Instituto Tecnológico de Buenos Aires

Proyecto Final de Carrera
de Bioingeniería

Clasificación de tumores renales sólidos a partir de imágenes tomográficas, utilizando algoritmos de Deep Learning

---

Alumna: Luciana Rey

Tutora: Candelaria Mosquera

Agradecimientos

En primer lugar, quiero agradecer al Hospital Italiano de Buenos Aires por permitir que me involucre en este proyecto y darme acceso a los datos provenientes de la institución para el desarrollo del mismo. Además quiero agradecer a mi tutora Candelaria Mosquera, que forma parte del Departamento de Informática en Salud, por responder mis dudas y guiarme en el proceso.

Quiero dar gracias a mi familia, especialmente a mi mamá, por estar siempre a mi lado y alentarme en el camino. También a mis amistades por escucharme y en particular a mis amigos del ITBA, quienes cursaron muchas materias conmigo, por acompañarme y mejorar la experiencia.

Índice

1. Resumen

Las masas tumorales renales pueden ser detectadas en tomografías computarizadas con contraste, sin embargo la diferenciación por imagen entre carcinomas de células renales (el tipo más frecuente de tumor maligno [1]) y oncocitomas renales (una de las categorías más comunes de tumor benigno [2]) es una tarea compleja, ya que las dos estructuras presentan características radiológicas similares. En consecuencia, el tratamiento aplicado es la resección quirúrgica de la neoplasia y el diagnóstico de malignidad se determina con el resultado de anatomía patológica después de la cirugía.

El objetivo del proyecto es la creación de un sistema automático que utiliza redes convolucionales para la clasificación de carcinomas y oncocitomas, a partir de tomografías computarizadas multifásicas. Esta herramienta propone un diagnóstico sin requerir un procedimiento invasivo y puede servir como apoyo en la toma de decisión de los urólogos.

Para su desarrollo se realizaron múltiples pruebas, empleando una base de datos anonimizada creada por el Servicio de Diagnóstico por Imágenes del Hospital Italiano de Buenos Aires. En los ensayos, se generaron distintas metodologías de preprocesamiento de las imágenes, se definieron factores de diseño y se determinaron posibles estructuras de la red neuronal, trabajando tanto con modelos en dos dimensiones como en tres dimensiones. Finalmente, se destacaron dos redes convolucionales que brindaron valores cercanos o superiores a 0,8 para todas las métricas de evaluación del grupo de testeo: *accuracy*, sensibilidad, especificidad, valor predictivo positivo y negativo, área bajo la curva ROC y Precisión-*Recall*.

Los resultados óptimos obtenidos indican que el uso de *Deep Learning* es un posible enfoque de trabajo para este proyecto. A futuro se espera la conformación de un nuevo conjunto de datos del Hospital Italiano y la evaluación del mismo, para analizar las métricas y verificar la representatividad del grupo de testeo actual respecto a la población total.

2. Glosario de contracciones

CCR: Carcinoma de células renales

ONC: Oncocitoma renal

CCR_n: Carcinoma número **n** de la base de datos del Hospital Italiano de Buenos Aires

ONC_n: Oncocitoma número **n** de la base de datos del Hospital Italiano de Buenos Aires

CNN: Red convolucional

GPU: Unidad de procesamiento gráfico

F_SC: Fase sin contraste

F_C: Fase corticomedular

F_N: Fase nefrográfica

F_E: Fase excretora

LR: *Learning Rate*

VP: Verdaderos positivos

VN: Verdaderos negativos

FP: Falsos positivos

FN: Falsos negativos

VPP: Valor predictivo positivo

VPN: Valor predictivo negativo

ROC: Característica Operativa del Receptor

PR: Precisión - *Recall*

AUC: Área bajo la curva

AUC-ROC: Área bajo la curva ROC

AUC-PR: Área bajo la curva PR

NIfTI: *Neuroimaging Informatics Technology Initiative*

2D: Dos dimensiones

3D: Tres dimensiones

3C: Tres canales

3. Introducción

Según registros mundiales de la Sociedad Americana de Cáncer correspondientes al año 2020, el cáncer de riñón representa el 2,2 % del total de casos nuevos de cáncer, lo que se traslada a 431.288 personas afectadas anualmente debido a dicha enfermedad. Además, estima la muerte de 179.368 pacientes por año como consecuencia de este tipo de afección [3].

En Argentina, en base a datos de la Agencia Internacional de Investigación sobre Cáncer, se ubica al cáncer de riñón en el quinto puesto de tipos de cáncer más comunes (Figura 1), abarcando el 3,9 % de los nuevos casos del 2020 (5.093 pacientes con esta afección) [4].

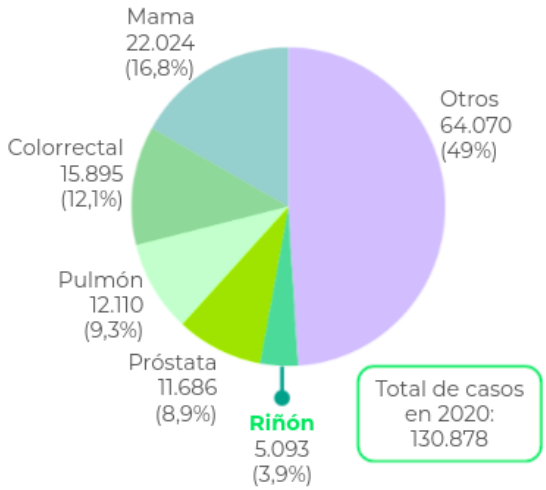

Figura 1: Gráfico circular que indica la cantidad de nuevos casos de distintos tipos de cáncer, estimados en Argentina en el año 2020 (para personas de ambos sexos y todas las edades) [4]. Se observa que el cáncer de riñón se encuentra entre los más comunes.

El cáncer es una enfermedad caracterizada por la formación de células anormales, que se dividen, crecen y se diseminan sin control en cualquier parte del cuerpo, destruyendo los tejidos corporales normales [5]. El carcinoma es el tipo más común de cáncer y se produce a partir de las células epiteliales, es decir, las células que cubren las superficies internas y externas del cuerpo [6]. En el caso del riñón, el tipo de tumor cancerígeno más frecuente es el carcinoma de células renales (CCR), el cual típicamente se origina en la corteza renal (Figura 2) y está presente en 9 de cada 10 casos de cáncer de riñón [1].

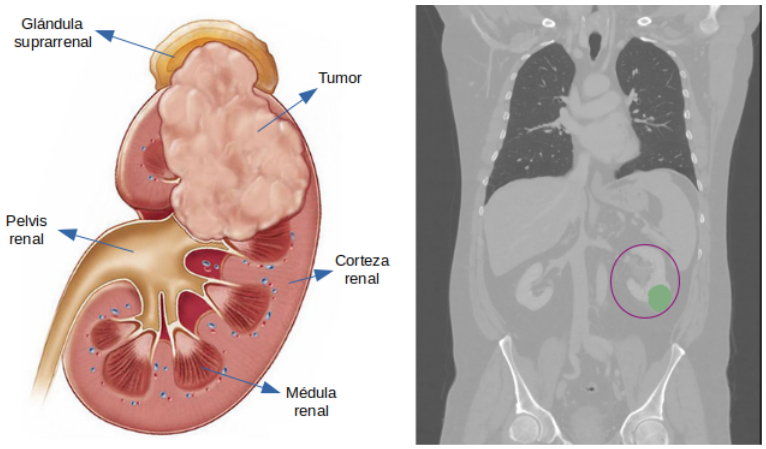

Figura 2: A la izquierda, representación gráfica de corte coronal del riñón con CCR, con tumor en etapa avanzada [7]. A la derecha, corte coronal de tomografía de paciente con CCR (en verde) visualizada en 3D Slicer; el tumor se ubica en la corteza renal del riñón izquierdo (recuadro violeta) [8].

Por otra parte, existen masas renales benignas, como los oncocitomas (ONCs) que derivan de las células del túbulo renal (componente de las nefronas presentes en el riñón). Tanto CCRs como ONCs pueden ser detectados en tomografías computarizadas; sin embargo, la diferenciación por imagen entre estos dos tipos tumorales es una tarea compleja para el médico especialista en *Diagnóstico por Imágenes*, ya que estas dos estructuras presentan características radiológicas similares. Además, la biopsia de tumores renales presenta riesgos clínicos. En consecuencia, el tratamiento aplicado es la resección quirúrgica y el diagnóstico de malignidad se determina con el resultado de la anatomía patológica después de la cirugía. Esto hace que muchas lesiones renales sean operadas aún cuando son benignas.

Debido a esta situación, el desarrollo de un sistema automático de clasificación de tumores es una herramienta útil de apoyo a la toma de decisión de los urólogos, reduciendo interpretaciones visuales erróneas y variabilidad intra-observador, y proponiendo un diagnóstico sin requerir un procedimiento invasivo. El resultado de la clasificación brindado por esta herramienta podría ocasionar un cambio en la conducta quirúrgica especialmente en casos complejos, como pacientes monorrenos (con sólo un tumor) o con múltiples tumores.

El objetivo de este proyecto es crear un algoritmo de *Deep Learning* para la clasificación de CCRs y ONCs, a partir de tomografías computarizadas multifásicas con contraste de pacientes adultos. Para intentar crear un sistema que cumpla con dicha tarea, se probaron distintos enfoques empleando redes convolucionales (CNNs). Se utilizó *Colaboratory* como plataforma de trabajo, la herramienta de *Google* que

permite programar en *Python* [9], se vincula con *Drive* y brinda acceso gratuito a una unidad de procesamiento gráfico (GPU). Es importante también la mención de la biblioteca *Keras* [10], escrita en *Python* con *TensorFlow* [11] como *backend*, que facilita el trabajo con redes neuronales.

El informe se organiza en cinco secciones: se comienza explicando conceptos claves en el *Marco teórico* incluyendo la formación y entrenamiento de CNNs, luego en *Materiales y métodos* se describen las pruebas realizadas con el fin de identificar la mejor opción para el cumplimiento de la tarea de clasificación y en *Resultados* se exponen los casos que brindaron las mejores métricas de evaluación. Por último, en la *Discusión* se analizan dichos resultados y se identifican las características de las pruebas que provocaron efectos positivos en la clasificación, mientras que en la *Conclusión* se determina si se han alcanzado los objetivos del proyecto y se plantean posibles mejoras para implementar a futuro.

4. Marco teórico

En esta sección se explican los conceptos necesarios para entender la metodología de trabajo con *Deep Learning*. Para ello se abarcan tres temas principales: redes neuronales artificiales, procedimientos con datos de la red y entrenamiento del modelo. En adición, se presenta la matriz de confusión usada para la evaluación de la clasificación binaria, y las métricas y curvas vinculadas con dicha matriz.

Por otra parte, se menciona la forma de adquisición de las distintas fases de la tomografía computarizada con contraste. Por último, se desarrolla el estado del arte de la clasificación de tumores renales, mediante la descripción de estudios previos con objetivos similares al de este proyecto.

4.1. Aprendizaje Profundo

El aprendizaje profundo (*Deep Learning*) es un conjunto de algoritmos de aprendizaje automático basado en el uso de redes neuronales profundas. Estas estructuras se conforman por múltiples capas, las cuales pueden interpretarse como representaciones de los datos con distintos niveles de significado o de abstracción [13]. Es un subcampo de *Machine Learning*, donde la extracción y selección de características se realizan de manera automática en el proceso de entrenamiento de la red, que ajusta parámetros de las capas para optimizar el desempeño de la tarea.

Existen tres factores que permiten la aplicación de redes neuronales de alta complejidad en la actualidad: el acceso a gran cantidad de datos, la disponibilidad de poder computacional debido a las arquitecturas de cómputo paralelo provistas por las GPUs y la adopción difundida de marcos de trabajo como *Keras* que ayudan a implementar modelos algorítmicos complejos con líneas limitadas de código.

El *Deep Learning* es ampliamente utilizado en la actualidad para problemas de visión computacional y análisis de imágenes. En particular, para tareas de clasificación es común implementar el aprendizaje supervisado, en el cual se entrena al modelo con un conjunto de ejemplos que incluye tanto los datos de entrada como la etiqueta de clasificación. De esta manera, la red aprende a mapear nuevas entradas con la salida categórica deseada [12].

Por ejemplo, en este trabajo se emplea aprendizaje supervisado para la clasificación binaria de tumores renales, entonces se brinda a la red neuronal: la información tomográfica de los tumores como datos de entrada y los resultados de anatomía patológica (CCR o ONC) como etiquetas. En la Figura 3, se muestra que la red neuronal emplea las imágenes como entrada y compara sus salidas con las etiquetas; en el proceso iterativo de entrenamiento, la red mejora su habilidad para clasificar los datos de entrada. Luego el modelo entrenado puede predecir la categoría de una imagen tomográfica no conocida previamente por la red neuronal.

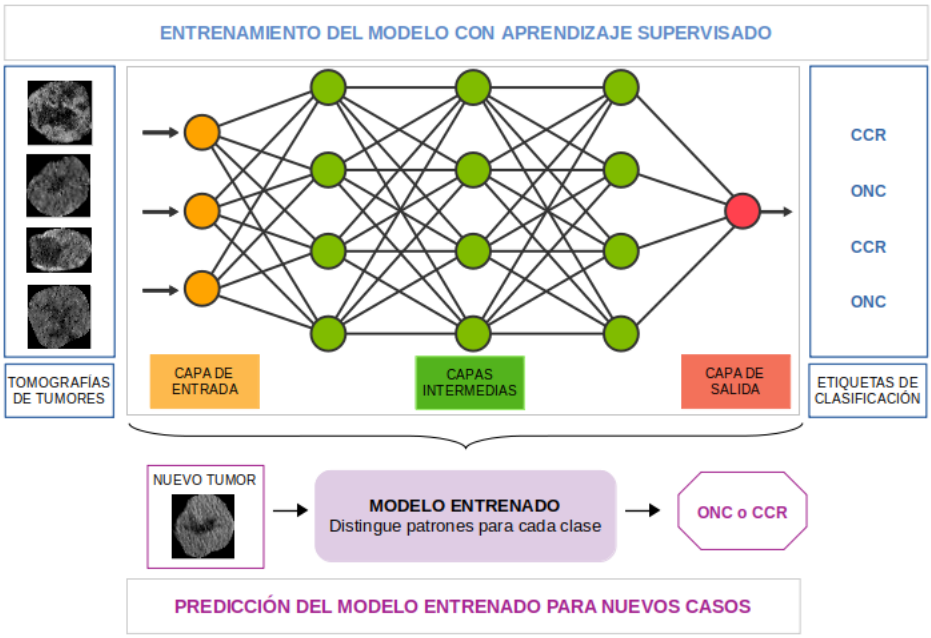

Figura 3: Representación del aprendizaje supervisado. En la sección superior se representa una red neuronal y los datos de entrenamiento que se transmiten a la misma (recuadros azules), en este ejemplo: tomografías de tumores y etiquetas de clasificación. En la sección inferior (en violeta), se simboliza la predicción de la categoría de un nuevo dato de entrada usando el modelo ya entrenado. Imagen adaptada de [14].

4.1.1. Redes neuronales artificiales

En *Deep Learning*, la estructura básica que contiene datos se denomina tensor. Es similar a una matriz, sin embargo es una entidad matemática que pertenece a una estructura e interactúa con otras entidades matemáticas. Por lo tanto, si se aplica una transformación a otra entidad vinculada, el tensor también debe modificarse debido a su propiedad dinámica [13]. En una red neuronal, los tensores se encuentran asociados, entonces un cambio en uno de ellos repercute en la conformación de otros tensores.

La neurona artificial es la unidad de procesamiento de las redes neuronales. Cada neurona transforma un tensor de entrada en un tensor de salida mediante la aplicación de una operación matemática. Estos cálculos pueden estar determinados por parámetros que se ajustan en el entrenamiento de la red (conocidos como pesos), por parámetros fijos (no entrenables), o incluso no tener parámetros [13].

Las neuronas se organizan en capas, y con estas últimas se conforman las redes. La forma en que se conectan las capas, se llama arquitectura del modelo. Las conexiones entre las capas de la red determinan el conjunto de operaciones matemáticas que se ejecutan durante el entrenamiento [13]. La organización más sencilla es la secuencial, donde la salida de una capa es la entrada de la siguiente (Figura 4).

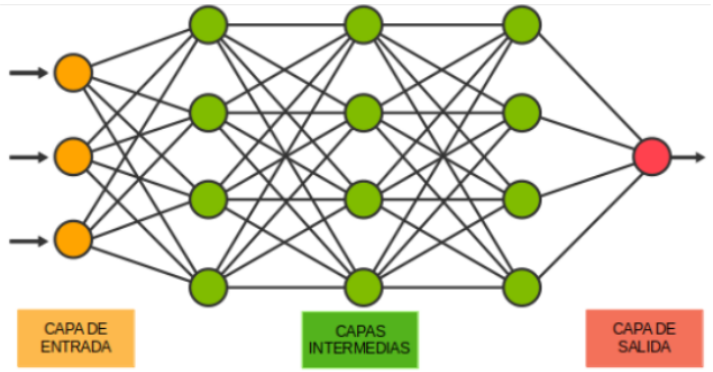

Figura 4: Ejemplo de red neuronal secuencial [14]. Cada círculo representa a una neurona y cada columna a una capa; se distinguen la capa de entrada (en amarillo), las intermedias (en verde) y la de salida (en roja). Las conexiones entre capas determinan la arquitectura del modelo; en este caso se conforma exclusivamente por capas densas.

En la arquitectura del modelo se puede distinguir la capa de entrada, la capa de salida y las capas intermedias u ocultas. En la primera capa se introducen los datos a la red, en las capas intermedias se aprenden representaciones a partir de dicha información, lo que permite brindar un resultado en la capa de salida.

Otra característica del modelo es la cantidad de parámetros que posee, es decir, el número de pesos que deben variarse en el entrenamiento utilizando algoritmos de optimización. Una mayor cantidad de capas en la arquitectura de la red o un mayor número de neuronas en las capas, provoca una cantidad de parámetros superior.

Existen distintos tipos de capas que pueden formar parte de una red neuronal. En las siguientes secciones, se mencionan las capas más utilizadas y se explica su funcionamiento.

4.1.1.1. Capas densas

En un conjunto de capas densas, también denominadas *fully connected*, cada neurona de una capa está conectada con todas las neuronas de la capa anterior y con todas las neuronas de la siguiente. De esta manera, todas las salidas de una capa se toman como entradas de cada neurona de la capa siguiente. A cada conexión entre dos neuronas, se le asigna un peso. La red neuronal expuesta en la Figura 5

está conformada únicamente por capas densas; además, como ejemplo, se señalan los pesos asociados con una neurona de la red.

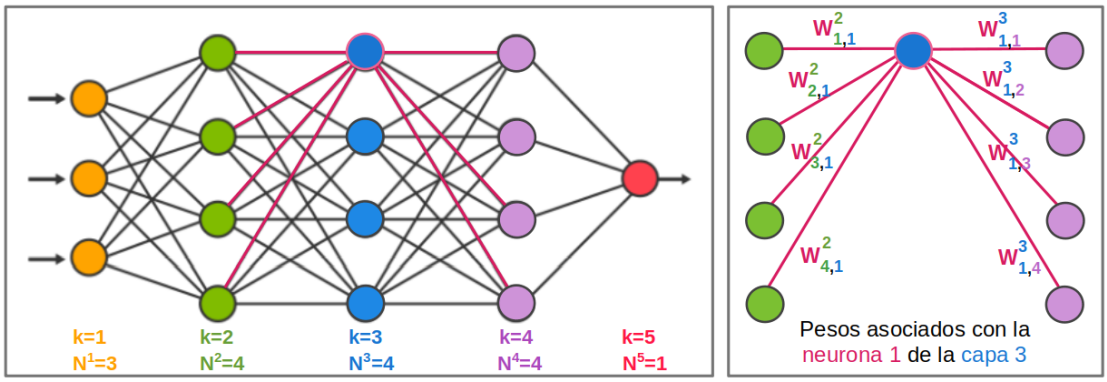

Figura 5: A la izquierda, ejemplo de una red neuronal formada con 5 capas densas; se menciona el número de capa (k) y la cantidad de neuronas por capa (N^k). Se remarca la primera neurona de la capa 3 y sus conexiones con las otras neuronas (en rosa). A la derecha, representación de las uniones de una neurona (en azul) de una capa densa: se conecta con todas las neuronas de la capa anterior (en verde) y con todas las neuronas de la siguiente (en violeta). Los pesos (w) se nombran considerando el número de capa (como superíndice) y los números de ambas neuronas que participan en la unión (como subíndice).

Cada neurona j de una capa densa número k realiza, en primer lugar, una combinación lineal de sus N^{k-1} entradas (x_i^{k-1}) con los pesos correspondientes ($w_{i,j}^{k-1}$) y suma un término *bias* (b_j^k). Luego aplica una función de activación (f), para finalmente obtener la salida (y_j^k). Dicha operación matemática se describe a continuación y en la Figura 6 se muestra un ejemplo:

$$y_j^k = f\left(b_j^k + \sum_{i=1}^{N^{k-1}} w_{i,j}^{k-1}\, x_i^{k-1}\right)$$

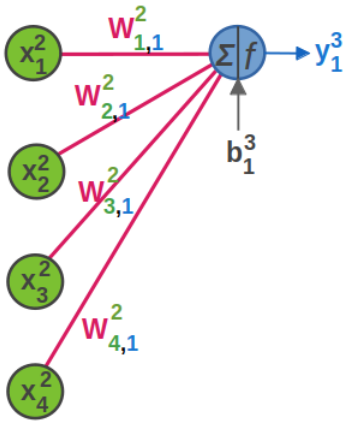

Figura 6: Operación realizada en la primera neurona (j=1) de la tercera capa (k=3) de una red neuronal conformada por capas densas (modelo visualizado en la figura anterior). Las entradas a dicha neurona corresponden a salidas la capa anterior (k=2): x_i^2 donde i es un valor entre 1 y 4 (ya que N^2=4). Los pesos provienen de distintas neuronas de la capa 2 y se dirigen a la neurona 1 de la siguiente capa, por eso se denominan $w_{i,1}^2$ (siendo i el número de neurona de origen de la conexión). Para obtener la salida (y_1^3), en primer lugar se calcula la sumatoria de los productos de cada entrada con su respectivo peso, luego se suma el bias (b_1^3) y, por último, se le aplica la función de activación (f).

En la ecuación, se observa que cada peso pondera a su correspondiente entrada. En el entrenamiento, se ajustan los parámetros según la importancia de cada entrada en el resultado final. Por otra parte, el *bias* representa factores que afectan la decisión, independientemente de las entradas [13].

En cuanto a la función de activación, es importante trabajar con funciones no lineales para lograr detectar e interpretar características no lineales [13], lo cual es necesario para cumplir tareas dentro del campo de visión computacional. La elección de una función de activación varía según la finalidad de la neurona. Por ejemplo, la función sigmoidea que produce una salida con valores entre 0 y 1 (Figura 7 A), puede ser utilizada en la capa final de un modelo de clasificación binaria, ya que el resultado puede interpretarse como una probabilidad de pertenencia a la clase positiva.

Otra función de activación muy utilizada es la ReLU (unidad lineal rectificada), la cual calcula el máximo entre cero y su entrada, obteniendo entonces una salida nula para casos negativos y una salida igual a la entrada para valores positivos (Figura 7 B). Una ventaja de esta función es su sencillez a nivel computacional y su fácil implementación. Además, su comportamiento casi lineal facilita la optimización por métodos basados en gradiente, y preserva propiedades que causan una mejor capacidad de generalización de los modelos lineales [13].

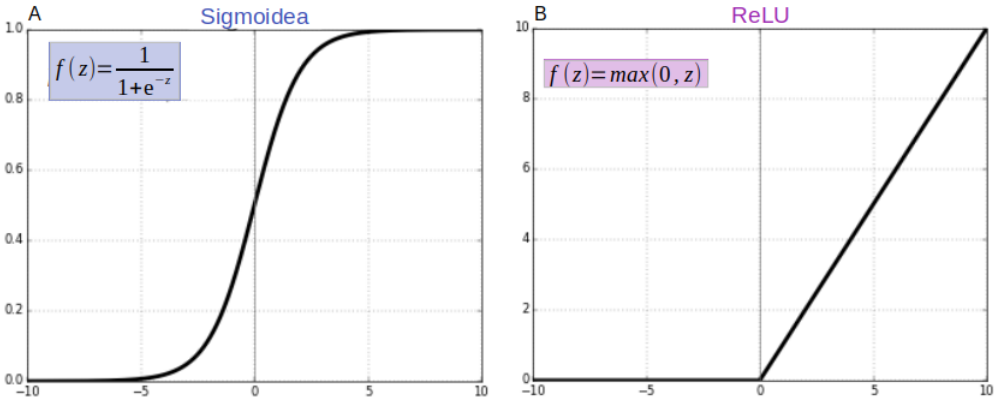

Figura 7: Gráfica y ecuación de dos posibles funciones de activación. A: Función sigmoidea. B: ReLU. En ambos casos, z es la salida parcial de la neurona, es decir, el resultado previo a la aplicación de la función de activación.

Las capas densas son muy utilizadas como clasificadores. Sin embargo, presentan como desventaja un gran costo computacional. En adición, como las entradas de una capa densa se organizan en un vector (tensor unidimensional), si se desea trabajar con imágenes debe considerarse cada píxel de manera separada y se pierde la correlación entre componentes vecinos [13]. Por estas razones, en las tareas de clasificación de imágenes se emplean también las capas convolucionales.

4.1.1.2. Capas convolucionales

La capa convolucional es el componente principal de las redes neuronales convolucionales (CNNs). En este tipo de capa, se comienza la transformación de los datos con el cálculo de la convolución entre las entradas de la capa y los filtros. Los filtros convolucionales (también denominados *kernels*) poseen los pesos que se varían en el entrenamiento del modelo; al modificar dichos valores se cambia el efecto de transformación de la capa y, en consecuencia, la salida resultante.

La finalidad del uso de dichos *kernels* es la extracción automática de características vinculadas con las imágenes de entrada de la red neuronal [13]. Por esta razón, cada salida de una convolución con un filtro se conoce como mapa de características (*feature maps*), es decir, un mapa de dos dimensiones que indica la presencia de un patrón en diferentes ubicaciones en una entrada (Figura 8) [12].

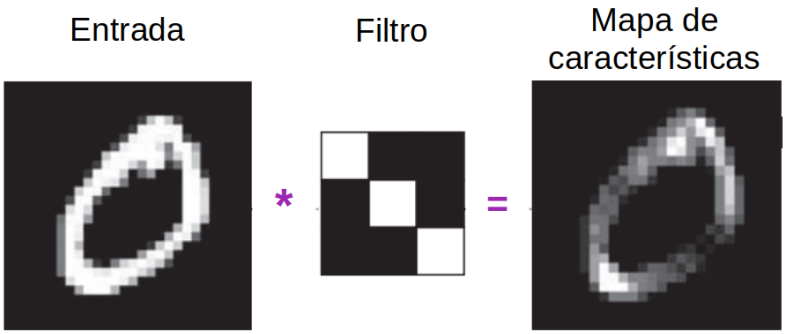

Figura 8: Ejemplificación de la operación convolución [12]. Se usa un filtro que identifica bordes diagonales decrecientes y se aplica el mismo a la entrada. La salida es el mapa de características que indica la presencia del patrón del filtro en diferentes ubicaciones de la entrada. Se observa entonces que los sectores de la salida con mayor intensidad se corresponden con dicho tipo de borde.

Para efectuar operación matemática de convolución, el *kernel* se desplaza sobre el tensor 3D (de tamaño HxWxD) y, en cada posición posible, se extraen secciones o parches del tensor con igual dimensión que el filtro (k_hxk_wxD) [12]. Se calcula el producto escalar entre cada parche y el mismo *kernel*: cada producto genera un valor que forma parte de la salida de la convolución o mapa de características, el cual presenta las dimensiones $(H - k_h + 1)$ x $(W - k_w + 1)$ [13]. En la Figura 9, se visualiza el desplazamiento del filtro sobre la entrada para realizar la convolución; además se indica dónde se ubica el resultado de cada producto escalar en la salida. En la Figura 10, se muestra un ejemplo y se exhiben las operaciones matemáticas implicadas en un producto escalar.

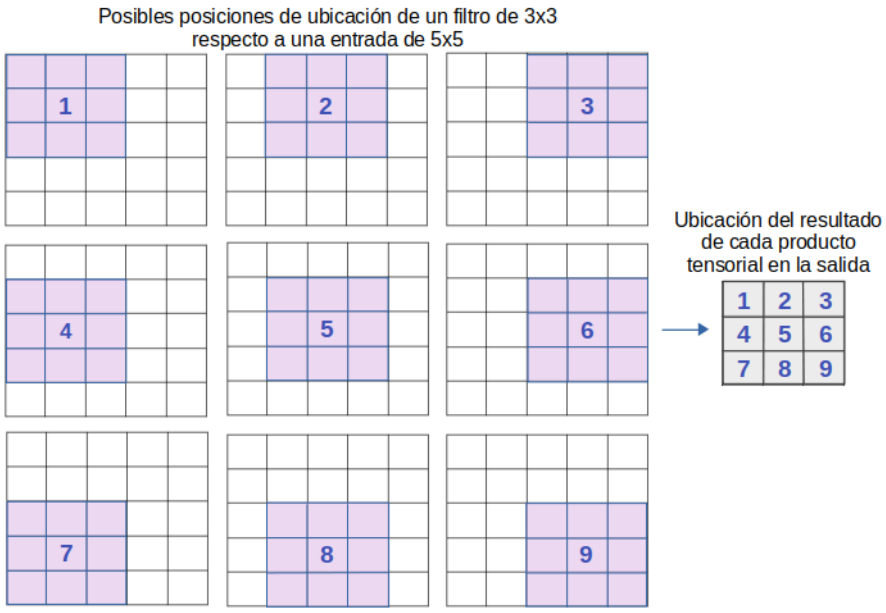

Figura 9: Posibles ubicaciones de un filtro de 3x3 en una entrada de 5x5, para realizar la operación convolución. El filtro (en violeta) debe estar totalmente superpuesto a un parche de la entrada de igual tamaño, para poder calcular el producto escalar. Por lo tanto, se obtienen 9 ubicaciones mediante la movilización del *kernel*: se traslada a la siguiente columna y, cuando no se puede realizar esta acción, a la fila posterior. La salida (en gris) de la convolución se compone con los resultados de dichos productos, según el orden en los que se realiza. En este ejemplo, la dimensión D es igual a 1, por eso la entrada y el filtro pueden representarse en el plano.

