[16] Siyuan Lu, Zhihai Lu y Yu-Dong Zhang. «Pathological brain detection based on AlexNet and transfer learning». En: *Journal of computational science* 30 (2019), págs. 41-47.

[17] Zar Nawab Khan Swati, Qinghua Zhao, Muhammad Kabir, Farman Ali, Zakir Ali, Saeed Ahmed y Jianfeng Lu. «Brain tumor classification for MR images using transfer learning and fine-tuning». En: *Computerized Medical Imaging and Graphics* 75 (2019), págs. 34-46.

[18] Muhammad Sajjad, Salman Khan, Khan Muhammad, Wanqing Wu, Amin Ullah y Sung Wook Baik. «Multi-grade brain tumor classification using deep CNN with extensive data augmentation». En: *Journal of computational science* 30 (2019), págs. 174-182.

[19] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. .ºReilly Media, Inc.", 2022.

[20] Amel Ali Alhussan, Doaa Sami Khafaga, El-Sayed M El-Kenawy, Abdelhameed Ibrahim, Marwa Metwally Eid y Abdelaziz A Abdelhamid. «Pothole and plain road classification using adaptive mutation dipper throated optimization and transfer learning for self driving cars». En: *IEEE Access* 10 (2022), págs. 84188-84211.

[21] S Deepak y PM Ameer. «Brain tumor classification using deep CNN features via transfer learning». En: *Computers in biology and medicine* 111 (2019), pág. 103345.

[22] Coşku Öksüz, Oğuzhan Urhan y Mehmet Kemal Güllü. «Brain tumor classification using the fused features extracted from expanded tumor region». En: *Biomedical Signal Processing and Control* 72 (2022), pág. 103356.

[23] Emrah Irmak. «Multi-classification of brain tumor MRI images using deep convolutional neural network with fully optimized framework». En: *Iranian Journal of Science and Technology, Transactions of Electrical Engineering* 45.3 (2021), págs. 1015-1036.

[24] Milica M. Badža y Marko Č. Barjaktarović. «Classification of Brain Tumors from MRI Images Using a Convolutional Neural Network». En: *Applied Sciences* 10.6 (2020), pág. 1999. ISSN: 2076-3417. DOI: 10.3390/app10061999. URL: http://dx.doi.org/10.3390/app10061999.

[25] Ahmet Çinar y Muhammed Yildirim. «Detection of tumors on brain MRI images using the hybrid convolutional neural network architecture». En: *Medical Hypotheses* 139 (2020), pág. 109684. ISSN: 0306-9877. DOI: https://doi.org/10.1016/j.mehy.2020.109684. URL: https://www.sciencedirect.com/science/article/pii/S0306987720301717.

[26] Jean Mas y Tzitziki Mora. «Comparación de metodologías para el mapeo de la cobertura y uso del suelo en el sureste de México». En: *Investigaciones Geográficas* (ene. de 2008).

[27] Rahul Kadam. *Kernels (filters) in Convolutional Neural Network (CNN), let's talk about them.* 2021. URL: https://medium.com/codex/kernels-filters-in-convolutional-neural-network-cnn-lets-talk-about-them-ee4e94f3319.

[28] David H Hubel y Torsten N Wiesel. «Receptive fields, binocular interaction and functional architecture in the cat's visual cortex». En: *The Journal of physiology* 160.1 (1962), pág. 106.

[29] URL: https://paperswithcode.com/method/max-pooling.

[30] Nafiz Shahriar. *What is Convolutional Neural Network CNN (deep learning)*. 2023. URL: https://nafizshahriar.medium.com/what-is-convolutional-neural-network-cnn-deep-learning-b3921bdd82d5.