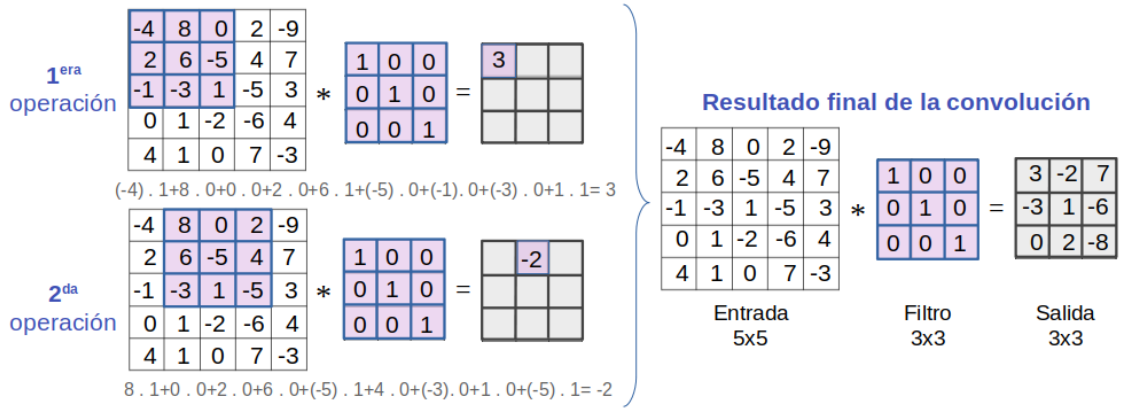

Figura 10: Ejemplo de convolución entre una entrada de 5x5 y un filtro de 3x3. La salida presenta una dimensión de 3x3 ya que se realizan 9 productos escalares, moviendo el *kernel* como se indica en la figura anterior. Las dos primeras de estas operaciones se exhiben en esta representación y además se desarrollan los cálculos de estos productos escalares. Imagen adaptada de [13].

En una capa convolucional se aplican una cantidad determinada de filtros (que se le asigna la letra F). Todos los *kernels* poseen igual tamaño, entonces cada uno genera un mapa de características con distinto contenido, pero igual dimensión. Por lo tanto, todas estas salidas se pueden concatenar en un tensor 3D, que representa la salida intermedia de la capa convolucional con tamaño $(H - k_h + 1)$ x $(W - k_w + 1)$ x F [13]. En la Figura 11, se representa una capa que utiliza dos filtros (es decir, F es igual a 2).

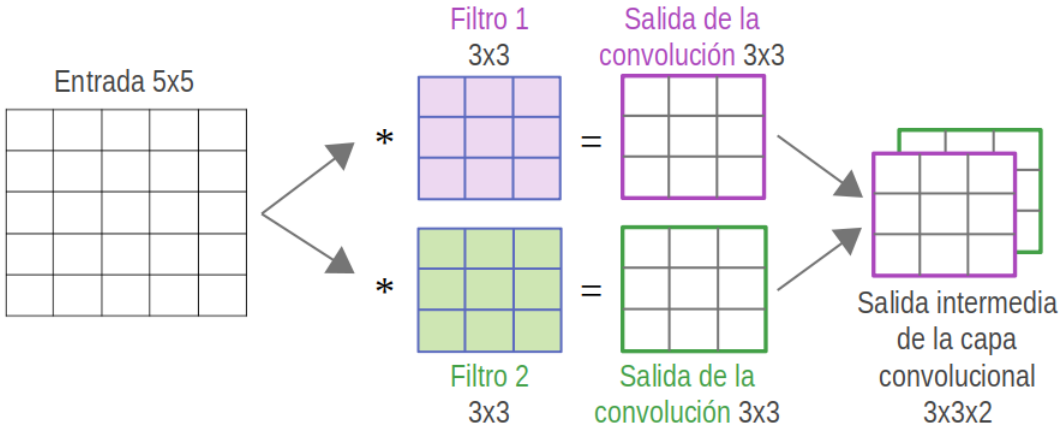

Figura 11: Procedimiento de obtención de salida intermedia de la capa convolucional. En este tipo de capas, se aplica más de un filtro; en este caso se utilizan 2 (uno representado en violeta y otro en verde). La convolución de la entrada (5x5) con cada *kernel* (3x3) brinda una salida distinta. Con estos mapas de características se conforma la salida intermedia de la capa convolucional, de dimensiones 3x3x2. Imagen adaptada de [13].

En la Figura 12, se replica el proceso de convolución realizado en este tipo de capas, pero con representaciones en tres dimensiones. En este caso, las entradas poseen tamaño 5x5x3 y se aplican dos filtros de dimensiones 3x3x3.

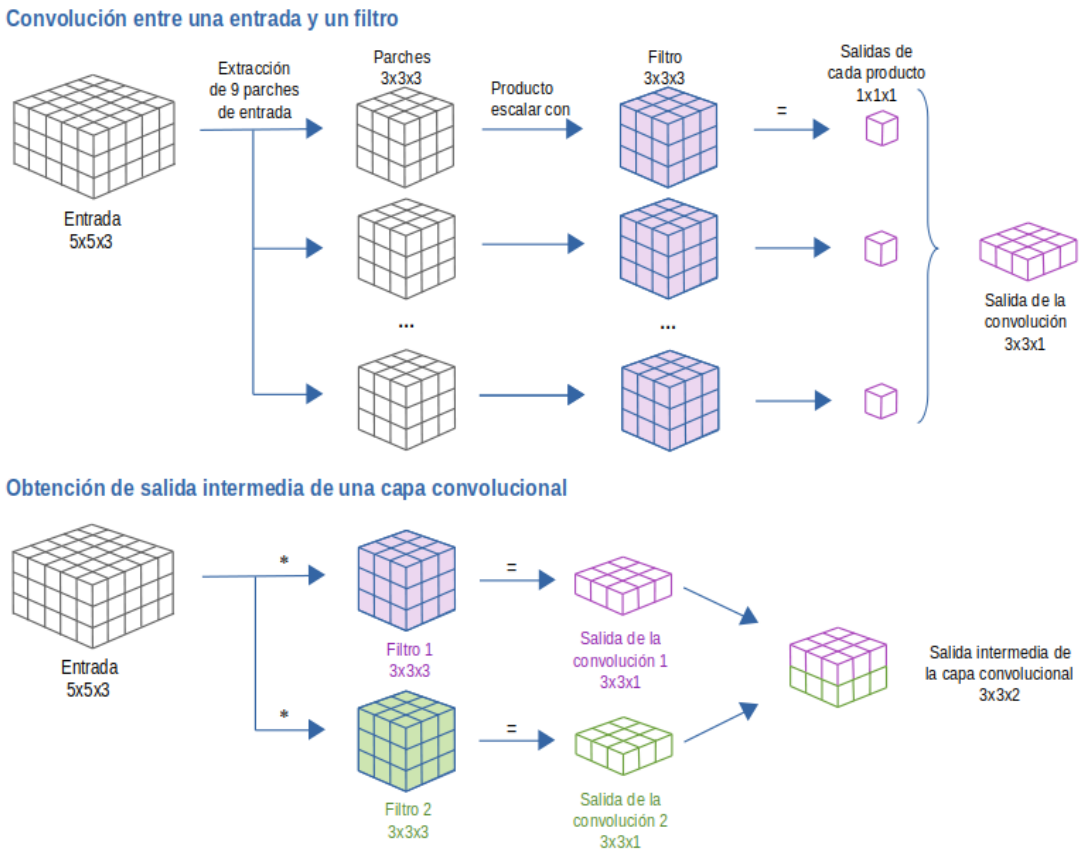

Figura 12: Representación de convoluciones realizadas en una capa convolucional. Por un lado, se esquematiza una convolución entre la entrada y un filtro, y por el otro la unión del resultado de más de una convolución en la salida intermedia. En este ejemplo, las entradas de la capa presentan una dimensión D igual a 3.

Cabe aclarar que la metodología de la convolución se modifica si se aplica alguna de las siguientes técnicas:

- **Uso de *padding*:** Se utiliza con la finalidad de obtener la salida de una capa con las mismas dimensiones de alto y ancho que su entrada. De esta manera, se puede evitar la disminución de tamaño de los mapas de características en una CNN secuencial.

  Se agregan un número de columnas y filas alrededor de la entrada de la capa, para permitir la convolución central de cada vóxel de entrada [12]. Entonces, la cantidad de columnas y filas agregadas depende del tamaño del *kernel*, para un filtro de 3x3 se agrega una columna a la izquierda y derecha y una fila superior e inferior (Figura 13); en cambio para un filtro de 5x5 se suman dos

columnas y dos filas en cada lado del entorno. En particular, el uso de bordes nulos es muy utilizado y se denomina *zero-padding*.

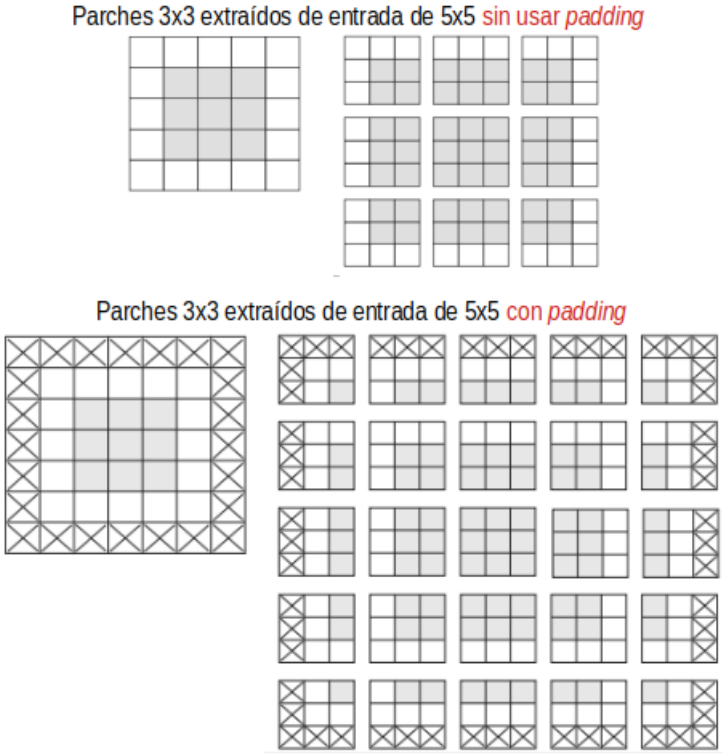

Figura 13: Uso de *padding* en una entrada de tamaño 5x5, para realizar la convolución con un filtro de 3x3. Las columnas y filas agregadas se indican con una cruz. Se visualiza que sin aplicar esta herramienta, sólo se obtienen 9 parches de 3x3. En cambio, implementando *padding* en la entrada, existen 25 ubicaciones posibles donde se puede efectuar un producto escalar; entonces se produce una salida de la convolución de tamaño 5x5 (con iguales dimensiones que la entrada). Imagen adaptada de [12].

- **Desplazamiento de la convolución:** En una convolución, el filtro se centra en vóxeles continuos (como se muestra en la Figura 9), cuando el desplazamiento (*stride*) elegido es 1. Este valor se puede aumentar, entonces el *kernel* saltea posiciones intermedias. Por ejemplo, si se configura un *stride* vertical y horizontal igual a 2, existe una separación de un vóxel entre los centros de los parches extraídos, como se muestra en la Figura 14.

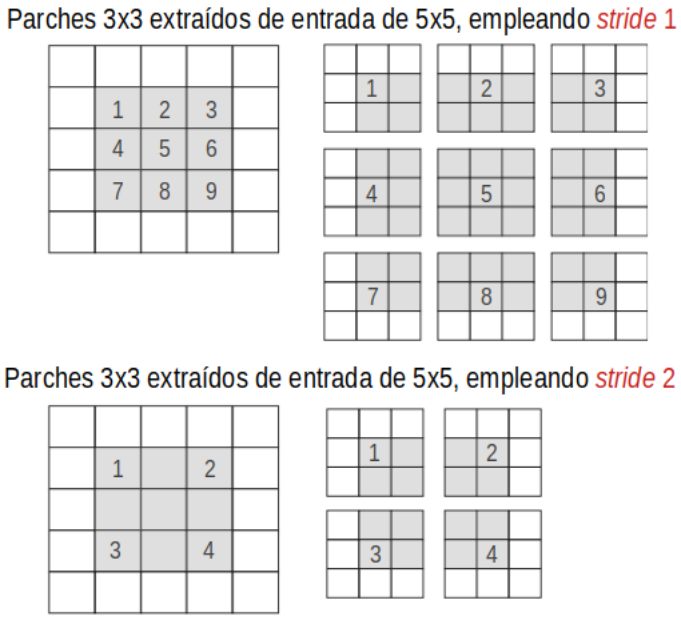

Figura 14: Diferenciación entre la cantidad de parches 3x3 de la entrada (de tamaño 5x5) obtenidos en una convolución con *stride* 1 y otra con *stride* 2. En el último caso, existe una separación de un píxel entre los centros de los parches extraídos; por lo tanto sólo se generan 4 productos escalares dentro de la operación convolución y su salida (mapa de características) es de 2x2. Imagen adaptada de [12].

> La aplicación de convolución con desplazamiento mayor que 1 produce la disminución de las dimensiones del mapa de características. En estos casos, considerando s_h al *stride* vertical y s_w al horizontal, la salida de una capa de convolucional presenta las dimensiones $\left(\frac{H-k_h+s_h}{s_h}\right)$ x $\left(\frac{W-k_w+s_w}{s_w}\right)$ x F [13].

Luego de obtener la salida de la convolución entre las entradas de la capa y los filtros con parámetros entrenables, continúa la transformación de los datos para hallar la salida de la capa convolucional: se suma un *bias* a cada mapa de características (es decir, varía según el filtro utilizado) y se emplea una función de activación. Generalmente esta función aplicada es no lineal, por ejemplo la función ReLU anteriormente mencionada es ampliamente utilizada en este tipo de capas. En la Figura 15, se muestran las operaciones de una capa convolucional.

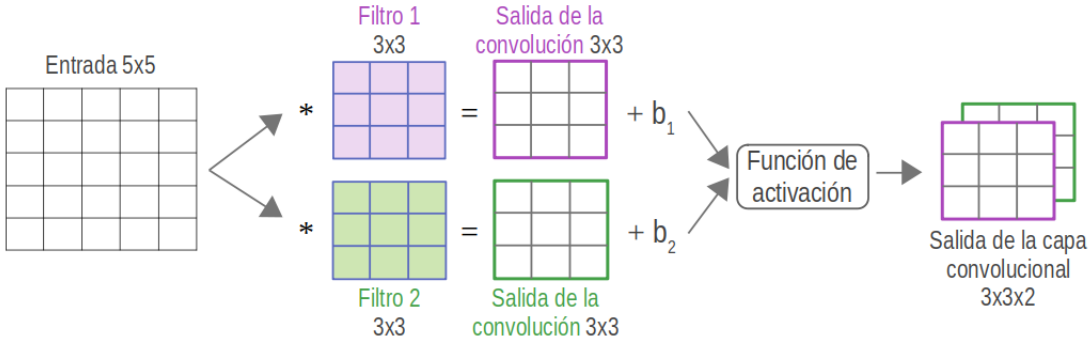

Figura 15: Representación de secuencia de operaciones aplicadas en una capa convolucional. Continuando con el ejemplo de la Figura 11, la entrada es de tamaño 5x5 y se aplican dos filtros de dimensiones 3x3 (sombreados respectivamente en violeta y en verde). Después, a la salida de cada convolución, se le suma un *bias* distinto. Por último, se aplica una función de activación a todos los valores, consiguiendo una salida general de tamaño 3x3x2. Imagen adaptada de [13].

Por lo tanto, la cantidad de parámetros de una capa convolucional corresponde a la suma del número de pesos de cada filtro más sus respectivos *bias*; al poseer los *kernels* igual tamaño, esto equivale al total de parámetros de un filtro más su *bias*, multiplicado por la cantidad de filtros [13]. Por ejemplo, en el caso analizado en la Figura 12 se utilizan 2 *kernels* 3x3x3, entonces la capa posee 56 parámetros.

$$\textit{Número de parámetros de capa convolucional} = (k_h \, k_w \, D + 1) \, F$$

Se determina entonces que la cantidad de parámetros de una capa convolucional es fija (independiente del tamaño de entrada), a diferencia de las redes densas, cuyo número de neuronas en la primera capa depende de la cantidad de píxeles de entrada [13]. El uso de menos parámetros es un beneficio de las CNNs. Otras dos ventajas del empleo de CNNs se explican a continuación:

- *Los patrones aprendidos son invariantes a la traslación.* Mientras que las capas densas memorizan patrones globales de la entrada, las capas convolucionales aprenden patrones locales hallados dentro de parches de la entrada con las dimensiones de los filtros [12]. En otras palabras, con el entrenamiento de la red, un filtro de una capa tiende a representar determinados patrones presentes en la entrada que ayudan a cumplir la tarea asignada; debido a la movilización del *kernel* en la operación convolución, dichos patrones pueden reconocerse en distintas posiciones de la entrada. Este fundamento causa una mayor eficiencia de las CNNs al trabajar con imágenes, ya que facilita la obtención de representaciones que pueden generalizarse en los casos de estudio.

- *Aprenden jerarquías espaciales entre patrones.* A medida que se profundiza en la CNNs las estructuras aprendidas (representadas en los filtros entrenados) son más complejas, como se observa en la Figura 16. Por ejemplo, en la primera capa se aprende pequeños patrones locales como bordes, luego en la segunda capa se aprenden patrones mayores formados por las características de la primera capa, y así progresivamente en toda la red. De esta manera, el modelo aprende de manera eficiente conceptos visuales cada vez más complejos y abstractos [12].

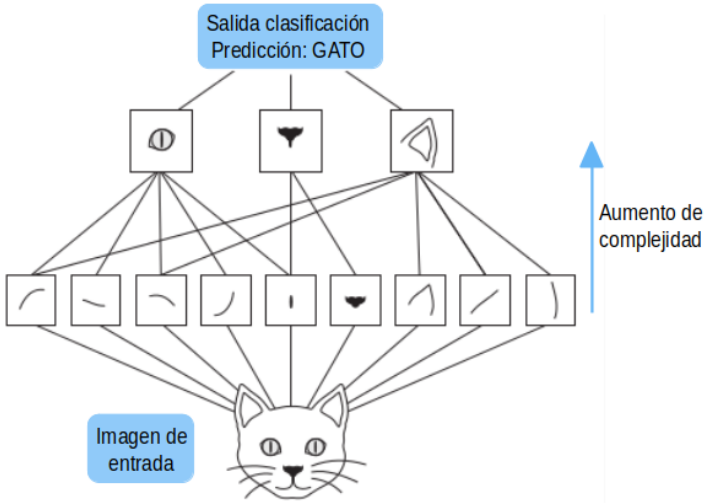

Figura 16: Diagrama representativo del aumento de complejidad de las representaciones aprendidas por la red, a medida que se profundiza en una CCNs [12]. En la primer capa convolucional se analiza la presencia de distintos tipos bordes; mientras que en la segunda capa convolucional pueden distinguirse objetos locales como partes anatómicas del animal. Con un clasificación posterior, a partir de las características obtenidas por la CNNs, se determina que la imagen original corresponde a un gato.

4.1.1.3. Capas *pooling*

Primero debe mencionarse el concepto de campo perceptual. El mismo se define como el tamaño de la región de la imagen de entrada de una CNNs, que produce una característica perteneciente a una salida de una capa determinada (Figura 17). Es importante porque limita la interpretación que puede tener la red de una región de la imagen; por ejemplo, un campo perceptual pequeño puede dificultar la detección o clasificación de objetos grandes con mayor tamaño que el campo perceptual [13].

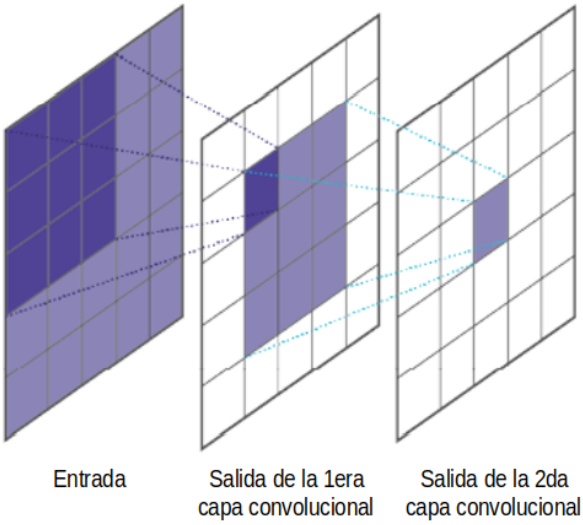

Figura 17: Sombreado del campo perceptual en la entrada [13]. En este ejemplo, se utilizan dos capas convolucionales secuenciales con un filtro de tamaño 3x3 y uso de *padding*. La primera capa genera el campo perceptual marcado en violeta oscuro: cada característica de su salida refiere a un parche de 3x3 de la entrada. En cambio, el campo perceptual determinado por la salida de la segunda capa convolucional, abarca toda la entrada (sombreado en violeta claro).

Con el uso de capas *pooling* se reducen las dimensiones de los mapas de características. De esta forma, se genera un aumento del campo perceptual, ya que provoca que cada componente de la salida refiera a una mayor sección de la entrada original.

Para cumplir su función, una capa *pooling* emplea un *kernel* de tamaño fijo que se desplaza (con cierto *stride*) por cada mapa de características de la entrada y en cada posición calcula un valor de la salida. Las dos capas *pooling* más comunes son el *max pooling* que determina el máximo entre los valores de la entrada dentro de cada *kernel*, y el *average pooling* que computa el promedio. Generalmente se aplican con *kernels* de tamaño 2×2 y *stride* igual a 2, para lograr reducir a la mitad el tamaño del mapa de características [12]. En la Figura 18, se observa un ejemplo de *max pooling* y otro de *average pooling*.

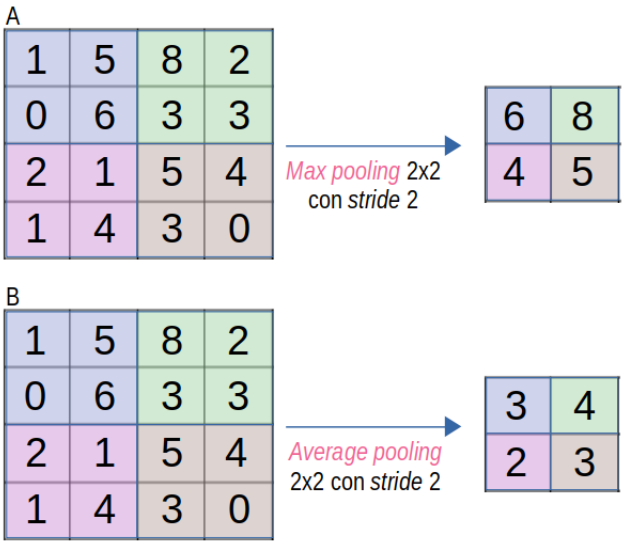

Figura 18: Ejemplificación del funcionamiento de capas *pooling*. Se utiliza un *kernel* de tamaño 2x2 con *stride* 2: con cada color se resalta una posible posición del *kernel* en la entrada (de dimensiones 4x4). A: Se muestra el accionar de una capa *max pooling* que extrae el máximo dentro de cada ventana (resaltada con un color diferente en la figura). B: Se aplica *average pooling*, se calcula el promedio de los valores incluidos en un mismo *kernel*. Imagen adaptada de [13].

Las capas *pooling* suelen colocarse luego de capas convolucionales en CNNs secuenciales. Así se logra que capas convolucionales sucesivas visualicen regiones mayores de la entrada original, lo que colabora en la formación de jerarquías espaciales entre los patrones de los filtros [12]. Además, como las operaciones aplicadas en las capas *pooling* (máximo o promedio) son determinísticas, estas estructuras no poseen pesos [13]. Por lo tanto, su utilización no aumenta los parámetros de un modelo.

4.1.1.4. Capas *global pooling*

Las CNNs contienen capas convolucionales que facilitan la extracción de características a partir de las imágenes de entrada. Al finalizar este bloque, se pueden incluir capas densas que se encarguen de la tarea de clasificación o regresión. De esta manera, todos los pesos se ajustan de manera conjunta con un mismo algoritmo de optimización, minimizando el error de punta a punta (*end-to-end*) .

Las entradas de una capa densa son tensores unidimensionales, a diferencia de las salidas de una capa convolucional que corresponden a tensores de tres dimensiones (HxWxF). En consecuencia, para unir estas dos estructuras en una misma red neuronal, se deben trasladar la información de la última capa convolucional a un vector.

La forma más simple de lograrlo es mediante la operación *flatten*, donde se obtiene un vector con todos los componentes del tensor 3D (de dimensión 1x1x(H.W.F)).

Por otra parte, se puede utilizar una capa de *global pooling*, que genera un valor por mapa de características y crea un vector de tamaño 1x1xF. En una capa de *global max pooling* se toma el mayor valor de cada mapa de características de la salida de la capa convolucional; en cambio en una *global average pooling* se calcula el promedio [13]. En la Figura 19 se ejemplifican ambas capas de *global pooling*.

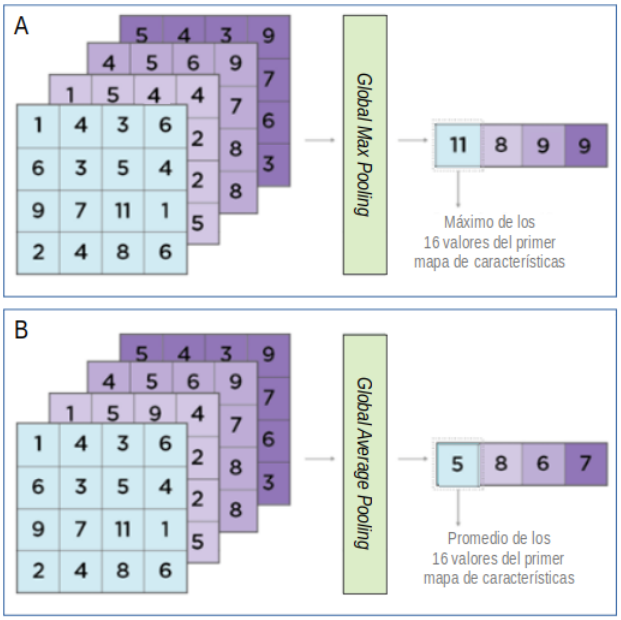

Figura 19: Ejemplo de funcionamiento de una capa *global pooling*. La salida de la última capa convolucional de la red, se corresponde con la entrada de la capa *global pooling*. En este caso, ese tensor 3D presenta un tamaño de 4x4x4; se compone entonces de 4 mapas de características (sombreados en distintos colores). A: Se aplica *global max pooling*, se identifica el máximo valor de cada mapa de características y se coloca en un vector de salida (de dimensiones 1x1x4). B: Se usa *global average pooling*, se computa el promedio de los valores de cada mapa de características y el resultado se transfiere al tensor unidimensional de salida con dimensiones 1x1x4. Imagen adaptada de [13].

4.1.1.5. Otros tipos de capas

En esta sección se mencionan otros dos tipos de capas muy utilizadas en redes neuronales: *batch normalization* y *dropout*.

- ***Batch normalization*:**

  La normalización de la salidas, tanto de capas densas como convolucionales, se puede realizar colocando a continuación una capa de *batch normalization*. Su utilización es importante porque ayuda a la propagación del gradiente para la optimización de los pesos del modelo [12].

  El método más empleado para normalizar datos es centrar la información en cero mediante la resta de la media, y obtener una desviación estándar unitaria al dividir los datos por esta métrica. Se supone entonces que los datos presentan una distribución normal.

  La capa de *batch normalization* ocasiona la normalización de los datos, aunque la media y varianza se modifiquen en el entrenamiento. Dichas medidas son consideradas parámetros no entrenables, ya que varían en el entrenamiento pero no son pesos que deben optimizarse.

- ***Dropout*:**

  Una capa *dropout* es utilizada principalmente a continuación de una capa densa. Su función es descartar aleatoriamente, en cada iteración del entrenamiento, una cantidad de características de la salida de la capa anterior, es decir, multiplicar por cero la salida de un determinado número de neuronas. Para conocer la fracción de salidas que se transforman a cero, se configura la proporción de *dropout* (*dropout rate*); por ejemplo un *dropout rate* igual a 0,25 indica que un 25 % de las salidas son anuladas (Figura 20). Por definición, esta proporción es menor que la unidad, sin embargo se suele elegir una medida menor a 0,5.

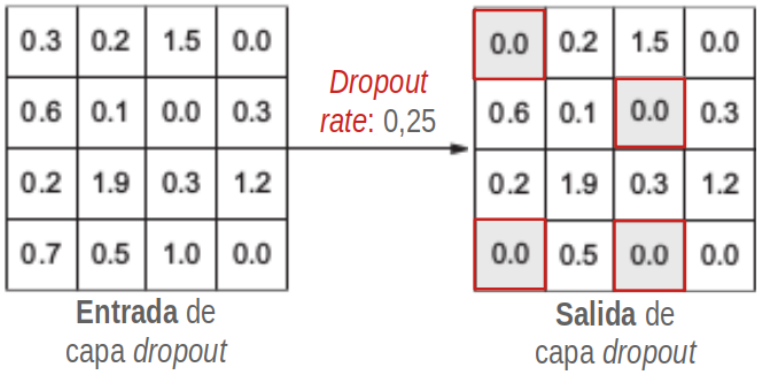

Figura 20: Resultado de la aplicación de una capa *dropout*. La entrada está conformada de 16 valores; con un *dropout rate* de 0,25, se transforman a cero un cuarto de los mismos. En la salida de la capa *dropout*, se remarcan los casos anulados. Imagen adaptada de [12].

Como en cada iteración del entrenamiento se cambia a cero una parte de las salidas de la capa densa de manera aleatoria, en cada oportunidad se selecciona un conjunto nuevo de neuronas anuladas, como se visualiza en la Figura 21. Entonces se utiliza una versión distinta de la red en cada iteración, donde se conservan un grupo diferente de pesos que se optimizan. El objetivo del empleo de una capa *dropout* es entrenar muchas versiones de una red, para aumentar la generalización [13].

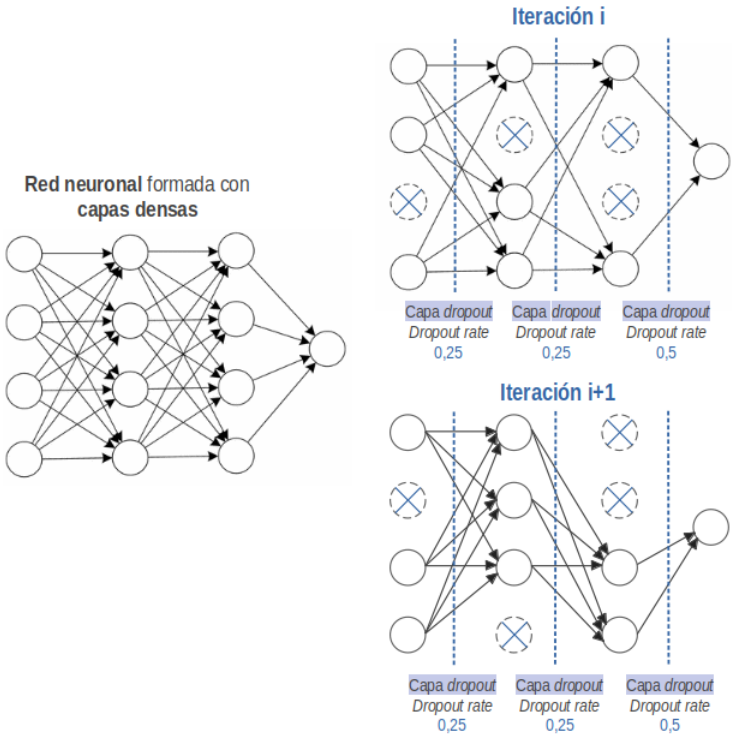

Figura 21: Anulación aleatoria de neuronas en cada iteración de entrenamiento. A la izquierda, se observa una red neuronal conformada por capas densas y sin capas *dropout*. A la derecha, las redes presentan capas *dropout* entre cada par de capas densas secuenciales. Se exponen dos posibles versiones de la red, cambiando las neuronas anuladas; la cantidad de unidades anuladas por capa depende del *dropout rate* de la misma.

Cabe aclarar que al momento de inferir resultados de la tarea de la red neuronal (como clasificación o regresión) en un grupo separado del entrenamiento, no se anulan las neuronas. En cambio, los valores de salida de la capa densa son multiplicados por el *dropout rate*, para equilibrar la presencia de más neuronas activas en comparación con el entrenamiento [12].

4.1.1.6. Definición del modelo con *Keras*

Keras es una biblioteca de código abierto escrita en *Python*, que proporciona una forma útil de definir y entrenar modelos de *Deep Learning*. Entre las ventajas de su utilización se puede mencionar la posibilidad de ejecución en GPU y el soporte integrado para CNNs [12].

Además es una interfaz fácil de utilizar para los usuarios por ser de alto nivel. *Keras* no se ocupa de operaciones de bajo nivel (por ejemplo operaciones con tensores o diferenciación); para esto emplea librerías específicas que le sirven de *backend* como *TensorFlow* [12].

Existen dos maneras principales de crear modelos desde cero, con una arquitectura específica según los objetivos de la red y las características de los datos de entrada. La primera es con la clase *Sequential*, que se emplea únicamente para arquitecturas secuenciales, donde la salida de una capa es la entrada de la siguiente. También puede emplearse la *Functional API*, que permite formar un amplio espectro de arquitecturas con topología no lineal, capas compartidas e incluso múltiples entradas o salidas [15].

Las capas son los componentes básicos de las redes neuronales en *Keras* [16]. Los tipos de capas explicados en las secciones anteriores, se encuentran a disposición del usuario para emplear en el modelo y configurar sus hiperparámetros:

- **Capa densa:** clase *Dense*; se elige el número de neuronas y la función de activación a utilizar.
- **Capa *dropout*:** clase homónima; presenta el dropout rate como hiperparámetro.
- **Capa *batch normalization*:** con clase del mismo nombre.
- **Capa convolucional:** según las dimensiones de las entradas de la red, se usa la clase *Conv1D*, *Conv2D* o *Conv3D*. Entre las configuraciones de la capa a definir se encuentran la cantidad de filtros y su tamaño, el *stride* de la convolución (en todos los ejes), el empleo de *padding* (con la opción "valid"que implica la omisión de esta herramienta y la opción "same"que provoca el uso de zero-padding en la entrada de la capa) y la función de activación.
- **Capa *pooling*:** clases *MaxPooling* o *AveragePooling* mencionando nuevamente las dimensiones de la entrada de la red (1D, 2D o 3D). Se definen los hiperparámetros de tamaño del *kernel*, *stride* de la operación y uso de *padding*.
- **Capa *global pooling*:** clases *GlobalMaxPooling* o *GlobalAveragePooling* (indicando 1D, 2D o 3D).

4.1.1.7. Transferencia de aprendizaje

En *Deep Learning*, otra alternativa a la definición de modelos desde cero, es el empleo de modelos previamente entrenados (redes pre-entrenadas). Estas estructuras se encuentran guardadas y se pueden reutilizar. Fueron entrenadas con una base de datos muy amplia y generalmente en una tarea de clasificación de imágenes a gran escala [12]. Por ejemplo, en muchos casos se utilizan los datos de *ImageNet* [17] (base de datos pública con más de 1.300.000 de imágenes etiquetadas) para aprender a clasificar imágenes entre 1.000 categorías (que incluyen distintos animales, vehículos, artículos de ropa, entre otros).

El fundamento de esta técnica es que, si un modelo se entrena en un conjunto de datos lo suficientemente grande y general, puede servir como un modelo genérico del mundo visual [13]. La transferencia de aprendizaje refiere entonces al concepto de emplear conocimientos adquiridos para cumplir una tarea, en la resolución de otra.

El uso de redes pre-entrenadas suele ser útil para tareas de aprendizaje supervisado con imágenes del mundo real. La adquisición de imágenes (y sus correspondientes etiquetas) específicas de un problema complejo es un trabajo dificultoso [13]; en consecuencia, comúnmente se consiguen base de datos pequeñas. Las características aprendidas por la red pre-entrenada pueden facilitar la resolución de la nueva tarea, a pesar de la limitación de cantidad de datos.

Se diferencian dos formas de emplear los modelos previamente entrenados:

- **Extracción de características (*feature extraction*):** Consiste en usar las representaciones aprendidas por la parte convolucional de la red pre-entrenada, para extraer características de la nueva base de datos [12].

  Para esto se entrena la red con los nuevos datos pero manteniendo las capas convolucionales "congeladas", es decir, sin permitir la actualización de los pesos de las mismas. De esta manera, no se pierden las representaciones aprendidas originalmente y a partir de ellas se obtienen los mapas de características para la nueva información.

  Por otra parte, las capas densas del modelo pre-entrenado no se conservan, ya que las representaciones aprendidas por el clasificador son específicas del conjunto de clases de la tarea original. Por ejemplo, si la red fue pre-entrenada usando *ImageNet*, en las capas densas se consigue información sobre la probabilidad de pertenencia de las imágenes a cada una de las 1.000 categorías posibles.

  Entonces, no se utilizan las capas densas de la red pre-entrenada y en su lugar se colocan nuevas, para que implementen el algoritmo de clasificación o

regresión. Las características extraídas de la base convolucional (y trasladadas a forma de vector) sirven como entrada de las capas densas, cuyos pesos se entrenan con los nuevos datos y, en consecuencia, son propios de la tarea actual. En la Figura 22 se resume el proceso de extracción de características.

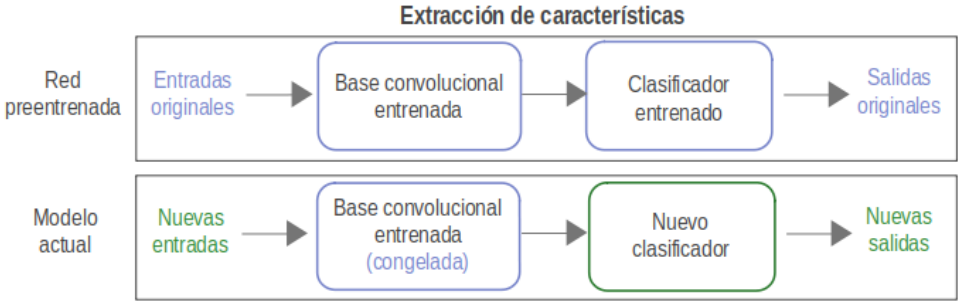

Figura 22: Esquema del proceso de extracción de características. En el recuadro superior se representa la red pre-entrenada (en celeste); en el recuadro inferior se muestran las partes que conforman el nuevo modelo. El modelo actual incluye la base convolucional previamente entrenada (en celeste) con sus capas congeladas; mientras que el clasificador es nuevo (en verde) y se entrena con los nuevos datos.

- **Ajuste fino (*fine-tuning*):** Esta estrategia es similar a *feature extraction*, pero en este caso se vuelven a entrenar algunas capas finales de la parte convolucional de la red pre-entrenada, usando en esta ocasión la nueva base de datos.

  Para su implementación (Figura 23), en primer lugar, con el nuevo conjunto de datos se entrenan las capas densas adicionadas, como se hace en el proceso de *feature extraction*. Luego, se "descongela" un número de capas convolucionales finales y se entrenan conjuntamente estas capas y las densas. El primer paso no puede saltearse, ya que evita que la inicialización del clasificador (o regresor) con valores aleatorios afecte a la conservación de las representaciones de las capas convolucionales, debido a la transmisión del error durante el entrenamiento.

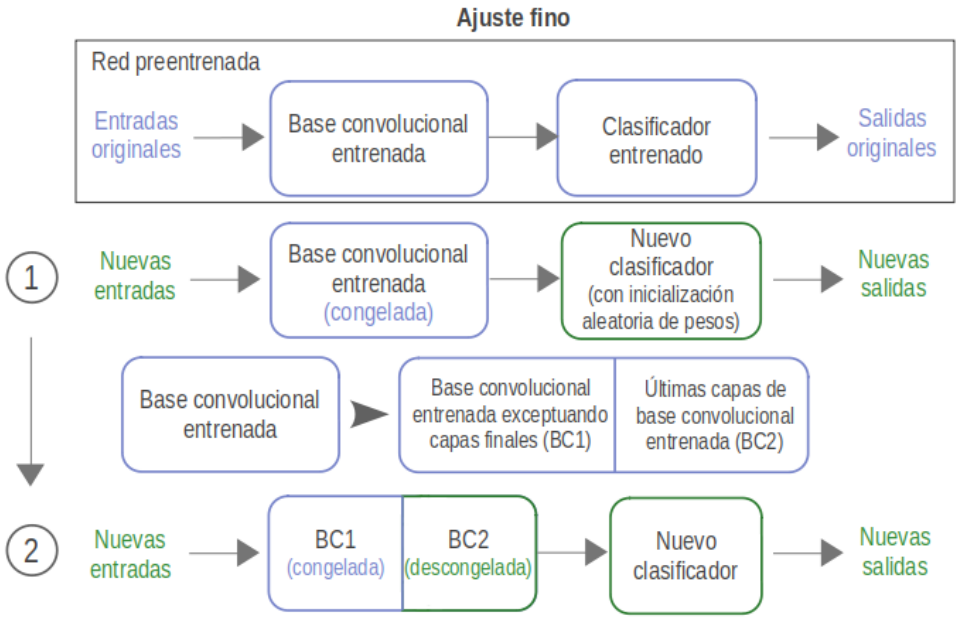

Figura 23: Pasos de la implementación de ajuste fino. En celeste se remarcan los bloques conservados de la red pre-entrenada; en verde se indican bloques que se entrenan con los nuevos datos. La base convolucional entrenada se representa en dos partes (BC1 y BC2) para facilitar la esquematización; BC2 corresponde a las capas que se descongelan en el segundo paso del ajuste fino.

> Con esta metodología se logra ajustar ligeramente las representaciones más abstractas provenientes del modelo pre-entrenado, provocando una mayor relevancia para el problema en cuestión. Se aplica únicamente a las últimas capas convolucionales para modificar sólo las representaciones más específicas de la tarea original y para evitar sumar gran cantidad de pesos entrenables [12].

Algunas familias de arquitecturas pre-entrenadas comúnmente empleadas para tareas de clasificación son: VGGNet, ResNet, Inception. Las mismas pueden implementarse con *Keras*.

4.1.2. Procedimientos con los datos de la red

En la presente sección, primero se menciona la necesidad de realizar la división de los datos disponibles en tres subgrupos, para utilizar cada uno de ellos con una finalidad distinta. Además se introducen los conceptos de *overfitting* e hiperparámetro.

Luego se describe la técnica de aumento de datos, aplicada para incrementar la cantidad de observaciones en el entrenamiento. Por último, se explica el impedimento para trabajar con la totalidad de datos en forma simultánea y la solución brindada con el uso del generador.

4.1.2.1. División de base de datos

Para trabajar con redes neuronales se divide la base de datos total en tres subgrupos: de entrenamiento, validación y testeo. En la Figura 24 se representa la separación mediante el método *hold-out*, donde los conjuntos complementarios formados se mantienen constantes entre distintas iteraciones de entrenamiento. Se observa que el mayor porcentaje de datos se destina al aprendizaje del modelo.

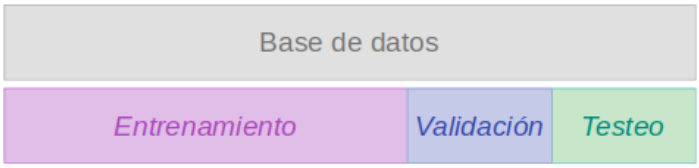

Figura 24: División de la base de datos en tres grupos: de entrenamiento, validación y testeo. En este caso, se utiliza el método de validación *hold-out*. El grupo de entrenamiento se conforma con la mayor cantidad de datos. Los porcentajes destinados para el grupo de validación y de testeo pueden variar según el número de datos totales disponibles.

El modelo se entrena con el primer conjunto mencionado. El objetivo es conseguir una red capaz de generalizar: que represente el mundo real y brinde buenos resultados para nuevos casos [13]. En consecuencia debe evitarse el sobreajuste (*overfitting*), es decir, el proceso en el cual la red aprende patrones específicos del conjunto de entrenamiento, que son irrelevantes para la clasificación de nuevos datos [12]. Principalmente, este fenómeno ocurre si el modelo presenta muchos parámetros o la cantidad de datos de entrenamiento no es suficiente.

Por otra parte, el conjunto de validación se utiliza para evaluar los resultados en un grupo diferente al empleado para el aprendizaje de la red. Sus métricas de desempeño permiten estimar la capacidad de generalización del modelo [13]. Por ejemplo, se puede determinar la presencia de *overfitting*, debido a la diferencia de las mismas con las métricas del grupo de entrenamiento, como se observa en la Figura 25.

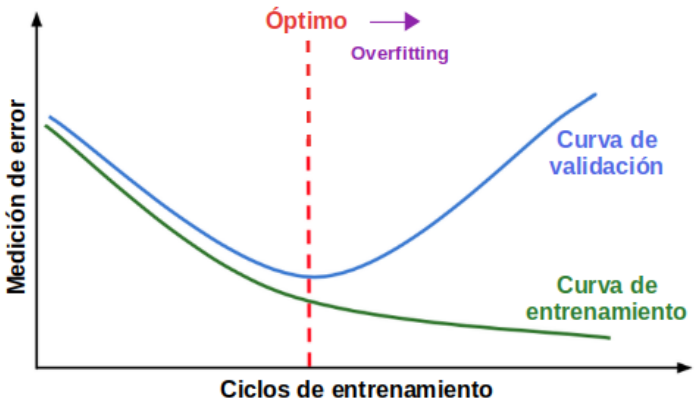

Figura 25: Gráficas de una métrica de medición del error (función de costo) según el número de ciclo de entrenamiento [18]. En verde, función de costo de los datos de entrenamiento; en azul, para el grupo de validación. Una cantidad de ciclos de entrenamiento óptima (línea roja) brinda el mínimo error en validación. Luego se observa que la métrica de error continúa disminuyendo para el conjunto de entrenamiento, mientras que aumenta para el grupo de validación. Analizando ambas gráficas se detecta *overfitting* en esta etapa: el modelo aprende características específicas del conjunto de entrenamiento y, en consecuencia, se reduce el error para dichos datos; por otra parte, la red entrenada no generaliza correctamente, entonces se incrementa el error vinculado al grupo de validación.

En adición, el conjunto de validación sirve para comparar distintas configuraciones de hiperparámetros. Estas decisiones de diseño las define el usuario previo al entrenamiento del modelo, a diferencia de los parámetros de la red que se ajustan de manera iterativa en el entrenamiento. Los resultados del conjunto de validación sirven para determinar cómo se varían los hiperparámetros. De esta manera, se selecciona el modelo más apto para este grupo y por lo tanto también se genera un sobreajuste ligado a dichos datos, denominado fuga de información (*data leakage*) [13].

Por esta razón, se trabaja con un conjunto de testeo. Estos datos son nuevos para la red neuronal, no fueron utilizados para ajustar parámetros o hiperparámetros y se utilizan para la evaluación final del modelo.

4.1.2.2. Aumento de datos

Una opción para evitar *overfitting*, es ampliar la cantidad de observaciones de entrenamiento. El aumento de datos (*data augmentation*) es una estrategia que produce datos sintéticos, mediante la aplicación de transformaciones a datos disponibles [13]. Esta técnica logra incrementar la diversidad del conjunto de entrenamiento con operaciones aleatorias pero realistas [19]. Al trabajar con imágenes, se pueden emplear transformaciones geométricas (como simetría horizontal, rotación y traslación), variaciones en brillo y contraste, acercamientos de la matriz, entre otros (Figura 26).

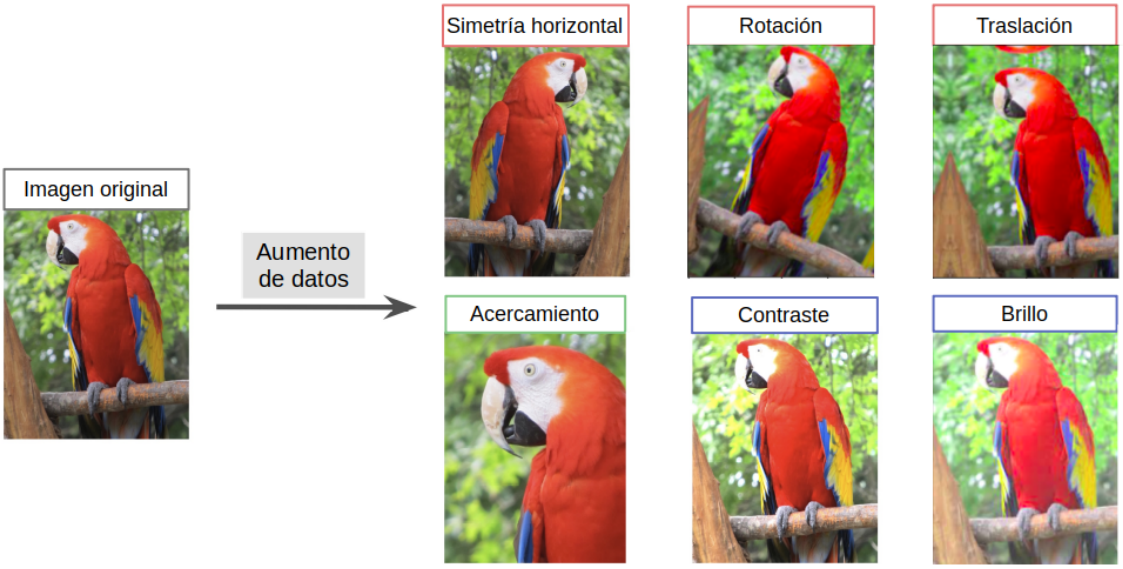

Figura 26: Ejemplos de resultados de la aplicación de *data augmentation* usando *Albumentations*. La imagen original se obtuvo de la página de documentación de dicha biblioteca [20]. Se observan transformaciones geométricas (nombres recuadrados en rojo), de acercamiento (en verde) y de intensidad en la imagen (en azul) aplicadas a la imagen original (en gris), conservando su tamaño. Cabe aclarar que al rotar o trasladar la imagen, quedan píxeles nulos, cuyo valores pueden ser modificados para que el dato sintético no difiera considerablemente del dato original. En los casos expuestos, se eligió la opción de *Albumentations* que utiliza el reflejo de la imagen para completar los bordes nulos.

En el presente trabajo, el aumento de datos se implementa usando la biblioteca *Albumentations* [21]. La misma también permite escalar imágenes a una dimensión elegida (hiperparámetro), función esencial para generar tensores de entrada de la red de igual tamaño. En la Figura 27, se ejemplifica la modificación de tamaño de una imagen.

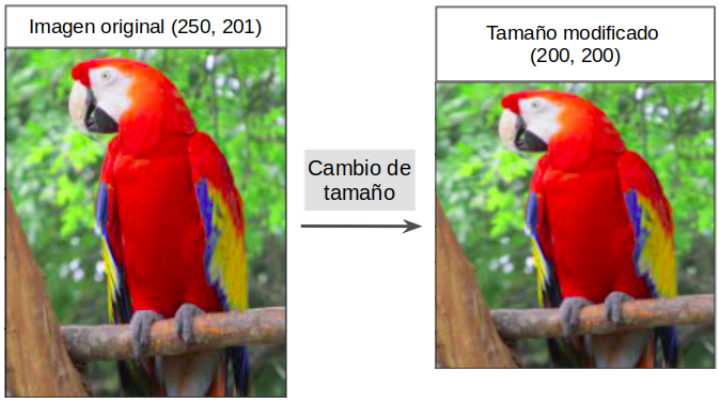

Figura 27: Ejemplo de modificación de tamaño, utilizando *Albumentations*. La imagen original se obtuvo de la página de documentación de dicha biblioteca [20]; es una imagen RGB con dimensiones (250,201). En este caso, se redimensiona la imagen a un tamaño de (200,200).

4.1.2.3. Los *batchs* y el generador

Como en *Deep Learning* se trabaja con muchos datos, no es posible cargar la totalidad de los mismos para el entrenamiento del modelo, ya que se supera la memoria del procesador; además, es sumamente costoso desde el punto de vista computacional. En consecuencia, en cada iteración se utiliza un grupo reducido de observaciones denominado *batch*, adquirido a partir de la base de datos total [13]. El tamaño de esta estructura se define con un hiperparámetro.

La elección de los datos que conforman cada *batch* se realiza de manera aleatoria, y asegurando que la información no se repita en otro *batch*. Una época o *epoch* es un ciclo de entrenamiento que concluye cuando todos los tensores de la base de datos de entrenamiento son enviados (una vez) como entrada de la red neuronal, para ser utilizados en el proceso de aprendizaje [13]. Es decir, un *epoch* incluye varias iteraciones, cada cual emplea un *batch*. La composición de los *batchs* varía en cada *epoch*, como se observa en la Figura 28.

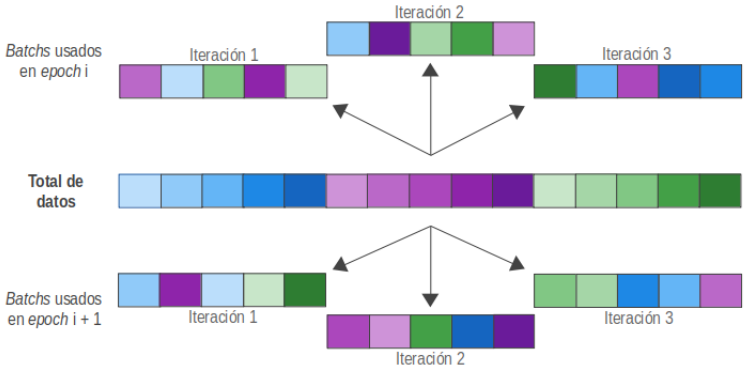

Figura 28: Variación en la composición de los *batchs* para distintos *epochs*. Se visualiza además que cada ciclo de entrenamiento conlleva varias iteraciones y que en cada iteración se trabaja con un *batch* distinto. En un *epoch* se emplean todos los datos sin repetirlos.

En la práctica, trabajando con la biblioteca *Keras*, se utiliza un generador para la formación de los *batchs*. El mismo soluciona el problema de disponibilidad de la memoria RAM, ya que carga las imágenes únicamente cuando se emplean como entradas de la red. Luego de obtener las imágenes, se produce el cambio de sus dimensiones para homogeneizar el tamaño y, para el grupo de entrenamiento, se aplica la estrategia de aumento de datos; con la información obtenida se forma el tensor de entrada del modelo.

Para realizar la selección de observaciones que integran cada *batch*, entre otras maneras, se puede brindar una tabla (*dataframe*) que lista el directorio de ubicación de cada archivo y su respectiva etiqueta (Figura 29). El generador elige de manera aleatoria (y sin repetir casos) filas de esta estructura y conforma los tensores de entrada de la red y también de etiquetas, para ser empleados en el aprendizaje supervisado.

	Nombre	Directorio	Etiqueta
0	Imagen_0	/content/drive/My Drive/Imágenes/Imagen_0.png	0
1	Imagen_1	/content/drive/My Drive/Imágenes/Imagen_1.png	1
2	Imagen_2	/content/drive/My Drive/Imágenes/Imagen_2.png	0
3	Imagen_3	/content/drive/My Drive/Imágenes/Imagen_3.png	0
4	Imagen_4	/content/drive/My Drive/Imágenes/Imagen_4.png	1

Figura 29: Ejemplificación de *dataframe* que indica el directorio de ubicación de cada archivo y su etiqueta; en este caso se estudia la clasificación binaria de imágenes.

Para trabajar con *dataframes* se emplea la biblioteca Pandas [22]. Cabe mencionar que la utilización de estas estructuras como fuente de información del generador, agiliza el trabajo con la base de datos. Por un lado, permite aplicar el sobremuestreo (*oversampling*) de datos de manera sencilla: repitiendo filas de la tabla, modificando únicamente la columna de nombre (Figura 30). Este proceso es útil para balancear las clases de una base de datos desequilibrada, mediante la repetición de filas de la categoría minoritaria, provocando que dichos casos sean considerados más de una vez por el generador. Por otra parte, el uso de *dataframes* facilita la selección de filas y columnas para el filtrado de la información, según los requisitos de cada prueba de entrenamiento.

	Nombre	Directorio	Etiqueta
0	Imagen_0	/content/drive/My Drive/Imágenes/Imagen_0.png	0
1	Imagen_1	/content/drive/My Drive/Imágenes/Imagen_1.png	1
2	Imagen_2	/content/drive/My Drive/Imágenes/Imagen_2.png	0
3	Imagen_3	/content/drive/My Drive/Imágenes/Imagen_3.png	0
4	Imagen_4	/content/drive/My Drive/Imágenes/Imagen_4.png	1
5	Imagen_1b	/content/drive/My Drive/Imágenes/Imagen_1.png	1

Figura 30: Sobremuestreo del *dataframe*. Se repite la fila con información de la imagen 1 y de esta manera se consigue igual número de casos para ambas clases.

Keras posee generadores estándar como *Image Data Generator*, pero también permite crear generadores customizados. Esta última opción, amplia las posibilidades de producción de *batchs*. Por ejemplo, en este proyecto se implementa un generador customizado para poder trabajar con datos en formato *numpy* o brindar información a modelos con múltiples tipos de entradas.

4.1.3. Entrenamiento del modelo

Los pesos de un modelo varían en el entrenamiento, en un proceso de aprendizaje de representaciones asociadas con los datos de entrada y útiles para la resolución de la tarea. El mecanismo de optimización de estos parámetros se explica a continuación.

4.1.3.1. Función de costo y optimizador

Los parámetros o pesos del modelo se modifican en cada iteración del entrenamiento. En las mismas se registra qué tan bien una determinada combinación de pesos representa a los datos existentes mediante el cálculo del error. Este valor analiza la diferencia entre las predicciones de la red para cada dato del *batch* y sus correspondientes etiquetas.

La variación de los pesos en cada iteración se realiza con el objetivo de minimizar dicho error en las predicciones del grupo de entrenamiento generadas por el modelo. Para su aplicación se utiliza la función de costo, que computa el error en la salida de la red, asociado a cierta combinación de parámetros [13], como se diagrama en la Figura 31.

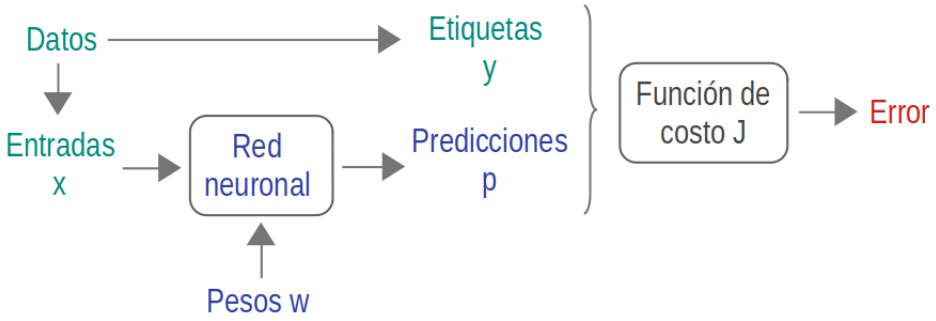

Figura 31: Esquema que expone los elementos asociados con el cálculo del error por medio de la función de costo. En verde se representan los *batchs* de datos que se brindan al modelo para el entrenamiento. En el proceso de aprendizaje varían los pesos aplicados y las predicciones generadas por la red (en azul). Con la función de costo se comparan las predicciones con las etiquetas y se computa el error.

Para tareas de clasificación con dos clases se suele emplear la entropía cruzada binaria como función de costo. Su ecuación se muestra a continuación, donde N es la cantidad de entradas analizadas, y_i es la etiqueta del caso i (negativa o positiva) y p_i es la probabilidad de que el caso i sea positivo predecida por el modelo:

$$J(y_i, p_i) = -\frac{1}{N} \sum_{i=1}^{N} [\, y_i \; log(p_i) + (1 - y_i) \; log(1 - p_i) \,]$$

Se observa que para una etiqueta negativa el primer término de la sumatoria presenta valor nulo; en caso contrario (etiqueta positiva), se anula el segundo término. De esta manera, se halla la entropía de cada caso analizado y la función de costo equivale al promedio de dichos valores con signo negativo.

Por otra parte, en cada iteración, el optimizador modifica los pesos en la dirección que minimiza la función de costo (Figura 32). Para eso calcula su gradiente, es decir, el vector conformado por las derivadas parciales del error con respecto a cada peso de la red [13].

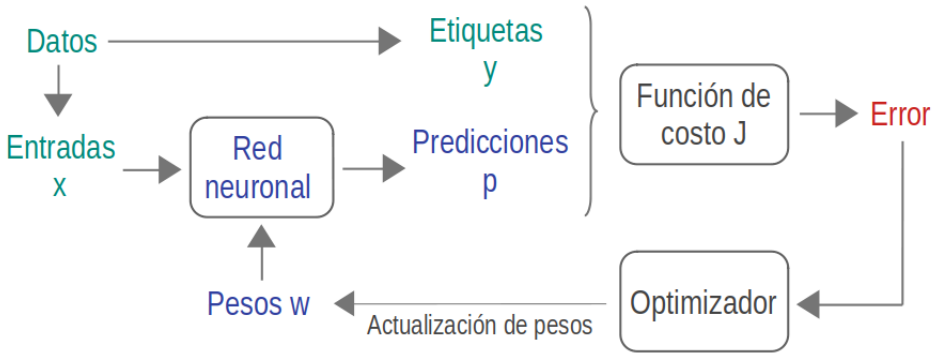

Figura 32: Diagrama del proceso de entrenamiento del modelo. En base al gradiente del error, el optimizador actualiza los pesos con el objetivo de disminuir el error en la siguiente iteración.

Como la red neuronal consiste en operaciones tensoriales conectadas, se pueden hallar las derivadas con la regla de la cadena. Entonces la obtención del gradiente se implementa con el algoritmo de *backpropagation*: se comienza cuantificando el error de la función de costo en la última capa de la red y se extiende el cómputo a capas anteriores consecutivas con la regla de la cadena, hasta calcular la contribución que genera cada parámetro a esta función [12].

Luego se actualizan los parámetros según el método de gradiente descendente, con pasos en el sentido opuesto al gradiente y proporcionales al hiperparámetro *Learning Rate* (LR). La siguiente ecuación expresa el funcionamiento del algoritmo, donde w^k representa los pesos de la k-ésima iteración de entrenamiento y ∇J_{w^k} es el gradiente de la función de costo J respecto del peso w^k.

$$w^{k+1} = w^k - LR\,\nabla J_{w^k}$$

Si se evalúa el caso de un único parámetro, la función de costo presenta dos dimensiones y se puede esquematizar la aplicación de la metodología de gradiente descendente en el plano, como se observa en la Figura 33. En dicho ejemplo, el valor de LR es óptimo. En caso de elegir un hiperparámetro demasiado pequeño, el hallazgo del mínimo del error demora más iteraciones. Por otra parte, si el LR es muy grande, puede no generarse la convergencia al mínimo de la función. En la Figura 34 se muestran estas dos situaciones que deben considerarse al momento de seleccionar el LR del optimizador.

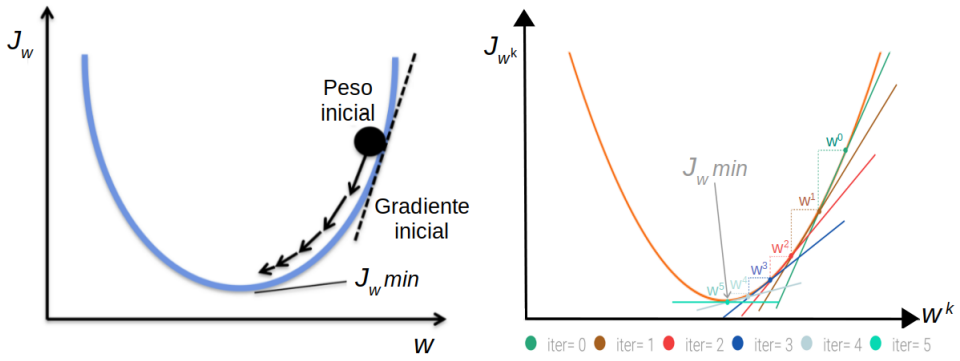

Figura 33: A la izquierda, esquematización del funcionamiento del algoritmo de gradiente descendente [23]. En la representación en el plano (evaluando un ejemplo con un único parámetro), el gradiente es la pendiente de la función de costo. Con un LR óptimo se tiende al mínimo del error. A la derecha, se remarca el gradiente en cada iteración k (∇J_{w^k}) y su modificación según la curva de función de costo [24]. A partir del peso inicial (w^0), se modifican los parámetros utilizando gradiente descendente y, en la iteración 5, se alcanza el mínimo del error.

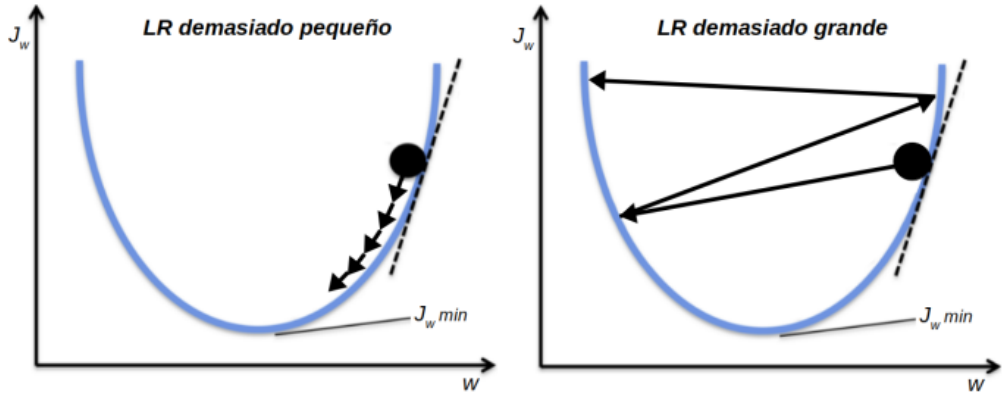

Figura 34: Posibles escenarios al elegir el hiperparámetro LR que se deben tratar de evitar. A la izquierda, se selecciona un LR pequeño que provoca la necesidad de más iteraciones para alcanzar el mínimo del error, en comparación con el ejemplo de la Figura 33 (izquierda) donde se utiliza un LR óptimo. A la derecha, un LR demasiado grande genera que no se alcance la convergencia al mínimo de la función de costo. Imagen adaptada de [23].

Cabe aclarar que si la función de costo es convexa, con un LR adecuada y suficientes iteraciones, el algoritmo de optimización converge. Sin embargo, en muchas ocasiones, la superficie de error definida no es convexa, en consecuencia no se garantiza la convergencia al mínimo global [13]. En la Figura 35, se observa una función de costo de tres dimensiones (dependientes de dos parámetros) y no convexa; se muestran dos

ejemplos de optimización: en el primero (línea negra) se localiza el mínimo global del error, mientras que en el segundo caso (línea marrón) solamente se converge a un mínimo local.

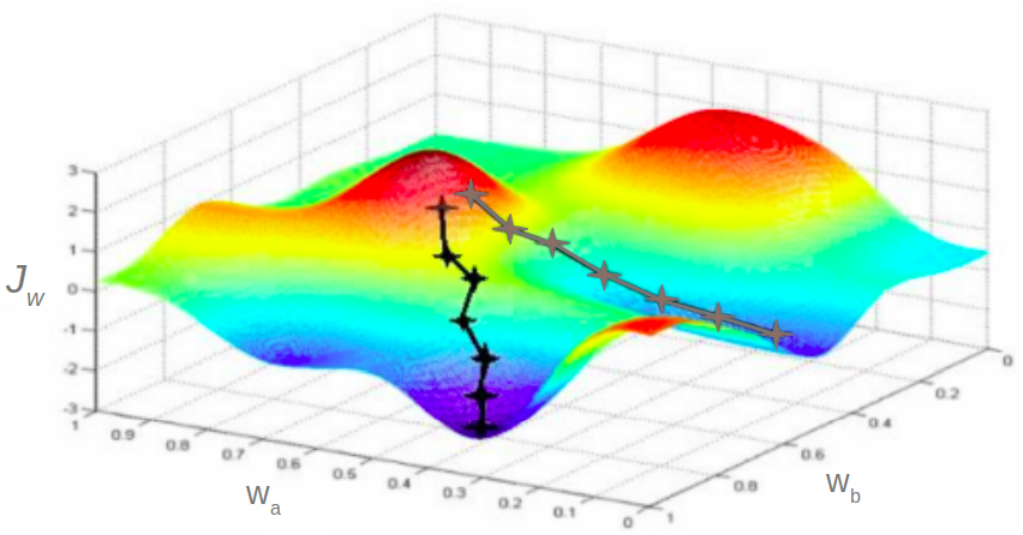

Figura 35: Función de costo no convexa de tres dimensiones y dos posibles caminos de optimización aplicados en el entrenamiento [25]. En negro, se halla el mínimo global del error. En cambio, en marrón se reducen los valores de la función de costo con cada iteración de entrenamiento pero se alcanza un mínimo local. Esta gráfica se presenta con el objetivo de explicar que la convergencia al mínimo global no se garantiza cuando la función de costo no es convexa.

En *Keras* se encuentra implementado el optimizador de gradiente descendente estocástico y algunas variantes del mismo. Por ejemplo, son ampliamente utilizados el optimizador RMSprop (propagación de raíz cuadrática media), Adam y Adagrad.

4.1.3.2. Métricas de entrenamiento

Para monitorear el progreso del entrenamiento, se utilizan métricas que se evalúan al final de cada *epoch*. Se calculan tanto para el grupo de entrenamiento como para el conjunto de validación, pero su cómputo es distinto.

Para el grupo de entrenamiento, el valor de la métrica corresponde al promedio de los resultados de la misma adquiridos en cada iteración del *epoch*. En cambio, para el conjunto de validación se realiza la predicción de todos sus datos sólo con el modelo ajustado con los pesos obtenidos al final del *epoch*; comparando dichas predicciones con las etiquetas se calculan la métricas de este grupo [13].

Entre las métricas monitoreadas siempre se encuentra la función de costo; también se estudian otras medidas como el *accuracy*, el error absoluto medio o el error cuadrático medio. Se analiza la variación de estas métricas con el incremento de los ciclos de entrenamiento, generalmente mediante gráficos como se muestra en la Figura 36.

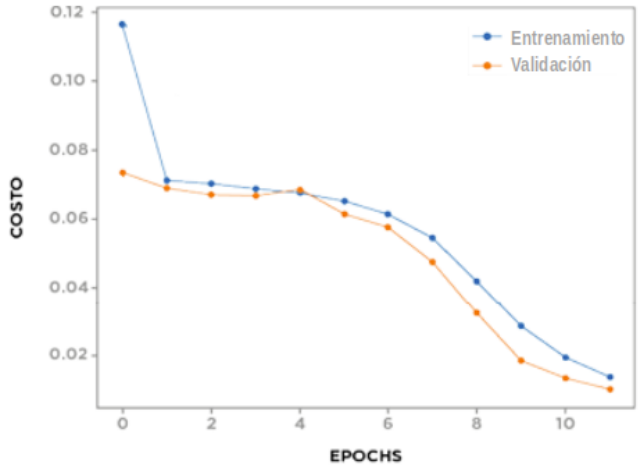

Figura 36: Valores de función de costo en los distintos *epochs*, tanto para el grupo de entrenamiento (curva azul) como para el de validación (curva naranja) [13]. En este ejemplo, se observa el decrecimiento de la métrica de función de costo de los datos de entrenamiento, debido al proceso de optimización. La curva de validación es similar a la de entrenamiento; esto indica una situación de *underfitting*, lo que significa que se debe continuar el entrenamiento por un número mayor de *epochs*, para poder hallar el mínimo de la función de costo.

4.1.3.3. Empleo de *callbacks*

Un *callback* es un tipo de objeto de Keras que se vincula con el modelo en el entrenamiento. Está diseñado para monitorear el desempeño de la métricas del entrenamiento (como la función de costo) y accionar en determinadas circunstancias [12]. A continuación se explica el funcionamiento de los *callbacks* más utilizados:

- *ModelCheckpoint*: Guarda el modelo o sus pesos durante el entrenamiento, luego de finalizar un *epoch*. De esta manera, el modelo se puede cargar posteriormente para continuar el entrenamiento o realizar una predicción. Este *callback* se puede configurar para conservar sólo el modelo que ha logrado el mejor rendimiento de determinada métrica, respecto al resto de los *epochs* [26].

- *EarlyStopping*: Interrumpe el entrenamiento cuando una métrica seleccionada deja de mejorar por un número fijo de *epochs* [12]. La decisión de qué métrica se monitorea y cuántos *epochs* sin su mejora se toleran, son hiperparámetros a definir. Por ejemplo, con este *callback* se puede interrumpir automáticamente el entrenamiento si la función de costo del grupo de validación no mejora en 5 *epochs*, esto ayuda a evitar el *overfitting*.

- *ReduceLROnPlateau*: Realiza la reducción de LR cuando una métrica elegida no varía significativamente en un número fijo de *epochs*. La modificación de LR en estos casos (de "meseta") es una estrategia efectiva para salir de los mínimos locales de la función de costo [12].

- *LearningRateScheduler*: Se define una función de variación de LR, que el *callback* aplica al comienzo de cada *epoch*. Con el nuevo LR se realiza el algoritmo de optimización [27]. Por ejemplo, la reducción programada del LR puede ayudar a hallar el mínimo de la función de costo.

4.1.3.4. Compilación, entrenamiento y predicción con *Keras*

Luego de crear el modelo, el mismo se compila en un proceso de configuración donde se define el optimizador, la función de costo y las métricas a emplear en el futuro entrenamiento. Esto se logra en *Keras* con el método *compile*.

Por otro lado, para el entrenamiento se emplea el método *fit*, que se vincula con el generador el cual le trasfiere los *batchs* creados en cada ciclo, tanto del grupo de entrenamiento como el de validación. En esta función también se determina el número de *epochs* de aprendizaje y los *callbacks* empleados.

Una vez entrenado el modelo, las predicciones a partir de los datos de testeo se adquieren con el método *predict*. Para brindar dichos resultados, los datos de entrada de la red se trasfieren nuevamente con un generador.

4.2. Evaluación de clasificación binaria

La matriz de confusión es una tabla utilizada para describir el desempeño de un clasificador, generalmente aplicado al grupo de testeo [28]. Esta matriz permite visualizar la cantidad de casos de cada clase, clasificados de manera correcta e incorrecta. Considerando una categorización binaria, existen cuatro medidas consecuentes de la comparación entre los valores reales (etiquetas) y los valores predecidos (resultados de la clasificación):

- **Verdaderos positivos (*VP*):** Casos predecidos como positivos cuando la etiqueta también es positiva.
- **Verdaderos negativos (*VN*):** Casos predecidos como negativos y con etiqueta negativa.
- **Falsos positivos (*FP*):** Casos predecidos como positivos, cuya etiqueta es negativa.
- **Falsos negativos (*FN*):** Casos predecidos como negativos, aunque la etiqueta es positiva.

En la Tabla 1, se observa la estructura de la matriz de confusión. En este trabajo, se le asigna la etiqueta positiva al CCR ya que implica la presencia de la enfermedad, en cambio, la etiqueta negativa corresponde al ONC.

La predicción de la red neuronal en base a un dato de entrada, es una probabilidad de pertenencia a la clase CCR; la misma se traduce a un resultado binario mediante el uso de un umbral (valor entre 0 y 1). Entonces si la predicción es superior o igual al umbral, la entrada se clasifica como CCR, en caso contrario, se considera ONC. Al categorizar los resultados, se pueden comparar con las etiquetas y determinar los valores que conforman la matriz de confusión.

		Valor predecido	
		Positiva	*Negativa*
Valor real	*Positiva*	*VP*	*FN*
	Negativa	*FP*	*VN*

Tabla 1: Matriz de confusión para clasificación binaria. El valor real equivale a la etiqueta de cada dato y el valor predecido al resultado de la clasificación obtenido. Se generan cuatro medidas: *VP*, *VN*, *FP* y *FN*.

4.2.1. Métricas de evaluación

Con los valores presentes en la matriz de confusión, se pueden calcular varias métricas de desempeño y evaluación, importantes para interpretar resultados provenientes de la utilización de la red neuronal. A continuación, se definen las mismas y se detallan sus ecuaciones [29]:

- *Accuracy*: Proporción de casos correctamente clasificados respecto del total de observaciones. En el campo médico, se puede interpretar como la razón entre la cantidad de pacientes bien diagnosticados según el resultado del testeo médico, y el total de casos analizados.

$$Accuracy = \frac{VP+VN}{VP+VN+FP+FN}$$

- *Sensibilidad (Recall) o Tasa de Verdaderos Positivos*: Proporción de casos positivos correctamente clasificados, respecto del total de casos positivos. Desde el punto de vista médico, indica qué proporción de pacientes con la enfermedad podrían presentar un resultado positivo [30].

$$Sensibilidad\ (Recall) = \frac{VP}{VP+FN}$$

- *Especificidad o Tasa de Verdaderos Negativos*: Razón entre los casos negativos clasificados de manera correcta y el total de observaciones negativas. En el campo médico, esta métrica representa la proporción de pacientes sin la enfermedad que podrían poseer un resultado negativo [30]. Al observar la ecuación se determina que la *Tasa de Falsos Positivos* = 1 - *Especificidad*.

$$Especificidad = \frac{VN}{VN+FP}$$

- *Valor predictivo positivo (VPP) o Precisión*: Razón entre los casos positivos clasificados de manera correcta y el total de observaciones predecidas como positivas. Aplicado a la medicina, el VPP representa qué proporción de pacientes con resultado positivo podrían padecer la enfermedad.

$$VPP\ (Precisión) = \frac{VP}{VP+FP}$$

- *Valor predictivo negativo (VPN)*: Proporción de casos negativos correctamente clasificados, respecto del total de casos predecidos como negativos. Desde el punto de vista médico, el VPN calcula la proporción de pacientes con resultado negativo que podrían no tener la enfermedad.

$$VPN = \frac{VN}{VN+FN}$$

- *Valor-F1*: Vincula las medidas de precisión y *recall* en un sola medición. Un mejor valor de esta métrica, indica un mejor desempeño del modelo.

$$Valor\text{-}F1 = \frac{2.Precisión.Recall}{Precisión+Recall}$$

Todos las métricas pueden presentar valores entre 0 y 1; cuando más cercano a la unidad se encuentre, más óptimo es el resultado. La sensibilidad, el VPP y el Valor-F1 describen mediciones vinculadas con la clasificación del conjunto de datos positivos. En cambio, la especificidad y el VPN sirven para analizar el desempeño de clasificación de la clase negativa.

Según las definiciones brindadas, se entiende que la sensibilidad y la especificidad representan la validez de la prueba diagnóstica. Mientras, el VPP y el VPN simbolizan la seguridad de esta prueba [31]. Otra diferencia importante es que la sensibilidad y la especificidad no dependen de la prevalencia de la enfermedad, pero el VPP y el VPN sí lo hacen [30].

4.2.2. Curva de Característica Operativa del Receptor y Curva Precisión-*Recall*

Una gráfica utilizada ampliamente para la evaluación del desempeño de la clasificación binaria es la curva ROC (acrónimo de Característica Operativa del Receptor). En esta representación se ilustra la sensibilidad y la especificidad obtenidas al utilizar distintos umbrales de discriminación para la clasificación dicotómica de los datos de la prueba, según sean inferiores o superiores al umbral elegido en cada caso [32].

En la Figura 37 se observa un ejemplo de la curva ROC. El *eje Y* corresponde a la sensibilidad, que también puede denominarse Tasa de Verdaderos Positivos, y el *eje X* a 1-especificidad, es decir, la Tasa de Falsos Positivos; ambos ejes incluyen valores entre 0 y 1. Cada punto de la curva ROC corresponde a un posible umbral de discriminación e informa su resultado de (1-especificidad, sensibilidad).

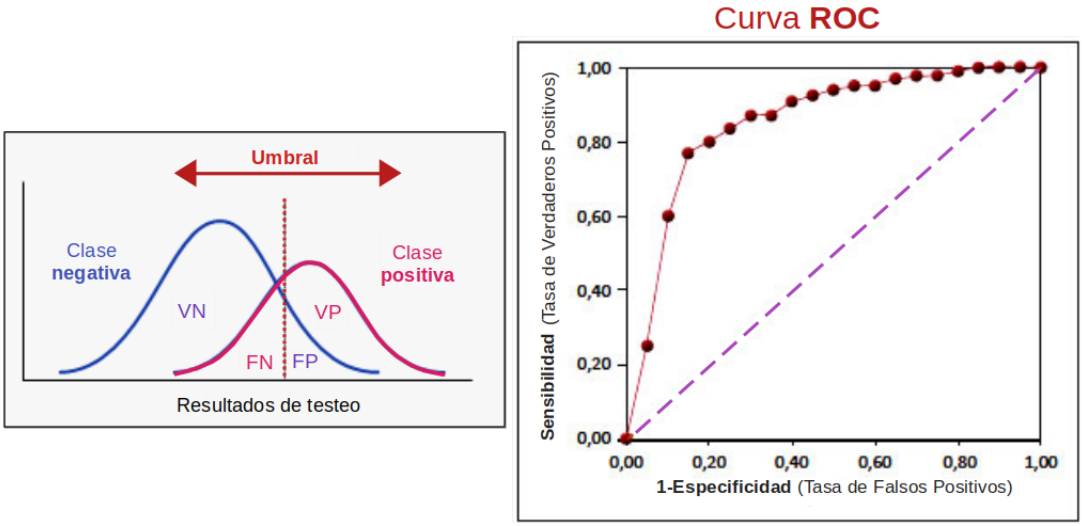

Figura 37: En el esquema de la izquierda, se observa la superposición entre la curva de resultados de los casos negativos (azul) y la de observaciones positivas (rosa). El resultado de testeo se convierte en una clasificación binaria utilizando el umbral (linea roja punteada): los valores a su izquierda se toman como pertenecientes a la clase negativa, mientras que se identifican como positivos los resultados mayores al umbral; a partir de dicho umbral se señalan gráficamente los VP, VN, FP y FN. En la imagen derecha [33], se muestra una curva ROC (en rojo); para representar la misma se toman distintos umbrales de discriminación y cada punto de la gráfica equivale al valor (1-especificidad, sensibilidad) obtenido con un umbral. Por otra parte, la línea de no-discriminación se indica con una línea punteada violeta.

La línea de no-discriminación, diagonal trazada entre los puntos (0,0) y (1,1), funciona como referencia ya que se compone por puntos con la misma proporción de verdaderos positivos y de falsos positivos, es decir, incapaces de discriminar casos positivos de negativos (equivale a una clasificación aleatoria entre ambas clases). La prueba diagnóstica tendrá mayor capacidad discriminativa en la medida que los puntos de la curva ROC se encuentren lo más lejano posible a la línea de no-discriminación y lo más cercano a los lados izquierdo y superior del gráfico [33], como se observa en la Figura 38.

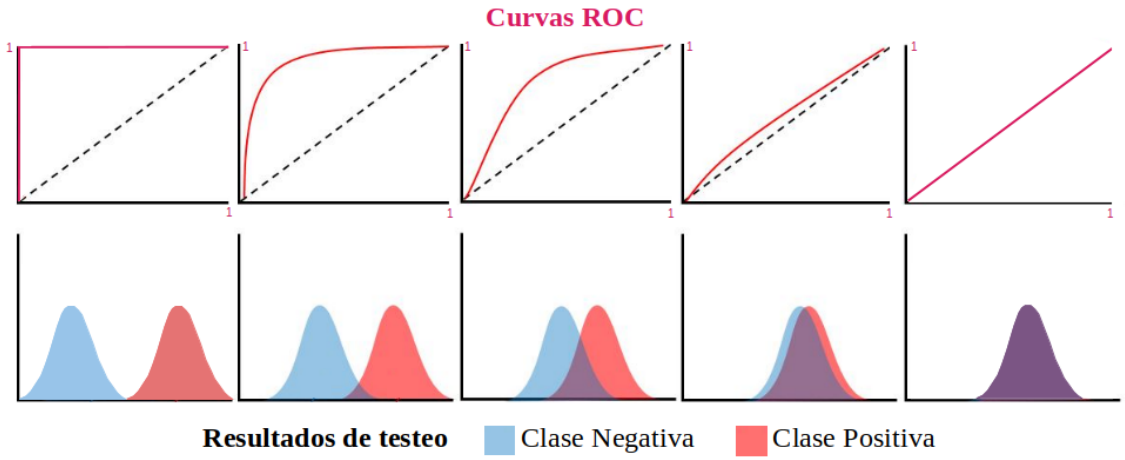

Figura 38: Variación de la forma de la curva ROC en base al cambio en la capacidad discriminativa de los resultados de testeo. En el primer caso, se pueden separar ambas curvas (resultados de la clase negativa y positiva), entonces la representación ROC alcanza el punto (0,1). Luego las curvas de resultados se empiezan a superponer, disminuyendo la capacidad discriminativa, esto repercute en la curva ROC que se acerca a la línea de no-discriminación. En el último ejemplo, existe una superposición total entre los resultados de ambas clases y la curva ROC equivale a la línea de no-discriminación. Imagen adaptada de [34].

El punto de la gráfica con mayor sensibilidad y especificidad conjunta, coincide con el que presenta un mayor valor del índice de Youden (Figura 39 A). Esta medida informa el rendimiento de la aplicación de un umbral y se calcula con la siguiente ecuación:

$$\textit{Índice de Youden} = \textit{Sensibilidad} + \textit{Especificidad} - 1$$

Gráficamente, el punto con mayor índice de Youden corresponde al punto de la curva ROC más cercano al ángulo superior-izquierdo de la gráfica, es decir, al punto (0,1), donde la sensibilidad y la especificidad alcanzan el valor máximo de 1 [33].

Por otra parte, a partir de la gráfica ROC se puede calcular el área bajo la curva (AUC). Esta métrica es una medida única que refleja la capacidad discriminativa entre clases, a lo largo de todo el rango de umbrales posibles (Figura 39 B); además es independiente de la prevalencia de la enfermedad en estudio [33]. Se provoca una mayor distinción entre clases, cuando el valor de AUC-ROC es más cercano a 1. En cambio, si la curva ROC coincide con la línea de no-discriminación, el AUC-ROC presenta una medida de 0,5.

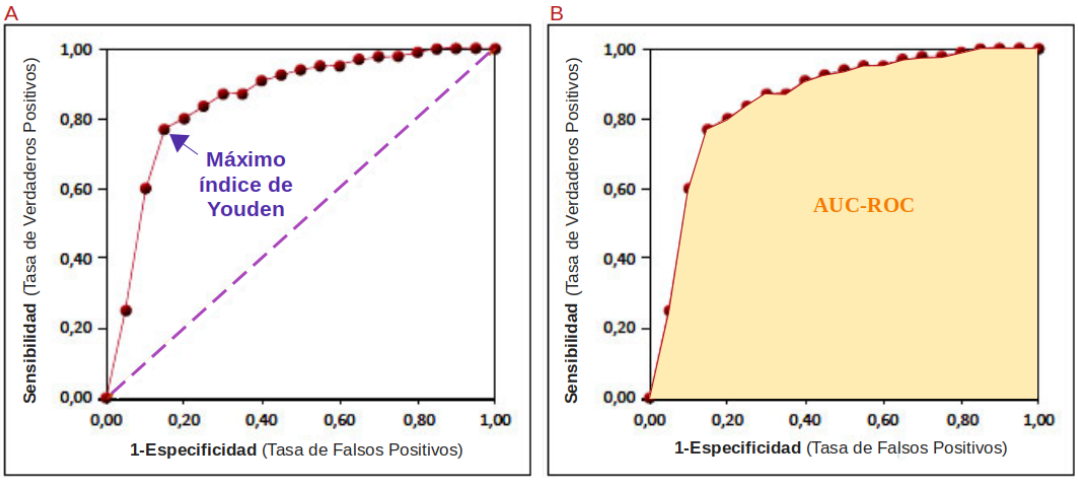

Figura 39: A: Identificación del punto de la curva ROC con máximo índice de Youden; se observa que corresponde al punto más cercano a la esquina superior izquierda. B: Esquematización del AUC de una curva ROC; un mayor valor de AUC-ROC equivale a una mayor separación entre la curva ROC y la línea de no-discriminación. Imagen adaptada de [33].

Otro gráfico utilizado para el análisis de resultados es la curva de Precisión-*Recall* (PR), que vincula dichas métricas de rendimiento y muestra sus resultados para un rango de umbrales de discriminación. Al relacionar esas métricas, la curva analiza el resultado de clasificación de la clase positiva. En consecuencia, es útil cuando se trabaja con una base de datos desbalanceada, donde la categoría positiva es la minoritaria; particularmente se recomiendan en casos con dominios muy sesgados, ya que la curva ROC puede proporcionar una visión optimista del rendimiento [35].

En la Figura 40 se muestra una curva PR: en el *eje X* se grafica el *Recall* (otra denominación de la métrica sensibilidad) y en el *eje Y*, la Precisión (o VPP); ambos ejes varían entre 0 y 1. Una curva más cercana a la esquina superior derecha, es decir, al punto (1,1), representa un mejor resultado ya que indica un mayor valor de ambas métricas. Además se visualiza en el gráfico una línea horizontal de referencia, que corresponde al resultado de un clasificador sin habilidad para distinguir las clases, cuya precisión equivale a la proporción de ejemplos positivos respecto al total de datos.

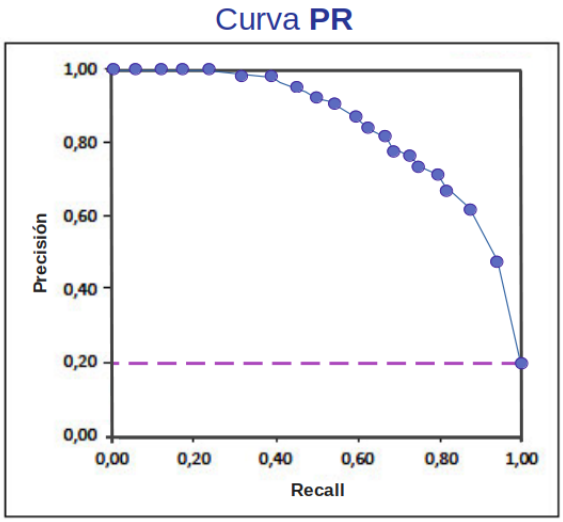

Figura 40: Representación de Curva *Precisión-Recall* (en azul). Cada punto del gráfico se corresponde con un umbral de discriminación distinto. La línea punteada violeta es la referencia, en base a un clasificador que asigna todos los casos como pertenecientes a la clase positiva. En este ejemplo, la base de datos se encuentra desbalanceada y posee un 20 % de observaciones positivas; entonces el clasificador de referencia tiene una precisión igual a 0,2.

Para la interpretación de la curva PR, también se analiza su AUC (Figura 41). Esta medida es independientemente de cualquier umbral y describe una métrica general de desempeño. Cuanto más se acerca el AUC-PR a la unidad, mejor es el rendimiento de clasificación de la clase positiva.

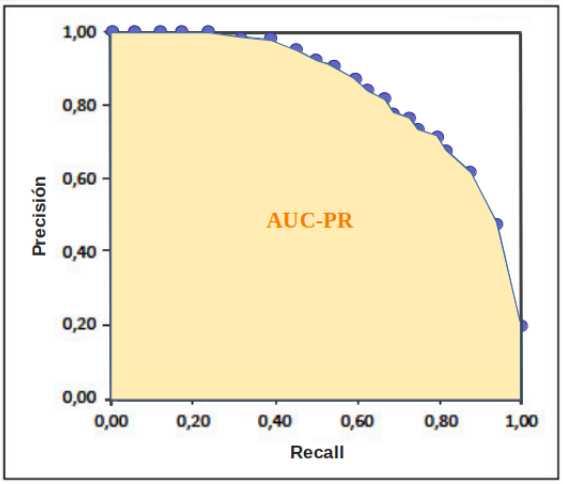

Figura 41: Señalización del AUC de la curva PR. Si la gráfica se acerca más al punto (1,1) se alcanza un mejor resultado. En consecuencia, el aumento de AUC-PR a un número cercano a 1, también significa una mejora en la clasificación de la clase positiva.

4.3. Tomografía computarizada con contraste

La tomografía computarizada es una técnica de imágenes médicas que permite obtener imágenes detalladas del interior del cuerpo del paciente, con fines de diagnóstico y de forma no invasiva. Como se ve en la Figura 42, el tomógrafo contiene un tubo de rayos X giratorio y al menos una fila de detectores, que logra medir las atenuaciones de los mismos en diferentes tejidos. Las múltiples mediciones de rayos X tomadas desde diferentes ángulos se procesan con algoritmos de reconstrucción y generan cortes axiales del cuerpo [36]. Con los cortes sucesivos se forma una imagen tridimensional del paciente, lo que facilita el entendimiento de la anatomía del paciente y la identificación de anormalidades.

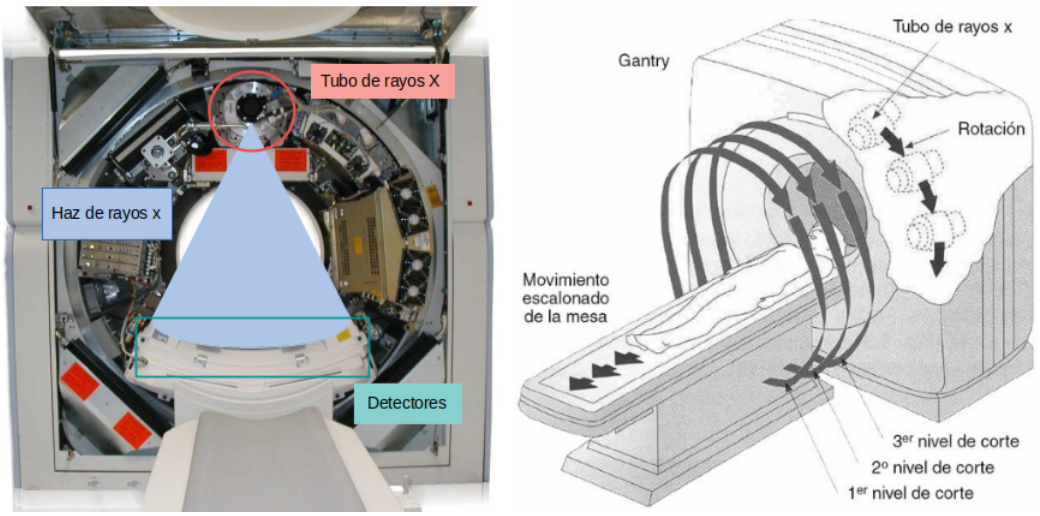

Figura 42: A la izquierda, el contenido interno del gantry del tomógrafo; se observa la posición enfrentada entre el tubo de rayos X y los receptores, lo que permite la medición de la atenuación de rayos X al atravesar el cuerpo del paciente. A la derecha, diagrama simplificado del funcionamiento del tomógrafo: dentro del gantry, el tubo de rayos X rota de manera coordinada con los receptores. El movimiento de la camilla permite que las emisiones atraviesen diferentes secciones transversales del paciente, generando los distintos cortes axiales luego del procesamiento de reconstrucción.

La urografía por tomografía computarizada permite examinar las vías urinarias, es decir, los riñones, la vejiga y los uréteres. Para realizar este estudio se le inyecta al paciente una solución de contraste de yodo en una vena de la mano o del brazo; dicha sustancia fluye en las vías urinarias y permite resaltar estas estructuras y delimitarlas de su entorno [37]. Las imágenes tomográficas se realizan en momentos específicos del examen médico, generando distintas fases de estudio:

- **Fase sin contraste (F_SC):** La adquisición de imágenes se realiza en una etapa pre-contraste. Sirve para identificar algunas anormalidades (por ejemplo cálculos renales) o para comparar el realce de lesiones en las otras fases.
- **Fase corticomedular (F_C):** Ocurre entre los 30 a 40 segundos luego de la inyección de contraste. Se resalta la corteza renal, ya que el flujo arterial se dirige en primer lugar a esta región y allí ocurre el proceso de filtración glomerular en las nefronas; en cambio la médula renal permanece menos remarcada [38]. En la Figura 43A se observa lo anteriormente descripto.
- **Fase nefrográfica (F_N):** Comienza dentro de los 80-120 segundos posteriores a la inyección de solución de yodo al paciente. Debido a la filtración tubular

en las nefronas, el medio de contraste se distribuye tanto en la corteza como en la médula renal, y produce el realce de ambas estructuras [38] (Figura 43B).

- **Fase excretora (F_E):** Esta última fase comienza entre 3 a 5 minutos después de su inyección de contraste, cuando el mismo se empieza a excretar a los cálices de los riñones. El marcado de la corteza y médula renal disminuye gradualmente con el tiempo, debido al decrecimiento de concentración plasmática de la solución de yodo en las nefronas [39]. Este cambio puede examinarse en entre las Figuras 43C y 43D. De forma rutinaria, se adquieren las imágenes de esta fase a los 4-5 minutos para asegurar el realce de los uréteres.

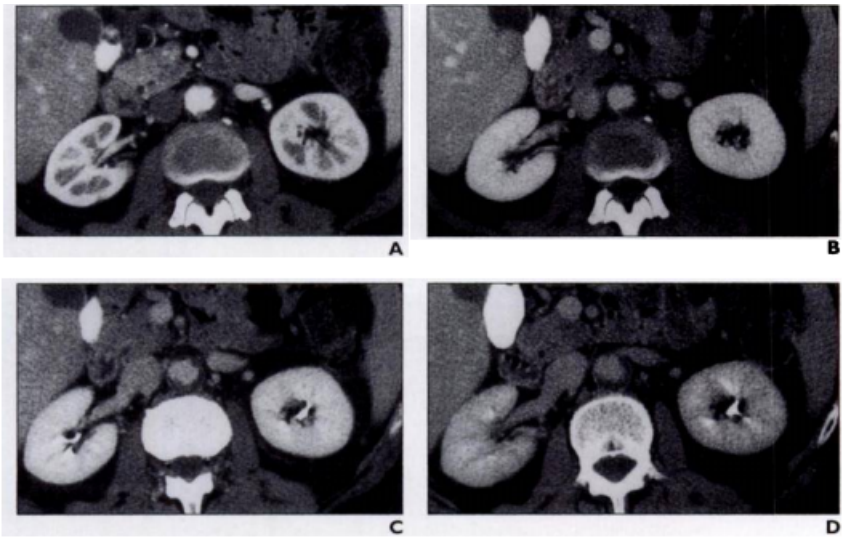

Figura 43: Se muestran 4 cortes axiales de tomografías computalizadas con contraste, tomadas en 3 fases distintas [39]. A: F_C, tomografía obtenida luego de 40 segundos de la inyección de contraste. Se remarca la corteza renal. B: F_N, tomografía adquirida a los 100 segundos de la inyección de contraste. Se distingue todo el parénquima renal. C-D: F_E, el primer corte es de una etapa temprana de la fase (150 seg.) mientras que el segundo es más tardío (300 seg.). Se muestra una disminución en el realce de la corteza y médula renal; también se marcan los cálices hacia donde se excreta el contraste.

Los tumores renales normalmente no presentan nefronas funcionales, por lo tanto, su realce en las imágenes tomográficas con contraste generalmente se debe a la vascularización [39]. Esta diferencia suele favorecer la distinción entre las masas tumorales y el tejido sano que lo rodea; siendo una mejora en comparación con la utilización de F_SC.

Sin embargo al utilizar la F_C, existen ocasiones en las que los tumores pueden confundirse con la médula renal no remarcada (Figura 44). Por otra parte, también

puede ocurrir la situación contraria en el caso de tumores hipervascularizados, donde su realce es mayor en esta primera fase vascular y se dificulta la diferenciación con la corteza del riñón (Figura 45). Estos problemas no suelen ocurrir en F_N y F_E, siendo F_N la opción más utilizada para el reconocimiento y segmentación de tumores.

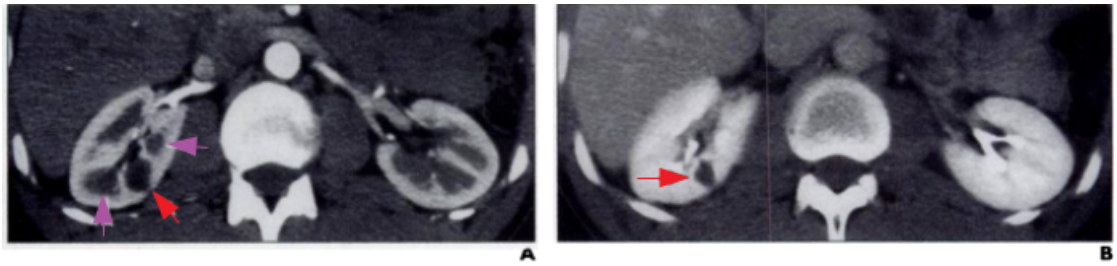

Figura 44: Limitación de imágenes de F_C para detección de tumores renales, debido a confusión con médula renal [39]. A: corte axial de tomografía de rinón en F_C; tumor (flecha roja) no es fácil de diferenciar de las secciones correspondientes a médula renal (flechas violetas) B: corte axial de tomografía de rinón en F_N; tumor (flecha roja) puede distinguirse y ser segmentado con facilidad.

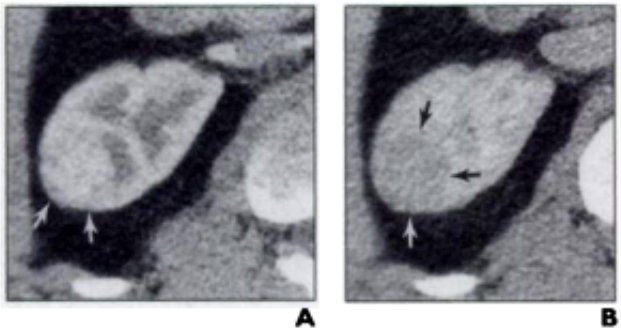

Figura 45: Limitación de imágenes de F_C para detección de tumores renales hipervascularizados, debido a confusión con corteza renal [39]. A: sección de corte axial de tomografía de rinón en F_C; tumor (flechas blancas) presenta una intensidad similar a la corteza renal en la imagen y se complica su diferenciación manual. B: sección de corte axial de tomografía de rinón en F_N; tumor (contorno marcado con flechas) puede distinguirse con mayor facilidad en esta fase.

4.4. Estado del arte

Las publicaciones sobre la clasificación de tumores renales utilizando CNNs se realizaron únicamente en años recientes, debido al aumento de la aplicación de *Deep Learning* en el campo médico. Las decisiones de manejo de la base de datos y

construcción de la red neuronal explicadas en estos trabajos fueron consideradas al momento de realizar las distintas pruebas en el proyecto. Además los resultados positivos indicaron la viabilidad del uso de CNNs para la clasificación de tumores sólidos. A continuación se explican brevemente estudios previos considerados relevantes porque poseen un objetivo similar al de este proyecto:

- ***Classification of renal tumour using convolutional neural networks to detect oncocytoma* - Pedersen, Andersen & Christiansen (2020) [40]:**

  Este estudio usa tomografías de 369 pacientes con tumores renales sólidos. Las neoplasias son ONCs o CCRs según el resultado de anatomía patológica luego de extracción o de una biopsia percutánea guiada por ultrasonido. Sólo 69 casos (19 %) corresponden a ONCs, mientras que los restantes 300 tumores son CCRs (81 %). En el grupo de entrenamiento, se implementa el balanceo de clases mediante un proceso de *oversampling* de los casos benignos.

  La base de datos se forma con cortes axiales de las tomografías, es decir, se utilizan imágenes de dos dimensiones; en consecuencia disponen de aproximadamente 20.000 datos obtenidos a partir de las 369 tomografías. Las dimensiones originales (512×512 píxeles) se modifican a 224x224 para adecuarse mejor a la red utilizada. La división en conjuntos de entrenamiento, validación y evaluación se realiza a nivel paciente, para evitar poseer imágenes similares en distintos grupos.

  En las imágenes no se segmentan los tumores, ya que se considera que la red neuronal puede identificar las características relevantes para la tarea de clasificación. Los autores no descartan una posible afección en el tejido circundante de los tumores malignos, que también podría ser utilizada por la CNN para la identificación de este tipo de tumores.

  Por otra parte, no se utiliza *data augmentation*, debido a que con su aplicación se observa un efecto negativo en las métricas de evaluación. Esto se vincula a la utilización de cortes de la tomografía en su totalidad, incluyendo estructuras anatómicas completas, lo que no permite la generación de imágenes rotadas o trasladadas similares a las originales.

  Como modelo se emplea la estructura ResNet50V2, modificando las capas de clasificación; los pesos son reentrenados para adaptarse mejor al problema tratado. La red neuronal se entrena por 10 *epochs*, utilizando la entropía cruzada binaria como función de costo y RMSprop como optimizador. Cabe mencionar que no se implementan métodos de regularización, ni *callbacks* para controlar el decaimiento del *learning rate*.

El estudio obtiene buenas predicciones para las imágenes del grupo de testeo (20 % de los datos totales): un *accuracy* del 93,3 %, una especificidad del 93,3 % y un AUC del 0.973. La clasificación del tumor por paciente se brinda mediante el voto mayoritario del 51 % de las categorizaciones de los cortes axiales pertenecientes a dicha neoplasia. En base a esta definición, el *accuracy* a nivel paciente alcanza el 100 % en el conjunto de testeo

- ***A multi-task convolutional neural network for renal tumor segmentation and classification using multi-phasic CT images* - Pan, Yang & Wang (2019) [41]:**

  El trabajo busca la clasificación entre tumores benignos (de tipo angiomiolipoma renal) y malignos (CCRs), utilizando imágenes tomográficas. Además se genera la segmentación de la neoplasia. Ambas tareas están asociadas mediante un proceso de retroalimentación en el entrenamiento: las métricas de clasificación generan un impacto positivo en los resultados de la segmentación y estos últimos brindan información de región de interés a la red de clasificación.

  La base de datos contiene 131 tomografías contrastadas de tumores (30 benignos y 101 malignos). En adición, posee las etiquetas de clase identificadas con anatomía patológica y la segmentación base demarcada por expertos del campo médico. Estas máscaras se usan únicamente para evaluar la segmentación automática.

  Se utilizan imágenes 2D correspondientes a cortes axiales de las tomografías, de tamaño 150x150. Se emplea *data augmentation*; se aplica más veces en los tumores benignos para balancear la base de datos.

  La estructura del modelo comienza con una CNN correspondiente a una ResNet50 modificada, después le sigue un módulo *pooling* piramidal que beneficia el aprendizaje de características en múltiples escalas. La ResNet50 es simplificada reduciendo el número de capas *pooling* y capas convolucionales con pasos mayores a 1, ya que el tamaño de las imágenes del estudio es menor a la entrada frecuente de la red pre-entrenada (224x224). El módulo *pooling* piramidal (Figura 46) presenta tres ramas donde se aplican diferentes disminuciones de las dimensiones de la imagen (con radios de 1/2, 1/4 y 1/8); cada camino contiene sus propias capas de convolución y, luego de un nuevo cambio de dimensiones, todas las salidas se unen en un único tensor. A partir de dicho módulo, el modelo diverge en dos bloques: uno de clasificación y el otro de segmentación.

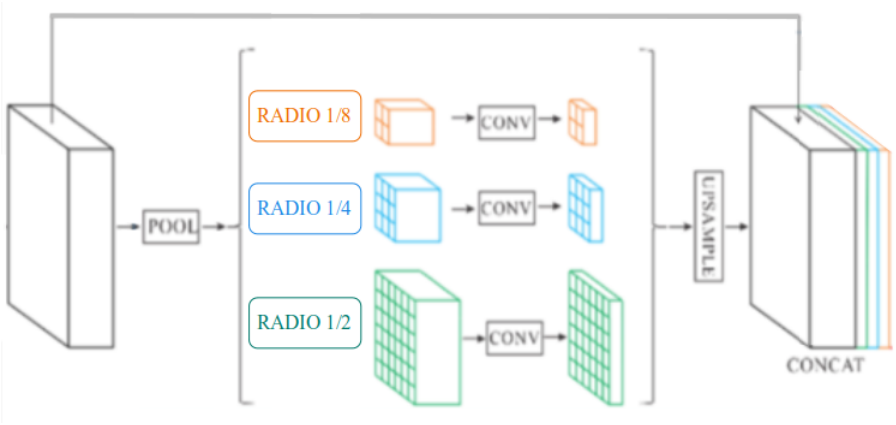

Figura 46: Representación del módulo *pooling* piramidal [42]. Mediante un proceso de *downsampling* con distintos radios, se generan tres ramas convolucionales. Luego se realiza *upsampling* para poder juntar las salidas de las 3 ramas en un único tensor. También se observa que el módulo contiene un atajo directo desde la entrada.

La función de costo general es la suma de la función de costo de la clasificación y de la segmentación; esta implementación puede prevenir el *overfitting*. La red se entrena por sólo 6 epochs y se utiliza validación cruzada para la evaluación de la clasificación.

En el grupo de testeo, con la segmentación obtenida para cada tumor, se determinan los cortes axiales de la tomografía que contienen la neoplasia. Con la clasificación de dichos cortes se calcula el voto mayoritario, lo que se identifica como la clasificación a nivel tumor. De esta manera se obtiene un *accuracy* del 100 %.

Por otra parte, se prueba únicamente el uso de la estructura ResNet50 para la clasificación de tumores, con el objetivo de comparar sus resultados con los obtenidos con el modelo planteado anteriormente. En este caso, se alcanza un *accuracy* del 85,1 % para las neoplasias malignas y del 90 % para los tumores benignos.

- ***Deep Learning for end-to-end kidney cancer diagnosis on multi-phase abdominal computed tomography* - Uhm, Jung, & Choi (2021) [43]:**

  El objetivo de la investigación es diferenciar cinco subtipos de tumores renales: tres tipos de neoplasias malignas (CCR de células claras, CCR papilar, CCR cromófobo) y dos tipos benignos (ONC y angiomiolipoma). También se identifican las lesiones en la tomografía computada sin requerir intervención manual.

Se dispone de tomografías con cuatro fases de contraste, de 308 pacientes con tumores renales; 50 casos son elegidos para conformar el grupo de testeo. Para la función de clasificación, se posee además las etiquetas que determinan el subtipo de cada neoplasia (adquiridas por anatomía patológica) y las categorizaciones de los tumores realizadas por seis radiólogos, para comparar con los resultados de esta tarea . Para el aprendizaje de la segmentación, se usan como referencia máscaras del riñón y del tumor, obtenidas manualmente por expertos.

El desarrollo del estudio tiene tres componentes principales, como se muestra en la Figura 47. En primer lugar la segmentación del riñón y del tumor, luego la registración entre distintas fases pertenecientes a un mismo paciente, y por último la clasificación en subtipos de tumor renal.

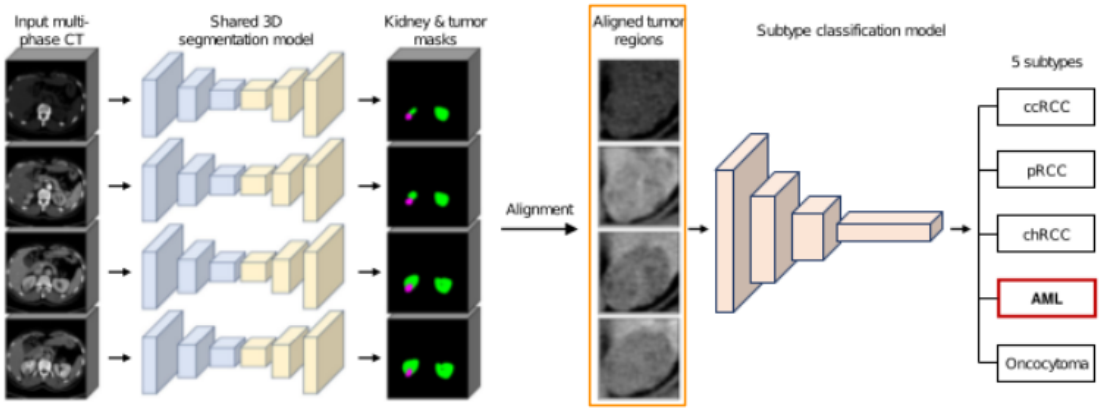

Figura 47: Esquema de trabajo aplicado en la investigación [43]. A partir de tomografías con distintas fases de contraste, se segmenta los riñones (verde) y el tumor (magenta) usando una CNN con tres dimensiones. Luego de un proceso de alineamiento, se extrae el corte axial mayor de cada fase del tumor (recuadro naranja). Con estas secciones se forman imágenes de tres canales que se utilizan como entrada de la CCN de dos dimensiones que se encarga de la tarea de clasificación. Los cinco subtipos son: CCR de células claras (ccRCC), CCR papilar (pRCC), CCR cromófobo (chRCC), angiomiolipoma (AML) y ONC.

La segmentación se realiza utilizando una CNN de tres dimensiones, cuyas entradas son las tomografías enteras y las máscaras de referencia; cada vóxel se clasifica como tumor, riñón o fondo de la imagen. En base a las regiones segmentadas, se aplica la registración entre pares de fases de un tumor (usando la F_N como referencia). Se emplean redes que modifican los parámetros de la matriz de afinidad iterativamente hasta la convergencia.

Para la clasificación se crea una base de datos conformada por imágenes de tres canales; a continuación se explica la metodología seguida para la obtención de matrices a partir de un tumor. Primero, para cada fase, se selecciona el corte

axial con mayor región segmentada. Entonces, si se poseen las cuatro fases de contraste para ese paciente, se extraen cuatro cortes completos de las tomografías. Luego se delimitan los planos a un contorno rectangular que encierra únicamente la región de la neoplasia. La salida de esta etapa, se muestra en el esquema de la Figura 47: las matrices de dos dimensiones (recuadro naranja) corresponden a recortes de la región de interés de tomografías de F_SC, F_C, F_N y F_E de un tumor.

Por último, se modifica el tamaño de estas imágenes a 224x224 y se unen en matrices de tres canales. Si las cuatro fases del paciente están presentes se forman tres imágenes: siempre se incluye la F_SC y se excluye una fase contrastada (F_C, F_N o F_E). Si hay una fase faltante, con las otras tres se produce una única imagen. Los casos con menos de tres fases son excluidos. Las imágenes conseguidas son las entradas de la red de clasificación, junto con las correspondientes etiquetas de los tumores.

Se emplea la red neuronal ResNet-101, usando los pesos pre-entrenados con *ImageNet* y modificando la última capa densa para obtener como salida la probabilidad de pertenencia del tumor a los cinco subtipos. Para el entrenamiento se utiliza entropía cruzada como función de costo y descenso de gradiente estocástico como optimizador.

Los resultados del grupo de testeo se adquieren al promediar las probabilidades predecidas para todas las imágenes de un mismo tumor. El modelo alcanza un *accuracy* del 72 % y un AUC promedio de 0.889; ambos resultados son mejores, en la mayoría de los subtipos, a los obtenidos por los radiólogos .

5. Materiales y métodos

La utilización de redes neuronales es un proceso experimental, donde deben probarse múltiples configuraciones sobre la implementación del preprocesamiento a partir de las imágenes tomográficas, los criterios de selección de datos de la red, la estructura del modelo y las condiciones de su entrenamiento, los métodos y métricas de evaluación del grupo de testeo, entre otros factores. En esta sección se describen todos los métodos evaluados durante el desarrollo del trabajo.

5.1. Base de datos

El Servicio de Diagnóstico por Imágenes del Hospital Italiano de Buenos Aires junto con el Departamento de Informática en Salud, iniciaron un proyecto para producir una herramienta de inteligencia artificial que asista en la diferenciación de tumores renales sólidos, con el objetivo de incorporarla en el flujo de trabajo de la institución, para ser utilizada como soporte a la toma de decisión en el diagnóstico entre CCRs y ONCs. Este centro de salud brindó la base de datos necesaria para el entrenamiento y prueba de las CNNs [8].

En este caso de estudio, la base de datos necesaria para el entrenamiento de la red por aprendizaje supervisado incluye los estudios anonimizados de tomografía computarizada con contraste, las máscaras tumorales segmentadas por radiólogos y las etiquetas que indican el tipo de tumor según el resultado de anatomía patológica (ONC o CCR).

Entonces, el primer paso realizado por el Servicio de Diagnóstico por Imágenes para la creación de la base de datos fue la búsqueda retrospectiva de registros institucionales. Se seleccionaron casos de pacientes mayores de 18 años que se sometieron a resección quirúrgica del tumor en el Hospital Italiano de Buenos Aires, entre enero de 2015 hasta diciembre de 2017, y que poseen el resultado de anatomía patológica confirmado a partir del tumor extraído. De este grupo, se conservaron los pacientes que realizaron una tomografía computarizada con contraste previa a la operación, en la institución y con un protocolo estándar.

En segundo lugar, se adquirieron las máscaras tumorales: un grupo de radiólogos experimentados efectuó la segmentación volumétrica manual de cada masa a partir de la F_N de la tomografía, empleando 3D Slicer [45] (programa de uso libre y gratuito). Este tipo de segmentación se consigue iterando entre los planos axiales (perpendiculares al eje cráneocaudal) de una tomografía y realizando el contorneado del perímetro del tumor en cada plano donde se encuentra [46] (Figura 48). La región resultante se representa como un mapa de etiquetas binario, donde un vóxel de la máscara posee valor unitario si pertenece al tumor, y valor nulo en caso contrario.

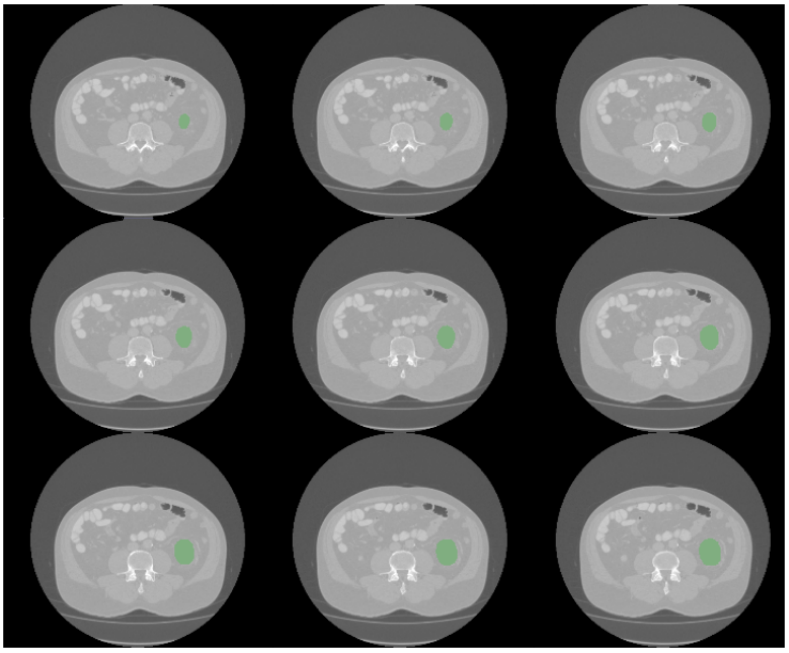

Figura 48: Máscaras segmentadas manualmente por un radiólogo (sección verde) en cortes axiales consecutivos donde se sitúa el tumor. Se observa que la superficie de la neoplasia en cada plano axial es distinta. La segmentación fue realizada con el programa 3D Slicer; este ejemplo corresponde a la F_N de la masa CCR_2 [8].

Por otra parte, las máscaras correspondientes a las otras fases de contraste, se obtuvieron a partir de la máscara segmentada en la F_N. Con el paquete Nibabel [47] se realizó la transformación de dicha máscara al sistema de coordenadas de cada fase tomográfica restante (F_SC, F_C y F_E). Luego, se visualizó si la delimitación de la máscara correspondía con el volumen del tumor en la nueva fase; en caso negativo, se segmentó la masa a partir de esta fase. Se aplicó este método debido a que la segmentación manual en todas las fases consume mucho tiempo a los especialistas y en ocasiones no es sencilla, por ejemplo para determinados tumores en F_SC o F_C como fue explicado anteriormente. En la Figura 49 se esquematiza el proceso de adquisición de los datos.

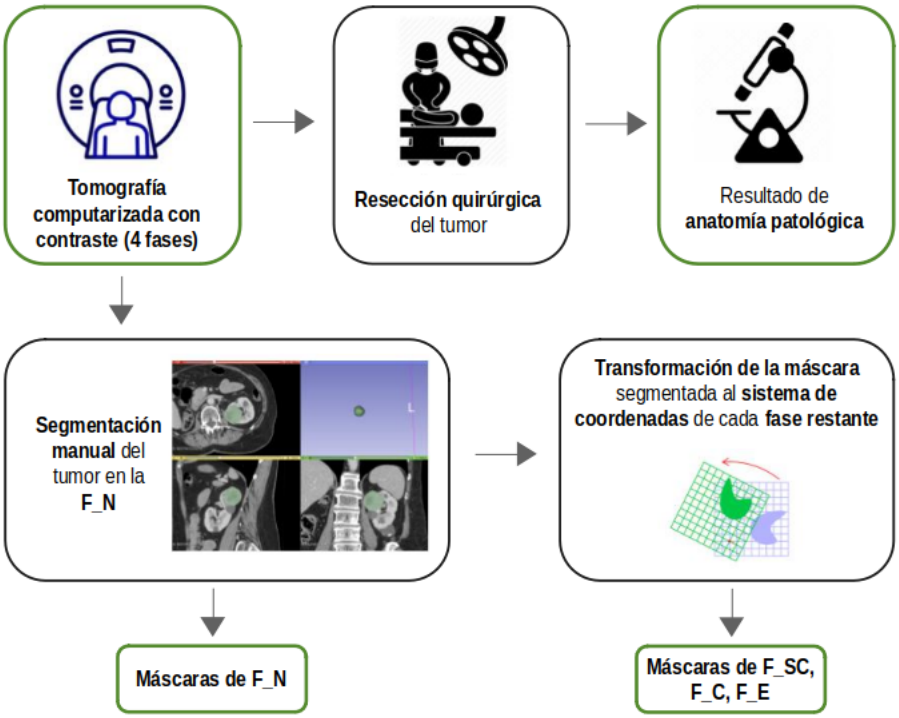

Figura 49: Esquema de obtención de los datos para el entrenamiento de la red. En verde se remarcan los elementos que conforman la base de datos: las cuatro fases de la tomografía (F_SC, F_C, F_N y F_E) con sus correspondientes máscaras y el resultado de anatomía patológica del tumor extraído.

De esta manera, el Hospital Italiano de Buenos Aires generó una base de datos que incluye tomografías de 165 pacientes con tumores renales: 109 masas corresponden a CCRs y las otras 56 son ONCs. La mayoría presenta las cuatro fases de la tomografía con contraste (F_SC, F_C, F_N y F_E); sin embargo, hay casos donde no se adquirieron todas las fases, debido a condiciones particulares de los estudios por tomografía computarizada. Cabe aclarar que para los pacientes cuya fase faltante es la F_N, la segmentación manual se realiza en otra fase, generalmente F_E.

Cabe aclarar que existen 7 pacientes con tumores bilaterales (3 con CCRs y 4 con ONCs), es decir, con masas tumorales en ambos riñones. En estas situaciones cada tumor se considera como un caso separado.

Tanto las tomografías como las máscaras fueron brindadas en formato NIfTI (*Neuroimaging Informatics Technology Initiative*) [48]; de esta manera los estudios médicos son anónimos y se respeta la privacidad de los pacientes. Se realizó la división de la base de datos en tres grupos para la aplicación de redes neuronales; esta separación se implementó a nivel paciente, para evitar poseer información similar entre los grupos, y de manera aleatoria:

- **Datos de entrenamiento:** 70 % de la base de datos, es decir, 115 tumores.
- **Datos de validación:** 10 % de la base de datos, 17 tumores.
- **Datos de testeo:** 20 % de la base de datos, 33 tumores.

La base de datos se encuentra desbalanceada, presentando un 66,06 % de CCRs y un 33,94 % de ONCs. Cada grupo de trabajo presenta un desbalance similar para que sean representaciones significativas; en la Tabla 2 se muestra la cantidad de tumores de cada tipo presentes en los grupos.

Datos	**CCR**	**ONC**	**Total tumores**
Entrenamiento	76 (66,09 %)	39 (33,91 %)	115
Validación	11 (64,71 %)	6 (35,29 %)	17
Testeo	22 (66,67 %)	11 (33,33 %)	33

Tabla 2: Cantidad (y porcentaje) de CCRs y ONCs presentes en cada grupo de trabajo.

En la Tabla 3, se exponen características descriptivas para ambos tipos de tumor de cada grupo de trabajo. Se menciona la edad (informando la mediana y el rango intercuartil) y el sexo (su frecuencia y porcentaje) de los pacientes de cada conjunto. Además se muestra una métrica referida al tamaño del tumor: la medida del *eje mayor de la masa*, es decir, se cuantifica la dimensión del mayor segmento del tumor coincidente con un eje anatómico.

Entrenamiento		
	CCR	*ONC*
Edad	64 años (56,75; 70)años	69 años (61,5; 75)años
Sexo	M: 26 (34,21 %) F: 18 (23,68 %) Sin información: 32 (42,10 %)	M: 25 (64,10 %) F: 14 (35,90 %)
Eje mayor del tumor	33,04mm (22,32; 42,91)mm $\leq$ 35mm: 40 (52,63 %) >35mm: 36 (47,37 %)	30,71mm (22,88; 37,99)mm $\leq$ 35mm: 23 (58,97 %) >35mm: 16 (41,03 %)
Validación		
	CCR	*ONC*
Edad	74 años (65,5; 76)años	71 años (66,25; 78,75)años
Sexo	M: 8 (72,72 %) F: 3 (27,27 %)	M: 3 (50,00 %) F: 3 (50,00 %)
Eje mayor del tumor	18,29mm (16,07; 24,98)mm $\leq$ 35mm: 9 (81,82 %) >35mm: 2 (18,18 %)	41,08mm (35,55; 56,06)mm $\leq$ 35mm: 2 (33,33 %) >35mm: 4 (66,67 %)
Testeo		
	CCR	*ONC*
Edad	61 años (52; 69)años	71 años (62; 75)años
Sexo	M: 18 (81,82 %) F: 4 (18,18 %)	M: 5 (45,45 %) F: 3 (27,27 %) Sin información: 3 (27,27 %)
Eje mayor del tumor	27,14mm (20,64; 32,63)mm $\leq$ 35mm: 18 (81,82 %) >35mm: 4 (18,18 %)	34,37mm (29,70; 42,87)mm $\leq$ 35mm: 7 (63,63 %) >35mm: 4 (36,36 %)

Tabla 3: Características descriptivas de ambos tipos de tumor de cada grupo de trabajo. La edad de los pacientes de cada subgrupo se representa con la mediana y el rango intercuartil. Para la variable sexo se contabiliza la cantidad de individuos masculinos (M) y femeninos (F) y se calculan sus respectivos porcentajes; cabe aclarar la falta de información sobre algunos pacientes con CCRs del grupo de entrenamiento y con ONCs del conjunto de testeo. Por último, se analiza la métrica *eje mayor del tumor* de dos formas: con la mediana y el rango intercuartil y mediante el registro de la cantidad de tumores con medidas inferiores o superiores al umbral de 35 milímetros.

5.2. Preprocesamiento

Se crearon distintos métodos de preprocesamiento para generar las entradas de la red. Se planteó tanto el uso de cortes axiales de la tomografía en dos dimensiones (2D) como la utilización de información en tres dimensiones (3D).

El empleo de cortes 2D presenta la ventaja de aumentar el número de observaciones, cuestión fundamental para el entrenamiento de las redes neuronales: mientras que una imagen 3D equivale sólo a una muestra, varios cortes 2D pueden extraerse del mismo. La inclinación inicial del proyecto fue utilizar matrices 2D, siguiendo la línea de trabajos previos que aplican CNNs para la clasificación de tumores renales sólidos [40]-[41] [43]-[44].

Todos los métodos de preprocesamiento se aplicaron a la base de datos inicial de archivos NIfTI; se ejecutaron con código en lenguaje *Python* de forma local, usando *Linux* como sistema operativo. Luego los resultados del preprocesamiento se subieron a *Drive* para poder acceder a esta información desde *Colaboratory*.

Cabe destacar que no se aplicaron técnicas de mejoramiento como eliminación de ruido o cambios de contraste y brillo, ya que las mismas se aplican de manera automática en las capas de la red convolucional, a partir de transformaciones de la imagen mediante la convolución con distintos filtros. Con el aprendizaje de la red neuronal se halla una combinación de dichas transformaciones que optimiza la tarea a realizar, en consecuencia no es necesario aplicar las técnicas de mejoramiento como parte del preprocesamiento de imágenes.

5.2.1. Obtención de datos en 2D

Los extremos del *eje Z* que contiene el tumor se determinan en base a la máscara segmentada. Para cada nivel del *eje Z* entre dichos límites, se extraen cortes axiales del tumor (Figura 50). De esta manera se obtienen 17.365 cortes 2D correspondientes al grupo de entrenamiento, 1.806 del conjunto de validación y 3.911 del grupo de testeo.

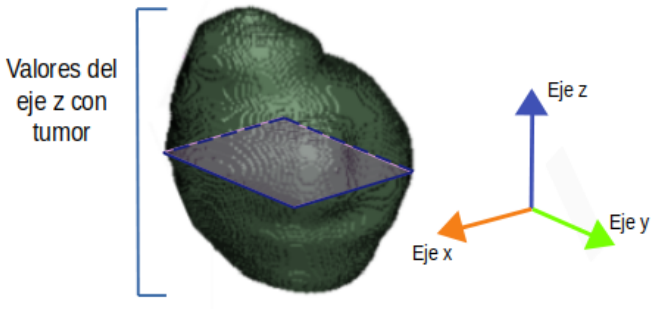

Figura 50: Representación del volumen de la máscara del tumor CCR_2 [8], realizada en el programa 3D Slicer. Para cada valor del *eje Z* que contiene información del tumor, se obtiene una sección axial del mismo. A la derecha se indica la orientación de los ejes. Aclaración: el *eje X* corresponde al eje laterolateral, el *eje Y* al eje anteroposterior y el *eje Z* al eje cráneocaudal.

Mientras que las entradas de la red neuronal son matrices rectangulares, las máscaras segmentadas son contornos heterogéneos debido a que imitan la forma del tumor. Por lo tanto, en el corte axial queda información que no corresponde a la neoplasia. Esta situación genera la primera decisión de diseño: disponer un fondo negro alrededor del corte del tumor o conservar los píxeles del tejido circundante.

5.2.1.1. Métodos sin tejido circundante

Las máscaras obtenidas manualmente permiten conocer los límites del tumor. La superposición de cada máscara con la tomografía que le corresponde, crea un tensor con la información tomográfica únicamente del tumor, mientras que al resto de los píxeles se les asigna valor nulo (Figura 51). Se considera entonces sólo la información de la neoplasia y no la de su entorno, descartando la importancia del mismo para la clasificación entre ONC y CCR.

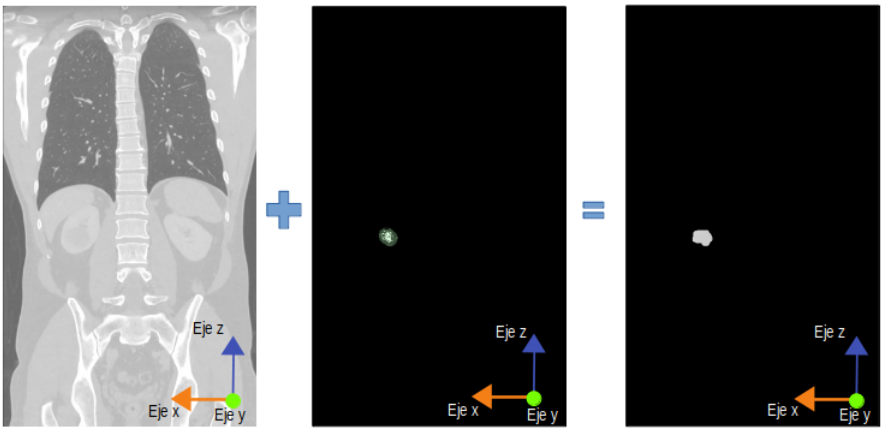

Figura 51: A la derecha, resultado de la superposición de una tomografía (imagen izquierda) y la máscara del tumor (imagen central); sólo se conserva la información tomográfica del tumor. Las imágenes de esta figura corresponden al paciente que contiene el CCR_31 en la base de datos del Hospital Italiano [8].

Los extremos del *eje Z* se demarcan al considerar las posiciones de dicho eje que contengan valores no nulos en la máscara. Por otra parte, los límites de cada plano 2D de un tumor se pueden definir de dos maneras diferentes según la metodología a aplicar.

- **Contorno por tumor:** En base a la máscara segmentada, se calculan las dimensiones mayores que posee el tumor en el *eje X* y el *eje Y*, a lo largo de todo el *eje Z*. A partir de estos valores se reconocen los extremos laterales del tumor que delimitan todos los cortes 2D obtenidos a lo largo del *eje Z* (Figura 52).

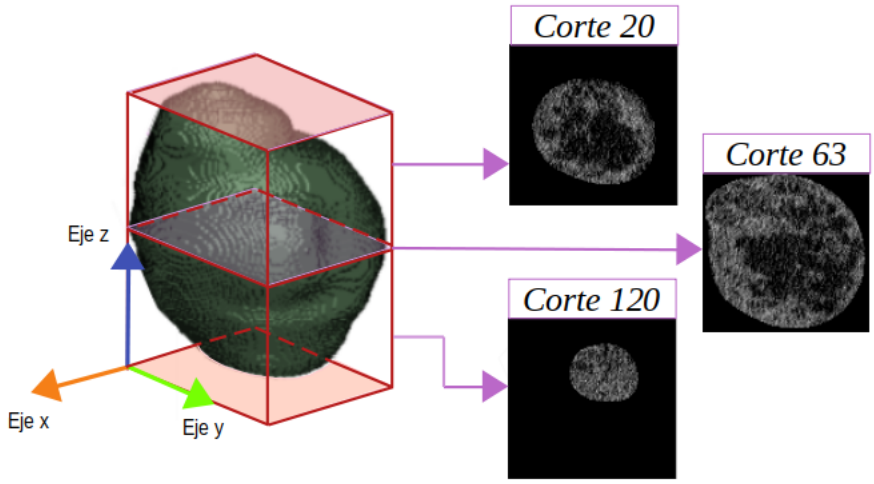

Figura 52: En la representación del volumen de la máscara del tumor CCR_2 [8], se marca el corte con mayores dimensiones en el *eje X e Y* (corte central coloreado) y los límites del tumor en el *eje Z*. Entonces todos los cortes 2D de este tumor, obtenidos por esta metodología, son secciones axiales del prisma rojo representado en esta figura.

De esta manera todos los cortes de un mismo tumor presentan la misma dimensión y la proporción entre píxeles con información del tumor y píxeles nulos varia dependiendo de la posición del corte respecto al centro del tumor (al alejarse del centro aumenta la cantidad de valores nulos). En la Figura 53 se muestran dos cortes diferentes de un mismo tumor y se pueden distinguir las características mencionadas.

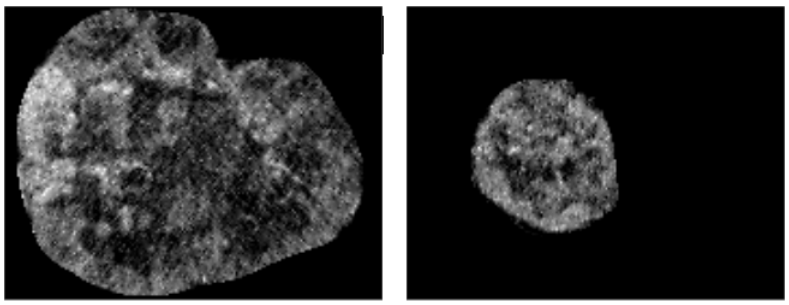

Figura 53: Cortes del tumor CCR_1 en F_N [8] (a la izquierda: corte número 73, a la derecha: corte número 22), obtenidos con el método *Contorno por tumor*. El primer corte presenta las dimensiones mayores de todo el tumor en *eje Y*, entonces es utilizado como referencia para establecer las dimensiones de los cortes obtenidos con esta metodología. Ambos cortes presentan iguales dimensiones pero el corte de la derecha contiene mayor cantidad de valores nulos al encontrarse más cercano al extremo del tumor.

- **Contorno por corte:** Se calculan los límites de la máscara en los *ejes X e Y*, para cada plano perpendicular al *eje Z*; en base a estos valores se recorta el tensor para obtener el corte 2D. Por lo tanto, los bordes de cada imagen coinciden con los límites del tumor en el plano que le corresponde. En la Figura 54 se esquematiza este procedimiento.

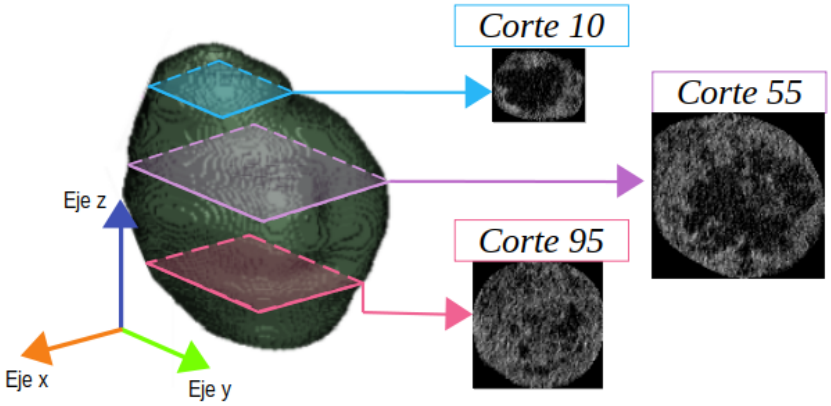

Figura 54: En la representación del volumen de la máscara del tumor CCR_2 [8], se demarcan 3 cortes distintos del tumor. Se observa que cada sección presenta diferente área, ya que la delimitación varía según la posición del *eje Z*.

Entonces, cada corte de un mismo tumor presenta distinta dimensión (Figura 55). Así se producen entradas 2D del modelo que contienen menos cantidad de píxeles negros que las matrices obtenidas en la metodología anterior (Figura 56).

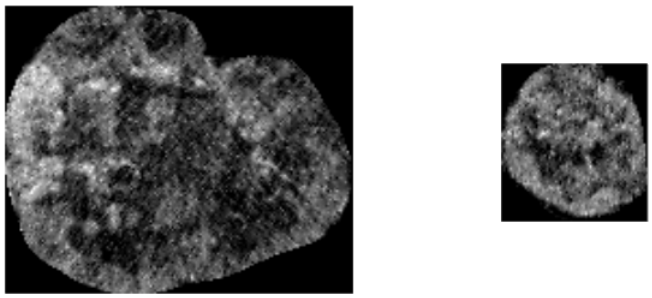

Figura 55: Cortes del tumor CCR_1 en la F_N [8] (a la izquierda: corte número 73, a la derecha: corte número 22), obtenidos con el método *Contorno por corte*. Se observa que cada corte es recortado en base a los límites del tumor en cada plano; cada corte 2D presenta distintas dimensiones.

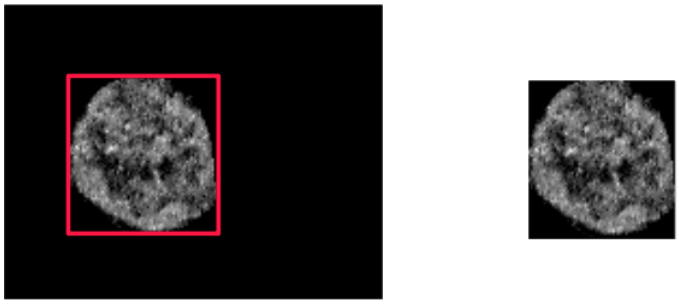

Figura 56: Comparación entre el uso de diferentes métodos de preprocesamiento, aplicado al corte 22 del tumor CCR_1 en la F_N [8]. Se observa que el resultado del preprocesamiento *Contorno por corte* (a la derecha) equivale a un recorte de la imagen obtenida por *Contorno por tumor* (a la izquierda) según los bordes del corte axial del tumor (recuadro rojo).

5.2.1.2. Métodos con tejido circundante

Además del tumor, los cortes 2D adquiridos incluyen píxeles de los tejidos que rodean al tumor. Esta configuración brinda información a la red sobre el entorno, lo que podría ser importante si el tejido del mismo se viese afectado de manera distintiva por la condición cancerígena del tumor [40].

- **Entorno por corte:** En base a las dimensiones de la neoplasia en cada plano del *eje Z*, se calcula el radio del tumor en dicho corte y se agrega un entorno rectangular de esa medida. Por consiguiente, cada corte 2D tiene distintas dimensiones según el tamaño del tumor, como ocurre en el preprocesamiento de *Contorno por corte*.

  En la Figura 57 se esquematiza la obtención de un corte en base al radio del tumor y en la Figura 58 se comparan distintos cortes de un mismo tumor.

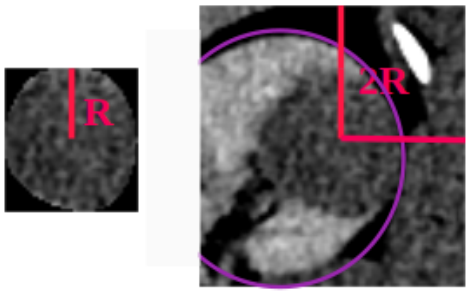

Figura 57: A la izquierda, corte 26 del tumor CCR_31 [8] representado según el preprocesamiento *Contorno por corte*; marcado con una línea (R) se indica el radio mayor del tumor en este plano. A la derecha se observa el mismo corte según la representación con contorno (*Entorno por corte*); en los *ejes X e Y* se le agrega, a ambos lados del tumor, una parte del entorno de dimensión igual al radio. Además puede observarse en la imagen de la derecha que el entorno abarca la mayoría del riñón (marcado en violeta) donde se encuentra el tumor.

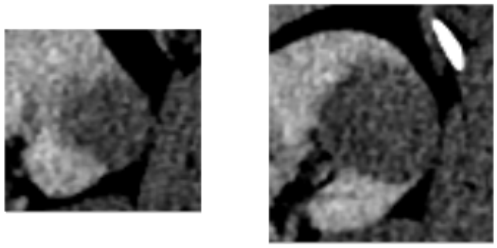

Figura 58: Cortes del tumor CCR_31 en la F_N [8] (a la izquierda: corte número 9, a la derecha: corte número 26), obtenidos con el método *Entorno por corte*. Se puede ver que el entorno tomado se incrementa al aumentar el tamaño del tumor en el corte; entonces cada corte 2D presenta distintas dimensiones. También se observa la facilidad de distinción del tumor en la F_N, debido a la diferencia de intensidad respecto a la corteza del riñón (que se ve más resaltada por efecto del contraste).

- **Entorno total:** Se toman los cortes axiales de tomografía en su totalidad, es decir, en su dimensión original, para valores de *eje Z* que presentan la máscara del tumor. Una de las diferencias con respecto a los otros métodos explicados es que brinda la ubicación del tumor respecto a la anatomía del cuerpo del paciente.

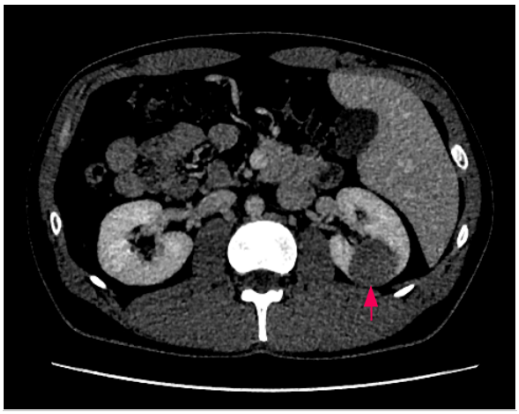

Figura 59: Corte 26 del tumor CCR_31 en la F_N [8], obtenido con el método *Entorno total*. Se observa el tumor en el riñón derecho (flecha rosa), pero se brinda mucha información innecesaria para la tarea de clasificación del tumor renal.

Finalmente no se utilizaron los cortes obtenidos con esta metodología para el entrenamiento de la red, ya que se consideró que se necesitaría un mayor poder de procesamiento y se desaprovecharía la segmentación manual presente en la base de datos. Además el aumento de datos se ve afectado debido al cambio de posicionamiento de la estructura anatómica del paciente, al aplicar transformaciones como desplazamiento o rotación de las imágenes [40].

5.2.1.3. Exclusión de datos con artefactos de cuadriculado

Es importante mencionar que todas las metodologías de obtención de datos 2D, sin y con entorno circundante, excluyen los cortes provenientes de tumores cuyas máscaras presentan un error exportado del programa 3D Slicer. Estas máscaras poseen un cuadriculado con valores nulos que al ser superpuesto con la tomografía genera un tumor con el mismo error (Figura 60).

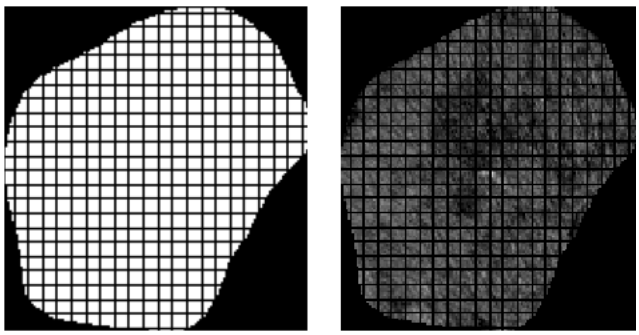

Figura 60: La máscara del CCR_4 en F_N [8] presenta un error que genera una cuadrícula en todos los cortes de este tumor al superponer la máscara con la tomografía del paciente. A la izquierda: corte 33 de la máscara segmentada con 3D Slicer. A la derecha: corte 33 del tumor CCR_4 en F_N [8] adquirido con el preprocesamiento *Contorno por corte*.

Dichos casos fueron detectados manualmente y descartados de la base de datos para no generar un sesgo de clasificación debido a la falla. Por ejemplo si la mayoría de los tumores cuadriculados son CCRs, la red puede aprender esta característica como propia de esta clase de neoplasia, lo que sería incorrecto.

La exclusión de cortes 2D con error provoca la eliminación de 763 imágenes del grupo de entrenamiento, 50 del conjunto de validación y 126 del grupo de testeo. Entonces la base de datos se conforma con 16.602 cortes 2D para el entrenamiento, 1.756 de validación y 3.785 en el grupo de testeo.

5.2.2. Obtención de datos en 3D

Otra metodología de trabajo es el entrenamiento de modelos en tres dimensiones. De esta manera se incluyen las relaciones entre vóxeles de distintos cortes del *eje Z*, información que se pierde al utilizar una base de datos 2D. Con el preprocesamiento en 3D, el grupo de entrenamiento posee 422 imágenes 3D, el conjunto de validación 62 y el de testeo 116.

En este caso los datos 3D se obtienen superponiendo la tomografía de cada tumor con su correspondiente máscara y recortando en base a las dimensiones mayores del tumor en cada eje. La información de cada tumor se guarda como matriz de numpy (usando la biblioteca NumPy [49]), conservando el rango de intensidad de la escala de Hounsfield de la tomografía, es decir, valores enteros entre -1024 y 3071.

Cada tumor presenta diferente cantidad de cortes, por lo tanto la dimensión del *eje Z* difiere entre las entradas de la red. Para unificar este valor, se elige un número de cortes adecuado y se agregan o eliminan capas de los tumores según el caso: si la neoplasia presenta menor cantidad de vóxeles en este eje, se agregan capas negras para alcanzar la dimensión deseada; en cambio si el tumor contiene una mayor cantidad de cortes, se descartan los extremos del tumor, conservando sólo las capas intermedias del mismo. En la Figura 61, se esquematizan ambas situaciones.

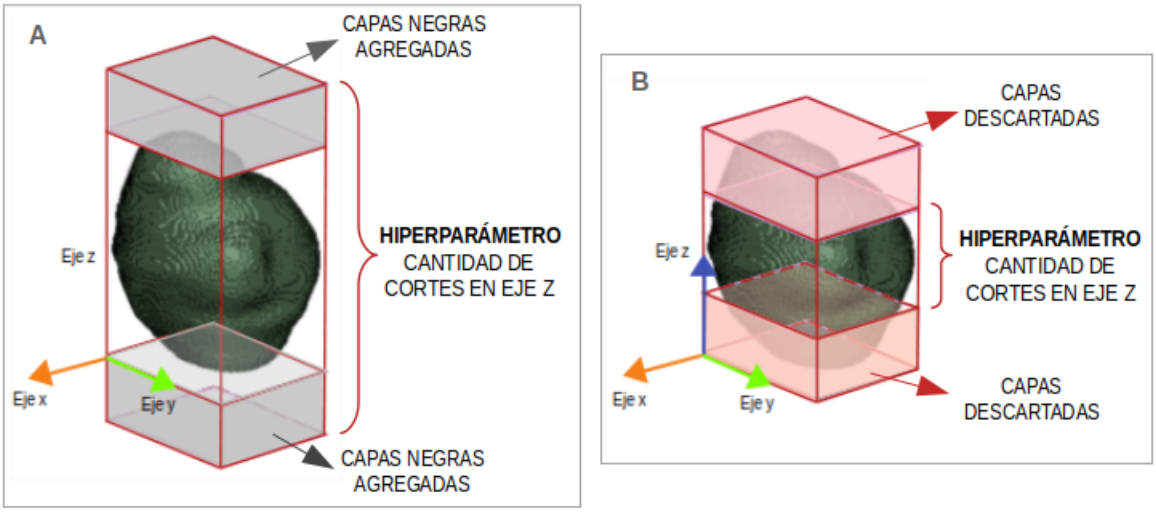

Figura 61: En verde se ve la representación del volumen de la máscara del tumor CCR_2 [8], que contiene 128 capas en el *eje Z*. Para estandarizar el tamaño de los tensores 3D de entrada de la red, se define el hiperparámetro de cantidad de cortes en este eje. A: Para CCR_2, si el hiperparámetro es mayor a 128, deben agregarse capas negras luego de ambos extremos del tumor. B: En cambio, si el valor es inferior a 128, se descartan capas de los extremos y se conservan la cantidad de cortes centrales que indica el hiperparámetro.

En el preprocesamiento 3D, también se trata la falla de cuadriculado presente en algunas máscaras de los tumores. Estas máscaras poseen valores nulos formando una cuadrícula dentro de la sección segmentada, lo que se traslada a la obtención de imágenes tomográficas de la masa tumoral con el mismo error. En los cortes 2D, el cuadriculado se observa en los *ejes X e Y*; en los cortes 3D existe además faltante de cortes en el *eje Z* (Figura 62).

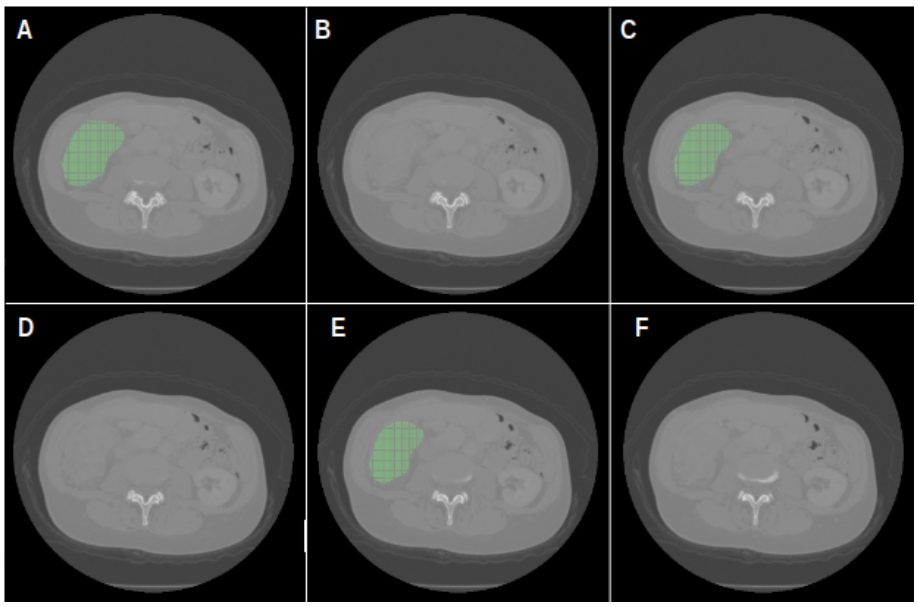

Figura 62: Planos axiales consecutivos (desde A hasta F) de la tomografía en F_N del paciente con tumor CCR_4 [8], visualizados con 3D Slicer. En verde se marca la máscara del tumor. En A, C y E: se observa el cuadriculado de la máscara en los *ejes X e Y*. En B, D y F: la falta de máscara indica la falla de cuadriculado en el *eje Z*.

Al utilizar la metodología 3D se consigue una menor cantidad de datos en comparación con métodos 2D, debido a que cada tumor equivale solamente a un tensor 3D por fase tomográfica. Entonces se decidió arreglar las máscaras para no eliminar información: se aplicaron operaciones morfológicas de cierre con filtros 3D para solucionar el error de cuadriculado. En la Figura 63, se muestra un corte axial de una máscara 3D original y se compara con la máscara arreglada.

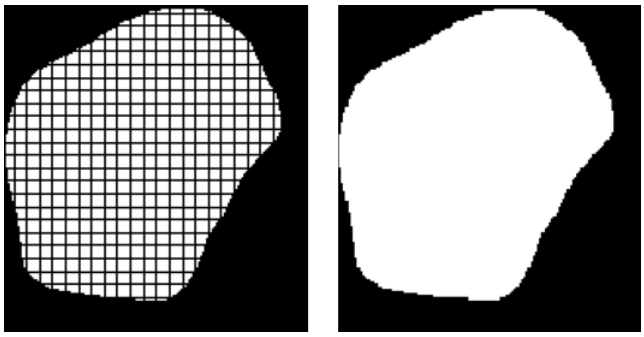

Figura 63: Cortes axiales de máscaras de segmentación del tumor CCR_4 en F_N [8]. A la izquierda: corte 33 de la máscara original; se observa el error por cuadriculado en los *ejes X e Y*. A la derecha: corte 66 de máscara corregida con cierre morfológico; posee igual forma que la máscara original pero soluciona la falla mencionada. Cabe mencionar que la operación morfológica con filtro 3D también reestablece los cortes faltantes en el *eje Z*. Por esta razón la máscara 3D arreglada presenta el doble de planos en dicho eje, y su corte 66 corresponde al corte 33 de la máscara original.

5.2.3. Obtención de imágenes de tres canales (3C)

Por último, se implementó un preprocesamiento que vincula distintas fases en una misma imagen con 3C. A continuación se explica el proceso de obtención de estas matrices numpy a partir de un tumor.

El método utiliza como base las imágenes de la neoplasia obtenidas con el preprocesamiento *Contorno por corte*; a partir de las mismas se calcula el área de cada corte, que se define como la multiplicación de la cantidad de píxeles de ambos ejes. Con estos datos, para cada fase, se selecciona un número de cortes del tumor con mayor área. Además se les asigna de forma ordenada un número de puesto, identificando a la matriz de mayor tamaño con el puesto 1.

Por ejemplo, si el hiperparámetro de número de corte es dos, se eligen los dos cortes mayores por fase: la sección con mayor área se reconoce como puesto 1, mientras que la otra se considera puesto 2. En la Figura 64 se muestra el *dataframe* que contiene la información de los cortes elegidos, con estas condiciones, para el tumor CCR_20.

	Nom_corte	Label	Fase	Num_tum	Num_corte	Path	Area_corte	Puesto
0	FCsep_020_CCR_21	1	0	20	21	_FC/CCR/FCsep_020_CCR_21.npy	1295	1
1	FEsep_020_CCR_24	1	1	20	24	_FE/CCR/FEsep_020_CCR_24.npy	1444	1
2	FNsep_020_CCR_21	1	2	20	21	_FN/CCR/FNsep_020_CCR_21.npy	1295	1
3	FSCsep_020_CCR_21	1	3	20	21	_FSC/CCR/FSCsep_020_CCR_21.npy	1295	1
4	FCsep_020_CCR_22	1	0	20	22	_FC/CCR/FCsep_020_CCR_22.npy	1295	2
5	FEsep_020_CCR_21	1	1	20	21	_FE/CCR/FEsep_020_CCR_21.npy	1444	2
6	FNsep_020_CCR_22	1	2	20	22	_FN/CCR/FNsep_020_CCR_22.npy	1295	2
7	FSCsep_020_CCR_22	1	3	20	22	_FSC/CCR/FSCsep_020_CCR_22.npy	1295	2

Figura 64: Tabla que contiene información sobre los dos cortes mayores por fase (puesto 1 y 2) del tumor CCR_20. Las secciones axiales del puesto 1 y 2 coinciden en el área; esto puede ocurrir por ser cortes consecutivos de la máscara segmentada. En el *dataframe* se exponen como columnas la etiqueta y número del tumor, los números de cortes elegidos para cada fase y su correspondiente área. Además se informa dónde se guarda cada matriz numpy para poder obtenerla. Se observa que para la F_C, F_N y F_SC la sección axial con mayor área es la 21; en cambio para F_E es la 24. El puesto 2 lo ocupa el corte 22 para las fases F_C, F_N y F_SC, mientras que para la F_E el segundo corte mayor es el 21. Esta situación se repite en varios pacientes, ya que las correcciones de las máscaras se realizaron mayormente en la F_E.

Luego se modifican las dimensiones de los cortes al tamaño de entrada de la red neuronal (hiperparámetro) y se unen las matrices de un mismo puesto para conformar imágenes de 3C. Si se poseen las cuatro fases tomográficas, se generan tres imágenes por puesto: el tercer canal siempre corresponde a la F_SC y, en las dos primeras capas, se varía la presencia de las fases contrastadas (F_C, F_N o F_E). En la Figura 65 se muestra un ejemplo, donde se indican los canales que conforman cada imagen. En cambio, si sólo se tienen tres fases de un paciente, se obtiene una única imagen (Figura 66). Los casos con menos de tres fases son excluidos.

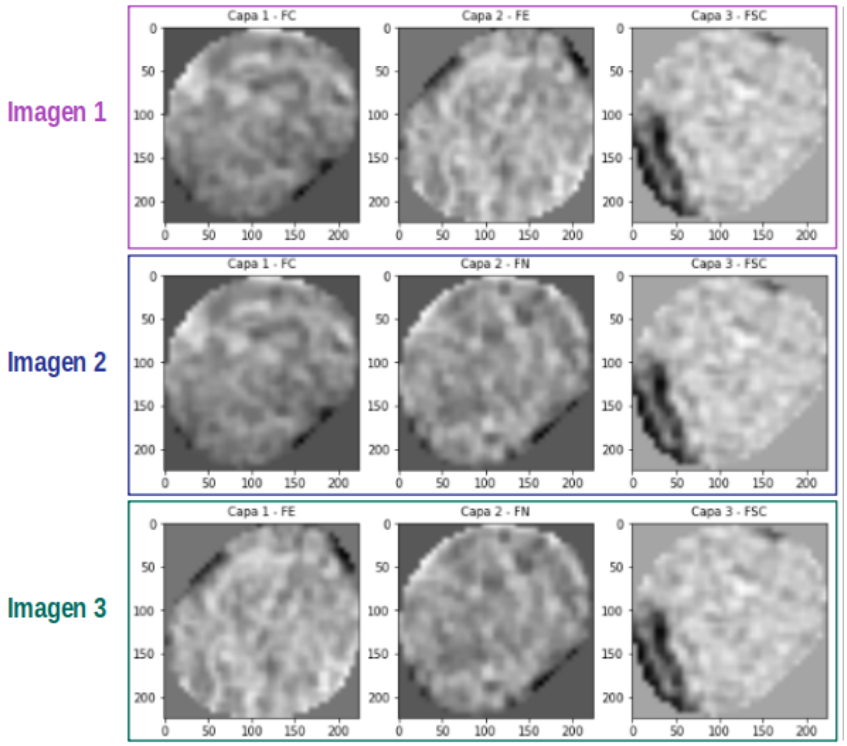

Figura 65: Canales de las imágenes 3C del puesto 1 del tumor CCR_20. Cada fila corresponde a una imagen 3C (de dimensión 224x224x3) y se muestra el orden de disposición de las fases en las capas. 1er imagen: F_C, F_E y F_SC; 2da imagen: F_C, F_N y F_SC; 3er imagen: F_E, F_N y F_SC.

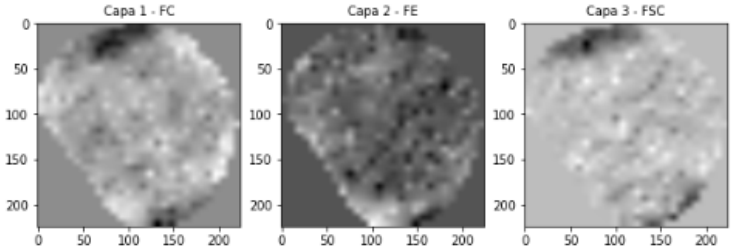

Figura 66: Canales de la imagen 3C del puesto 1 del tumor CCR_35. En este caso, el estudio disponía de 3 fases: F_C, F_E y F_SC; por lo tanto, se obtiene sólo una matriz de 3C. La dimensión de esta imagen es 224x224x3.

Con este tipo de preprocesamiento y usando sólo el primer puesto, se obtienen 244 imágenes de 3C del grupo de entrenamiento, 38 del conjunto de validación y 69 del grupo de testeo. Al usar determinado número de puestos, las cantidades de imágenes aumentan en dicho factor; por ejemplo si se emplean dos puestos, el volumen de datos se duplica.

Esta metodología es similar a la aplicada en la publicación previa mencionada en la sección *Estado del arte* [43]. Sin embargo, no fue necesario realizar un proceso de registración, ya que los bordes de las imágenes utilizadas coinciden con los límites del tumor en los distintos planos. En adición, se agregó la opción de seleccionar más de un corte de mayor área, generando un mayor número de imágenes de entrada a la red neuronal.

5.3. Factores de diseño

Se identifican con este nombre a características de diseño que deben decidirse previo al entrenamiento de la red. Se distinguen los factores de diseño vinculados a la base de datos 2D y los relacionados con la estructura del modelo de CNNs.

5.3.1. Factores de diseño de la base de datos en 2D

Es importante mencionar la adición de metadatos a las imágenes. Se consideraron que podrían ser útiles para el filtrado de la base de datos o para el entrenamiento de la red.

- **Fase:** Se establece de manera numérica la fase de cada tensor de entrada del modelo. Se asignan valores entre 0 y 3 a las fases F_C, F_E, F_N y F_SC respectivamente.
- **Ubicación del corte:** Se identifica el corte medio del tumor con posición 0 (Figura 67). Al alejarse del centro en sentido craneal se consideran valores positivos, es decir la posición de los cortes aumenta hasta llegar al extremo superior con ubicación 100 (como porcentaje). La posición de cada corte se calcula de manera lineal, dependiendo de cuántos cortes existen entre la ubicación 0 y 100. Los cortes ubicados en el sentido caudal respecto del centro del tumor, son asignados de igual manera pero con valores negativos; el extremo inferior presenta ubicación -100.

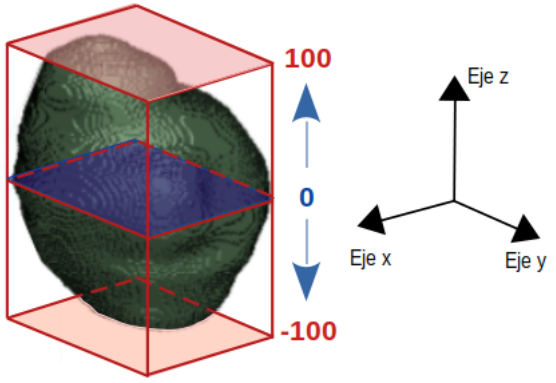

Figura 67: En la representación del volumen de la máscara del tumor CCR_2 [8], se marca el corte central con color azul y se le asigna el valor de ubicación 0. El *eje Z* es el eje cráneocaudal; su sentido positivo corresponde al sentido craneal, mientras que el negativo se condice con el sentido caudal. La ubicación del corte del extremo superior se considera 100; en cambio la ubicación del corte del extremo inferior tiene el mismo módulo pero negativo (-100).

- **Área del corte:** Se calcula utilizando las imágenes obtenidas por el preprocesamiento *Contorno por corte*, que delimita los cortes en base a los límites del tumor en cada plano. Se determina la cantidad de píxeles del *eje X* y del *eje Y*; se calcula el área multiplicando ambas medidas.
- **Porcentaje de píxeles con tumor/píxeles totales:** Contabiliza la cantidad de píxeles con información del tumor presentes en un corte extraído con *Contorno por tumor* y calcula la proporción entre este valor y el total de píxeles de la imagen (Figura 68). De esta manera se puede identificar si una imagen contiene mucho fondo y pocos píxeles del tumor.

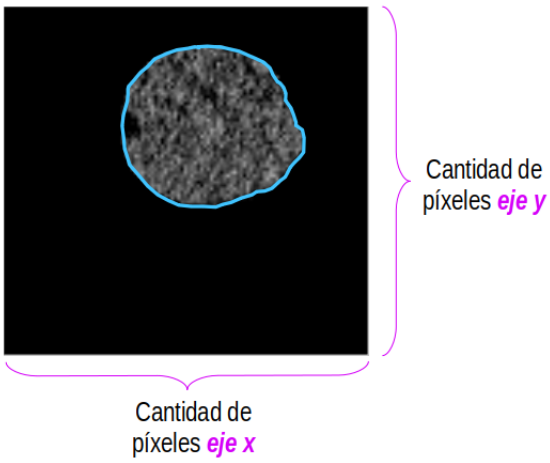

Figura 68: Corte 115 del tumor CCR_2 [8] obtenido con preprocesamiento con *Contorno por tumor*. En celeste se marca el contorno de los píxeles que representan el tumor. Por otro lado, el total de píxeles de la imagen se calcula multiplicando la cantidad de píxeles presentes en ambos ejes (*eje X* y *eje Y*).

A continuación, en la Figura 69, se exhibe como ejemplo una tabla con información sobre algunos cortes del tumor CCR_20. Las últimas cuatro columnas corresponden a las variables anteriormente alistadas. Al alejarse del extremo superior del tumor (corte 0 al inicio de la tabla), se distingue el incremento de las variables *Área del corte* y *Porcentaje de píxeles con tumor/píxeles totales*; en cambio, el valor de *Ubicación del corte* disminuye.

	Nom_corte	Path	Label	Num_tum	Num_corte	Fase	Ubic_corte	Area_corte	PorcentajeTum/Tot
0	FNsep_020_CCR_0	_FN/CCR/FNsep_020_CCR_0.npy	1	20	0	2	100.00	110	5.14
1	FNsep_020_CCR_1	_FN/CCR/FNsep_020_CCR_1.npy	1	20	1	2	95.83	195	9.86
2	FNsep_020_CCR_2	_FN/CCR/FNsep_020_CCR_2.npy	1	20	2	2	91.67	304	14.12
3	FNsep_020_CCR_3	_FN/CCR/FNsep_020_CCR_3.npy	1	20	3	2	87.50	378	18.18
4	FNsep_020_CCR_4	_FN/CCR/FNsep_020_CCR_4.npy	1	20	4	2	83.33	460	22.23
5	FNsep_020_CCR_5	_FN/CCR/FNsep_020_CCR_5.npy	1	20	5	2	79.17	550	25.95
6	FNsep_020_CCR_6	_FN/CCR/FNsep_020_CCR_6.npy	1	20	6	2	75.00	621	29.53
7	FNsep_020_CCR_7	_FN/CCR/FNsep_020_CCR_7.npy	1	20	7	2	70.83	700	32.77
8	FNsep_020_CCR_8	_FN/CCR/FNsep_020_CCR_8.npy	1	20	8	2	66.67	783	35.88
9	FNsep_020_CCR_9	_FN/CCR/FNsep_020_CCR_9.npy	1	20	9	2	62.50	840	39.19

Figura 69: *Dataframe* con datos de los primeros 10 cortes del tumor CCR_20 en F_N. La etiqueta (*label*) de valor 1 indica que es un tumor maligno; la fase con valor numérico 2 equivale a F_N. La variable *Ubicación del corte* se denomina *Ubic_corte* y *Área del corte* se menciona como *Area_corte*. Por último, la columna con información del *Porcentaje de píxeles con tumor/píxeles totales* se nombra *PorcentajeTum/Tot*.

5.3.1.1. Datos como matriz o imagen

En los múltiples métodos de preprocesamiento de cortes 2D, se puede almacenar cada sección axial como una matriz numpy o transferir dicha información a una imagen en PNG. Este formato de compresión no genera pérdida de calidad (a diferencia de JPG), sin embargo limita la intensidad de los píxeles a valores entre 0 y 255. En cambio, en los cortes almacenados como una matriz numpy no se modifica el rango de intensidad del píxel y se conservan los valores de la tomografía, asignados según la escala de Hounsfield.

Esta diferencia en el almacenamiento representa bases de datos distintas y determina los métodos de normalización. Por eso la elección del tipo de base de datos (numpy o imágenes) es una de las primeras decisiones tomadas antes de entrenar el modelo.

5.3.1.2. Exclusión de datos de entrada

Se pueden aplicar criterios de exclusión de cortes 2D basados en los distintos metadatos, es decir, conservando únicamente los casos que se encuentren dentro de un rango de valores de determinado metadato. Por ejemplo, existen pruebas de entrenamiento donde se seleccionan sólo los cortes del tumor cuya ubicación se encuentre entre -50 y 50 o con área del corte mayor a 400 píxeles. En la Figura 70, se aplica el criterio de exclusión por área al *dataframe* de la Figura 69.

	Nom_corte	Path	Label	Num_tum	Num_corte	Fase	Ubic_corte	Area_corte	PorcentajeTum/Tot
4	FNsep_020_CCR_4	_FN/CCR/FNsep_020_CCR_4.npy	1	20	4	2	83.33	460	22.23
5	FNsep_020_CCR_5	_FN/CCR/FNsep_020_CCR_5.npy	1	20	5	2	79.17	550	25.95
6	FNsep_020_CCR_6	_FN/CCR/FNsep_020_CCR_6.npy	1	20	6	2	75.00	621	29.53
7	FNsep_020_CCR_7	_FN/CCR/FNsep_020_CCR_7.npy	1	20	7	2	70.83	700	32.77
8	FNsep_020_CCR_8	_FN/CCR/FNsep_020_CCR_8.npy	1	20	8	2	66.67	783	35.88
9	FNsep_020_CCR_9	_FN/CCR/FNsep_020_CCR_9.npy	1	20	9	2	62.50	840	39.19

Figura 70: Tabla resultado de la exclusión de cortes con área menor a 400 píxeles. Se eliminaron las 4 primeras filas del *dataframe* original creado con los primeros 10 cortes del tumor CCR_20 en F_N y mostrado en la Figura 69.

Por otra parte, hay cortes que contienen muy poca información del tumor debido a que la máscara en dicha posición del *eje Z* es pequeña; un ejemplo es mostrado en la Figura 71. Para eliminar dichos cortes se realiza un filtrado del *dataframe* según la variable *Porcentaje de píxeles con tumor/píxeles totales* en base a las imágenes obtenidas con *Contorno por tumor*: así un corte proveniente de una máscara con menor área genera un número menor de esta variable. En todos los modelos 2D entrenados se excluyen los casos con valores inferiores al 3 %, ya que se considera que no aportan suficiente información al modelo.

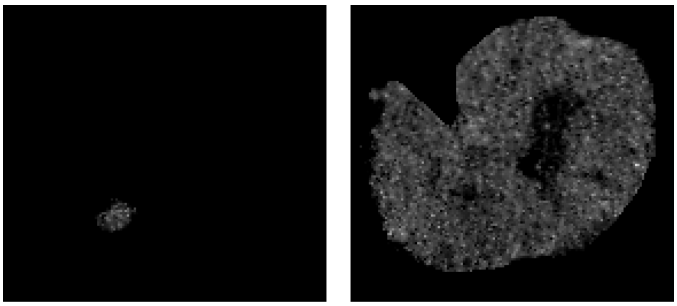

Figura 71: Cortes del tumor CCR_5 en F_N [8] adquiridos con el preprocesamiento *Contorno por tumor* (a la izquierda corte 0 ubicado en el extremo superior del tumor y a la derecha corte 99, más central). El corte 0 no brinda información útil sobre el tumor como lo hace el corte 99; por lo tanto se elimina de la base de datos mediante el proceso de filtrado que considera el porcentaje de tumor en dicha sección axial respecto al tamaño total del corte.

5.3.1.3. Selección de cortes centrales

Los tumores presentan distintos tamaños, por lo tanto el número total de cortes varía dependiendo de la neoplasia. Esto ocasiona que las masas tumorales más grandes representen un mayor número de entradas a la red, lo que puede ser perjudicial para el entrenamiento del modelo. Por este motivo, se define un factor de diseño que selecciona una cantidad fija de cortes centrales, para que cada tumor brinde igual cantidad de cortes 2D.

Para elegir los datos, en primer lugar se identifica en el *dataframe* el corte central de cada tumor, es decir, la sección axial cuyo valor de *Ubicación de corte* es cero. Luego se define un hiperparámetro que establece la cantidad de cortes, superiores e inferiores al central, que se conservan. Por ejemplo, si el hiperparámetro es 4, para cada masa tumoral se selecciona el corte central, los 4 cortes consecutivos en sentido positivo del *eje z* y otros 4 cortes en sentido negativo; de esta manera se obtienen los 9 cortes centrales de cada tumor (Figura 72).

	Nom_corte	Path	Label	Num_tum	Num_corte	Fase	Ubic_corte	
0	FNsep_020_CCR_20	_FN/CCR/FNsep_020_CCR_20.npy	1	20	20	2	16.67	CORTES SUPERIORES
1	FNsep_020_CCR_21	_FN/CCR/FNsep_020_CCR_21.npy	1	20	21	2	12.50	
2	FNsep_020_CCR_22	_FN/CCR/FNsep_020_CCR_22.npy	1	20	22	2	8.33	
3	FNsep_020_CCR_23	_FN/CCR/FNsep_020_CCR_23.npy	1	20	23	2	4.17	
4	FNsep_020_CCR_24	_FN/CCR/FNsep_020_CCR_24.npy	1	20	24	2	0.00	CORTE CENTRAL
5	FNsep_020_CCR_25	_FN/CCR/FNsep_020_CCR_25.npy	1	20	25	2	-4.17	CORTES INFERIORES
6	FNsep_020_CCR_26	_FN/CCR/FNsep_020_CCR_26.npy	1	20	26	2	-8.33	
7	FNsep_020_CCR_27	_FN/CCR/FNsep_020_CCR_27.npy	1	20	27	2	-12.50	
8	FNsep_020_CCR_28	_FN/CCR/FNsep_020_CCR_28.npy	1	20	28	2	-16.67	

Figura 72: *Dataframe* con la selección cortes centrales del tumor CCR_20 en F_N, cuando el hiperparámetro es 4. El corte 24 corresponde al central (recuadros violetas). Las secciones axiales 20 a 23, se encuentran en sentido craneal respecto al plano central del tumor, por eso su *Ubicación de corte* es positiva; en cambio los cortes 25 a 28 se ubican en sentido caudal.

5.3.1.4. Balanceado de la base de datos

La utilización de un dataframe balanceado o desbalanceado, es un factor de diseño considerado en el entrenamiento de los modelos. Como ya fue mencionado, la base de datos se encuentra desbalanceada, presentando un 66,06 % de CCRs y un 33,94 % de ONCs. Al obtener los cortes 2D los porcentajes son similares (varían según la exclusión de datos).

Se implementa el balanceo de clases en el *dataframe* del grupo de entrenamiento: se aplica un sobremuestreo de las filas de los ONCs hasta crear un conjunto de datos equilibrado, es decir, con igual número de cortes para ambos tipos de tumor. De esta manera existe igual probabilidad de que el generador elija un tumor benigno o maligno cuando forma un *batch*. El sobremuestreo puede provocar un sobreajuste (*overfitting*) a los datos de entrenamiento [40], ya que imágenes de ONCs se repiten más de una vez en un mismo ciclo de entrenamiento; sin embargo la aplicación de aumento de datos disminuye el riesgo de que esto ocurra.

5.3.2. Factores de diseño del modelo

Deben decidirse distintas características de diseño para definir la estructura del modelo. Por ejemplo, si se utiliza un modelo con múltiples entradas incorporando información adicional a las imágenes del tumor, o si se entrena cada fase tomográfica en un modelo por separado, entre otras opciones que se describen a continuación.

5.3.2.1. Parches como entradas del modelo

Bajo la suposición de que los tumores podrían clasificarse únicamente según su textura (sin utilizar la forma de la neoplasia como atributo), se desarrolló una metodología para aumentar la cantidad de muestras: en vez de utilizar todo el corte 2D, se toma un número de recortes del mismo como entrada del modelo. Estas secciones se denominan *parches*.

Entonces, un *parche* es un recorte de posición aleatoria (y de un tamaño determinado) de una matriz original obtenida a partir del método *Contorno por corte*. No se utilizan las matrices adquiridas con *Contorno por tumor* para evitar la obtención de muchos *parches* con contenido nulo.

La cantidad de *parches* elegidos por corte 2D y el tamaño de los mismos son hiperparámetros a configurar. En la Figura 73 se esquematiza la adquisición de 5 *parches* cuadrados con lados de 25 píxeles.

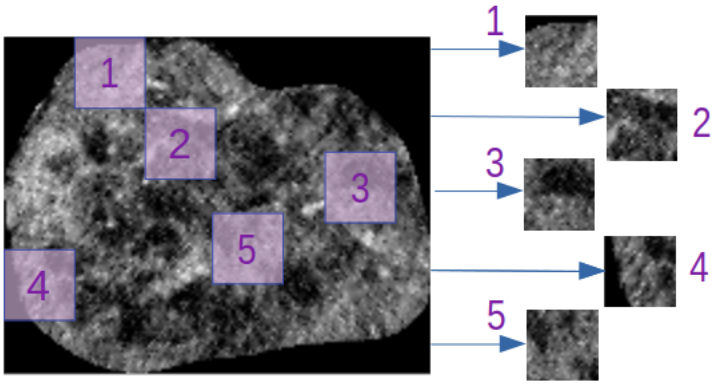

Figura 73: Esquema de la obtención de *parches* a partir de un corte axial. Se muestra el corte 80 del tumor CCR_4 en F_N [8]; se observa el posicionamiento aleatorio de los 5 *parches* generados.

Por otra parte, el método también se aplica para modelos 3D. En este caso, los *parches* poseen tres dimensiones: son recortes de la imagen 3D; se incorpora una condición para no tomar *parches* que contengan sólo valores nulos. Nuevamente son hiperparámetros el tamaño y la cantidad de *parches* utilizados.

Uno de los beneficios considerados con esta práctica es la generación de tensores de igual tamaño, sin precisar la aplicación de interpolación para igualar las dimensiones de las entradas a la red. Por otra parte, implica el uso de modelos de menor complejidad lo que conlleva menor tiempo de procesamiento.

5.3.2.2. Entradas adicionales de la red

Se consideró aplicar un modelo con múltiples entradas que, además del tensor con la imagen del tumor, incorpore metadatos que podrían ser importantes para el entrenamiento. Estas variables se disponen como columnas del *dataframe*. En los modelos 2D son: *Fase*, *Ubicación del corte*, *Área del corte*, *Porcentaje de píxeles con tumor/píxeles totales*; mientras que en los modelos 3D corresponde sólo a la *Fase*. Considerar alguna variable o combinación de las mismas como entrada del modelo es un factor de diseño.

El metadato más utilizado como entrada de la red fue la *Fase* (Figura 74). Como se explicó en la sección *Tomografía computarizada con contraste*, las fases son adquiridas a diferente tiempo luego del consumo de contraste y en ellas se realzan distintas estructuras anatómicas. Esta desemejanza lleva a examinar si es necesario mencionar el tipo de fase de cada imagen de entrada, para lograr que la red neuronal distinga las diferencias entre fases de las particularidades vinculadas con la clasificación del tumor, siendo estas últimas las que debe aprender.

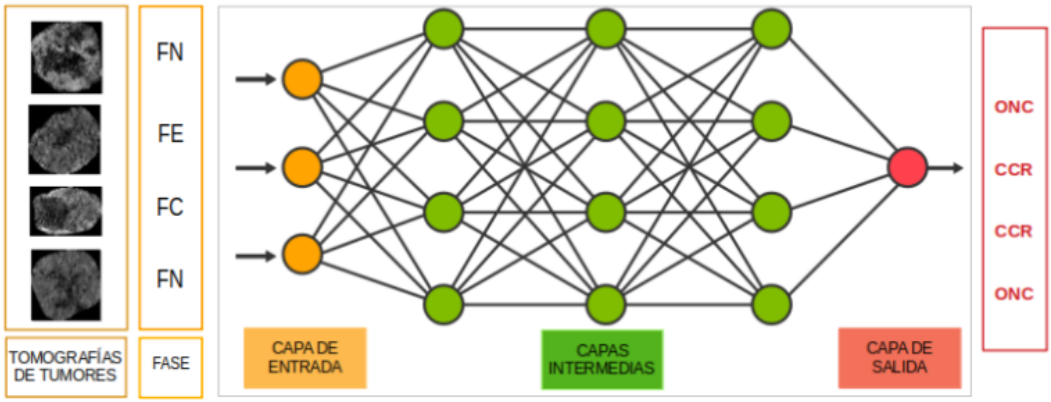

Figura 74: Esquema de una red neuronal con múltiples entradas; en este caso hay dos entradas (recuadros amarillos de la izquierda): tumores segmentados en tomografías contrastadas con su correspondiente tipo de fase. La salida de la red es la clasificación de los tumores de entrada en ONCs o CCRs. Imagen adaptada de [14].

5.3.2.3. Modelos por fase del tumor

En adición se probó utilizar cuatro modelos separados, cada uno entrenado con una fase distinta, como se muestra en la Figura 75. Luego se clasificó cada muestra del grupo de testeo según el modelo de la fase que le corresponde para proponer una clasificación conjunta.

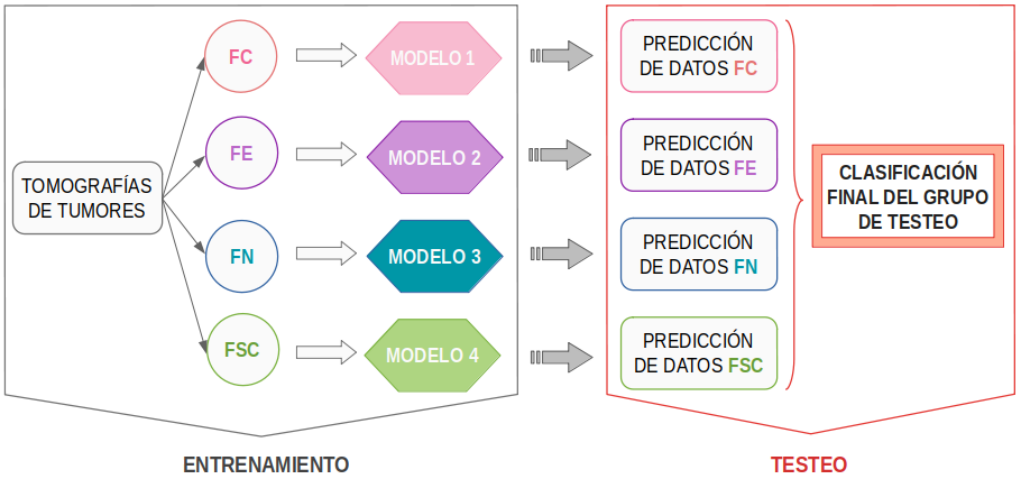

Figura 75: Esquema representativo del uso de modelos por fase. En primer lugar, cada red se entrena por separado con los datos tomográficos de una fase. Las predicciones de las entradas del grupo de testeo se realizan con el modelo entrenado correspondiente a su fase. Por último, la clasificación final se establece vinculando las predicciones de todas las fases del tumor.

Por otra parte, se consideró usar únicamente la F_N, ya que la segmentación de tumores fue realizada originalmente en las tomografías de esta fase. Esto confirma que las máscaras son correctas en la F_N, mientras que pueden existir problemas de registración en las otras fases. Considerar como entrada de la red neuronal sólo los resultados de preprocesamiento de esta fase elimina un posible error.

En el caso de emplear sólo la F_N, con el preprocesamiento 2D se tienen 5.042 cortes en el grupo de entrenamiento, 470 en el conjunto de validación y 1.055 en el de testeo. En cambio, con el preprocesamiento 3D existen 115 tensores en el grupo de entrenamiento, 17 en el conjunto de validación y 30 en el de testeo; estos valores coinciden con el número de tensores presentes en cada conjunto de datos, a excepción del grupo de testeo donde no existe información de la F_N en 3 CCRs.

5.3.2.4. Modelos por tamaño del tumor

Al analizar los tamaños de los cortes de los tumores presentes en la base de datos 2D, se notó que la mayoría presentaba un área menor que 3500 píxeles. Sólo 27 tumores (17 CCRs y 10 ONCs) presentan cortes con áreas superiores; en la Figura 76 se observa un histograma con esta información.

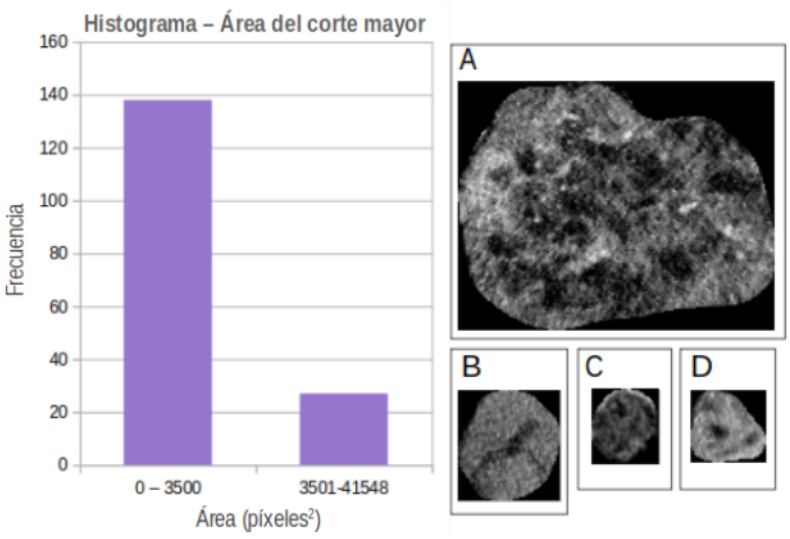

Figura 76: A la izquierda, histograma de la distribución del área del corte mayor de cada tumor. La mayoría de las neoplasias presentan todas sus secciones axiales con áreas menores a 3500 píxeles; únicamente 27 tumores contienen al menos un corte con área mayor a dicho umbral. A la derecha, 4 cortes de distintos tumores en F_N [8] obtenidos con el método *Contorno por corte*, que muestran la variabilidad de tamaños en la base de datos. A: Corte 80 del tumor CCR_1 (área de 20160 píxeles), se nota en un estadío avanzado; B: Corte 30 del tumor ONC_14 (2912 píxeles); C: Corte 25 del tumor CCR_20 (1292 píxeles); D: Corte 25 del tumor CCR_63 (1406 píxeles).

Un factor de diseño es la decisión de dividir la base de datos en dos grupos según dicho umbral de área y con cada uno de ellos entrenar un modelo distinto (Figura 77), debido a las siguientes consideraciones:

- Se examinó la posibilidad de que los tumores grandes aportaran más información de textura debido a la mayor cantidad de píxeles. Si el modelo con tumores mayores daba buenos resultados en comparación con el modelo de tumores chicos, podría suponerse que la menor superficie de los cortes del tumor afecta a la obtención de texturas útiles para la clasificación.
- Las dimensiones de las imágenes son modificadas para obtener un tamaño constante de entrada del modelo. Al utilizar dos modelos con datos con diferente rango de área, se pueden elegir hiperparámetros (de tamaño de entrada de la red) más específicos para cada grupo, logrando una menor modificación de la imagen en dicho proceso.

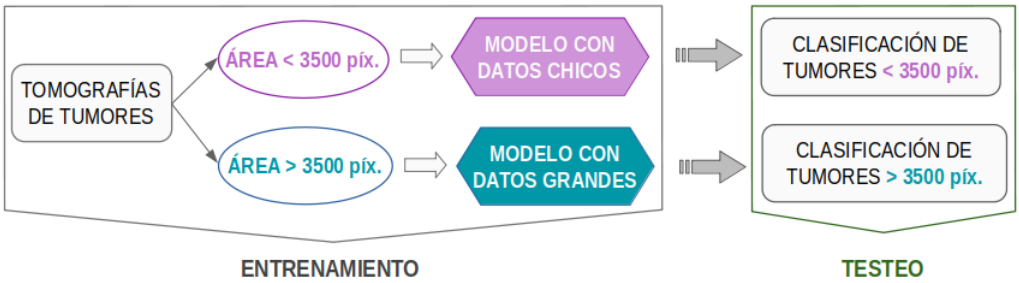

Figura 77: Esquema representativo del uso de modelos por tamaño del tumor. Se entrenan dos modelos por separado: uno con datos de los tumores mayores al umbral (3500 píxeles) y otro con el resto de las neoplasias. La clasificación de cada tumor del grupo de testeo se realiza utilizando el modelo que le corresponde según su tamaño.

Para su aplicación, la distribución entre conjunto de entrenamiento, validación y testeo fue rehecha en ambos grupos de tumores, para asegurar un igual desbalance de clases dentro de cada conjunto a evaluar.

5.4. Modelos utilizados

5.4.1. Modelo 2D con arquitecturas pre-entrenadas

En primer lugar se buscó utilizar modelos previamente entrenados con la base de datos *Imagenet*, variando las capas finales para adaptarlas a las dos salidas de este problema de clasificación. El modelo más utilizado fue ResNet50V2, cuya estructura se muestra en la Figura 78. Además de *Feature extraction* se probó la estrategia de *Fine-tuning* para ajustar mejor las representaciones generales al caso de estudio.

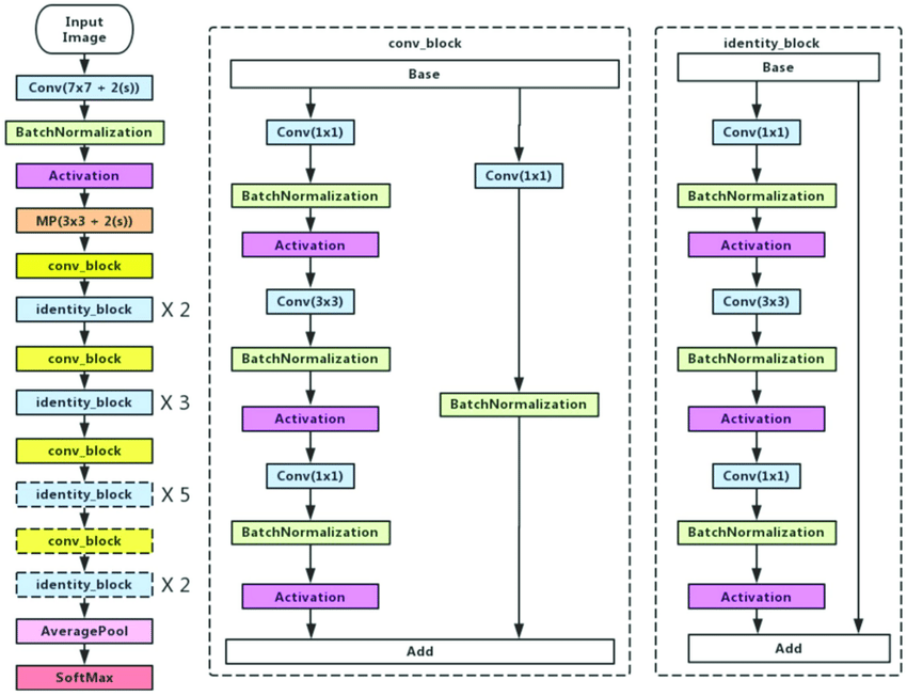

Figura 78: Estructura de la red ResNet50 [50]. Posee una capa de entrada, 5 bloques convolucionales, una capa *Global Average Pooling* y una capa densa que clasifica las entradas dentro de 1000 posibles clases. Las entradas de la red tienen dimensiones (224,224,3) y solamente la parte convolucional implica 23.519.360 parámetros entrenables. Para aplicar la red al proyecto de distinción de ONCs y CCRs, las últimas 2 capas fueron descartadas y reemplazadas por otro bloque de clasificación.

5.4.2. Modelo 2D con arquitectura propia

Por otro lado, se crearon redes ideadas desde cero para poseer un menor número de parámetros. Se realizaron múltiples pruebas con distintos modelos, pero a continuación se detalla solamente el modelo que fue utilizado en el caso expuesto en la sección *Resultados*. Estas redes presentan como única entrada los datos provenientes de las imágenes tomográficas de los tumores.

La CNN creada presenta una capa de entrada (**Input**) seguida por cuatro bloques. Los primeros tres conforman la estructura convolucional que extrae características de los datos de entrada. En cambio, el último bloque contiene capas densas para generar la clasificación de los tumores. En la Figura 79 se representa la estructura de la red neuronal, que posee 159.681 parámetros entrenables.

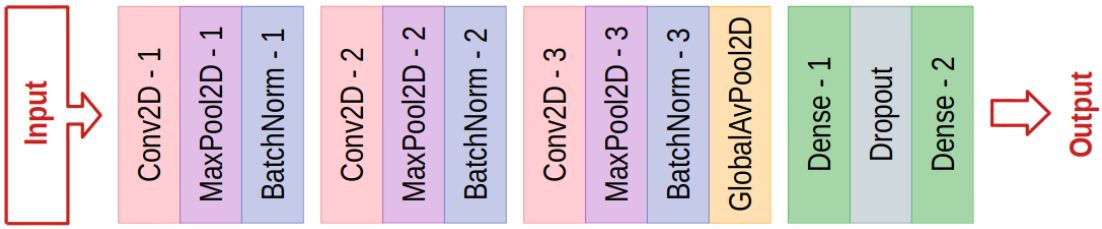

Figura 79: Esquema de la red neuronal creada y utilizada en el entrenamiento de cortes 2D. Los 3 primeros bloques contienen capas convolucionales (recuadros rosas), mientras que el bloque final tiene capas densas (recuadros verdes). Las entradas son información tomográfica de los tumores y la salida la clasificación de los mismos.

Como se observa en la Figura anterior, cada bloque convolucional presenta:

- Una capa convolucional 2D (**Conv2D**); la dimensionalidad del espacio de salida de este tipo de capa se duplica en cada bloque: respectivamente el número de filtros de salida es 32, 64 y 128. El *kernel* de convolución utilizado es de tamaño 3x3 y se utiliza la función de activación ReLU. Además, se configura un *stride* horizontal y vertical igual a 1 para la convolución, y no se emplea *padding*.
- Una capa *Max Pooling* (**MaxPool2D**), con un *kernel* de tamaño 2x2 y un *stride* horizontal y vertical igual a 2. No se emplea *padding*.
- Una capa *Batch Normalization* (**BatchNorm**), para normalizar los datos.

Únicamente en el tercer bloque, también se incluye una capa *Global Average Pooling* (**GlobalAvPool2D**). Esta promedia los datos de las distintas dimensiones espaciales, convirtiendo la salidas 3D en 2D. De esta manera, la red puede utilizar a continuación las capas densas.

El bloque de clasificación comienza con una capa densa (**Dense**) de 512 unidades y función de activación ReLU. Luego contiene una capa intermedia de *Dropout*, que convierte un 30 % de las características a cero. Por último, se dispone otra capa densa, con 1 unidad y función de activación sigmoidea: así brinda como salida la probabilidad de que el tumor sea un CCR (categoría positiva).

5.4.3. Modelo 3D

La red neuronal que utiliza como entrada tensores 3D, es equivalente a la CNN desarrollada en la sección anterior, pero usa capas convolucionales, *Max Pooling* y *Global Average Pooling* en 3D (respectivamente **Conv3D**, **MaxPool3D** y **GlobalAvPool3D**). Este modelo contiene 344.577 parámetros entrenables y su estructura se muestra en la Figura 80.

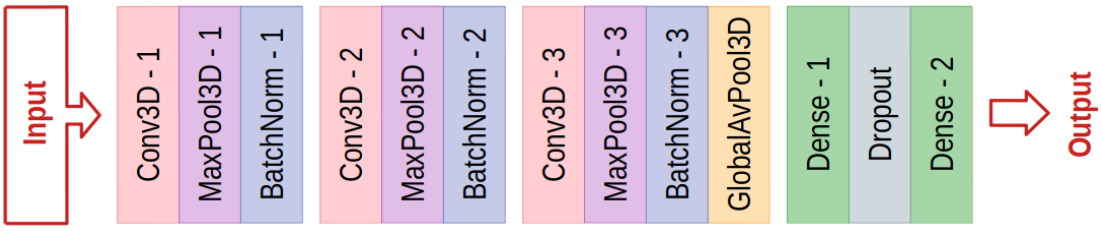

Figura 80: Esquema de la red neuronal creada y utilizada en el entrenamiento de tensores 3D. Los 3 primeros bloques contienen capas convolucionales (recuadros rosas), mientras que el bloque final tiene capas densas (recuadros verdes). Las entradas son información tomográfica de los tumores y la salida la clasificación de los mismos.

5.4.4. Configuración del entrenamiento

La función de costo usada fue la entropía cruzada binaria. La misma se utiliza para aplicaciones de clasificación binaria como la trabajada en este proyecto. Por otra parte, existen varios hiperparámetros que deben configurarse:

- *Dimensiones de las entradas:* Hay diversidad de tamaños entre los tumores y por lo tanto también entre los tensores extraídos de las tomografías. Estas matrices se escalan a las mismas dimensiones mediante la aplicación de interpolación, para ser utilizadas como entradas de la red. Entonces es fundamental la elección de dimensiones apropiadas.
- *Optimizador* y *Learning Rate:* Al compilar el modelo, se decide el tipo de optimizador que se utiliza y el *Learning Rate*. Debe buscarse un valor adecuado para esta variable: no debe ser demasiado grande para encontrar el mínimo de la función de costo y tampoco debe ser muy pequeña, para que el entrenamiento converja en un tiempo propicio.

- *Tamaño del batch:* La cantidad de datos cambia al trabajar con tensores 2D o 3D. Además según la prueba se puede variar el grupo de datos seleccionados como entrada de la red, en base a los factores de diseño elegidos (por ejemplo se puede emplear sólo una fase tomográfica, excluir datos con poca información del tumor, seleccionar los cortes centrales, etcétera). Entonces es importante modificar el tamaño de *batch* y elegir un valor menor cuando existen menos datos de entrada del modelo.

- *Cantidad de epochs* y *utilización de callbacks:* El número de ciclos de entrenamiento del modelo es un hiperparámetro a definir. Además la aplicación de *callbacks* puede ayudar en el proceso de entrenamiento, por ejemplo reduciendo el *Learning Rate* o deteniendo el entrenamiento cuando una métrica es constante por determinado número de *epochs*. Cabe mencionar que en todas las pruebas se utilizó el *callback Model Checkpoint*, el cual guarda los parámetros que brindaron los mejores resultados de la función de costo o cierta métrica.

Por último, la métrica de desempeño utilizada es el *accuracy*, que calcula la fracción de clasificaciones predecidas correctas respecto del total. Al trabajar con una base de datos desbalanceada se debe tener cuidado al usar este índice. Por ejemplo, si el modelo tiende a clasificar todas las entradas como CCRs (que representa aproximadamente un 66 % de la base de datos), resulta en un número alto de verdaderos positivos pero la clasificación no es efectiva.

5.5. Evaluación del modelo

Para cada entrada del grupo de testeo, ya sea cortes 2D o tumores 3D, se calcula una predicción utilizando un modelo entrenado; en la Figura 81 se expone como ejemplo un *dataframe* con predicciones a partir de cortes 2D. Cabe aclarar que todas las predicciones se pueden conseguir utilizando una misma red neuronal o distintas. Esto último ocurre en la técnica *Modelos por fase del tumor* o *Modelos por tamaño del tumor*.

Luego se establece un umbral mediante el cual se traslada la predicción a un resultado binario (ONC o CCR). Es decir, si la predicción es superior o igual al mismo, la entrada se clasifica como CCR; en caso contrario, se considera ONC. El umbral es un hiperparámetro y se define como el valor que brinda el mejor resultado de clasificación en el grupo de validación.

	Nom_corte	Fase	Num_tum	Num_corte	Label	Predicción
0	FNsep_089_CCR_0	2	89	0	1	0.656483
1	FNsep_089_CCR_1	2	89	1	1	0.653804
2	FNsep_089_CCR_2	2	89	2	1	0.651162
3	FNsep_089_CCR_3	2	89	3	1	0.651747
4	FNsep_089_CCR_4	2	89	4	1	0.649291

Figura 81: Tabla ejemplo que muestra las predicciones obtenidas para los primeros 5 cortes del tumor CCR_89 en F_N. Se observa que las predicciones son cercanas a 0.65. Por lo tanto si el umbral se establece como 0.6, todos los cortes son clasificados como CCRs. En cambio, si se elige un umbral igual a 0.7, las secciones axiales se identifican como ONCs, lo que es incorrecto ya que difiere de la etiqueta del tumor.

Los métodos de clasificación del grupo de testeo varían según se trabaje con entradas 2D o 3D; a continuación se explica la manera en que se aplican. En adición, se enumeran las métricas de evaluación utilizadas.

5.5.1. Método de evaluación 2D

Una vez aplicado el umbral a las predicciones de todo el grupo de testeo, se contabiliza para cada tumor la cantidad de cortes clasificados como CCRs y el número de cortes identificados como ONCs. Si un tumor presenta mayor cantidad de cortes clasificados como CCRs, se reconoce al mismo como CCR. En caso contrario, se clasifica al tumor como ONC. Por otra parte, si existe igual número de secciones axiales pertenecientes a ambas clases, se considera CCR. Esta última condición, se debe a la preferencia de obtener un falso positivo respecto a un falso negativo en el campo médico, es decir, es mejor suponer que el paciente tiene cáncer cuando el tumor es benigno, en vez de considerar que no debe extraerse la neoplasia por una clasificación incorrecta como ONC.

Los resultados de clasificación de cada tumor del grupo de testeo se expresan en una tabla (Figura 82) que muestra la etiqueta del mismo, la predicción final y la cantidad de cortes clasificados como CCRs y ONCs. En la última columna se compara la predicción con la etiqueta para determinar si la clasificación es correcta (*True*) o incorrecta (*False*).

	Nom_tum	Label	Cortes_CCR	Cortes_ONC	Clasificación	Comparación
0	CCR_88	CCR	63	26	CCR	True
1	CCR_89	CCR	30	2	CCR	True
2	CCR_90	CCR	41	23	CCR	True
3	CCR_91	CCR	16	16	CCR	True
4	CCR_92	CCR	48	4	CCR	True
5	CCR_93	CCR	37	40	ONC	False

Figura 82: Tabla con un posible conjunto de predicciones de los 6 primeros CCRs del grupo de testeo en 2D. En este caso, sólo el tumor CCR_93 está mal clasificado. La suma de los cortes identificados como CCRs y los reconocidos como ONCs equivale a la cantidad de secciones axiales de ese tumor considerando todas las fases.

Por otra parte, cuando se usan *parches* como entrada de la red (tanto en 2D como 3D), se aplica el mismo procedimiento: con el modelo se predice la clase de cada uno de los *parches* extraídos del grupo de testeo y se aplica un umbral para su clasificación. Se define cada tumor como ONC o CCR según la categoría que presenta mayor cantidad de casos.

5.5.2. Métodos de evaluación 3D

En el preprocesamiento 3D, cada tumor equivale solamente a una entrada por fase tomográfica. En consecuencia, como máximo hay 4 datos de entrada por tumor en el grupo de testeo (en el caso de presentar todas las fases). Se aplicaron dos técnicas para clasificar las neoplasias:

- **Clasificación según mayoría de predicciones:** Similar al procedimiento anteriormente mencionado. Si el número de fases clasificadas como CCRs es mayor o igual que la cantidad de fases identificadas como ONCs, se considera que el tumor es un CCR (Figura 83). En caso contrario, la masa tumoral se establece como ONC. De esta manera, por ejemplo si dos fases se predicen como CCRs y las otras dos como ONCs, el tumor se clasifica como CCR, nuevamente para evitar falsos positivos.

	Nom_tum	Label	Fases_CCR	Fases_ONC	Clasificación	Comparación
0	CCR_88	CCR	2	0	CCR	True
1	CCR_89	CCR	0	1	ONC	False
2	CCR_90	CCR	4	0	CCR	True
3	CCR_91	CCR	2	0	CCR	True
4	CCR_92	CCR	2	1	CCR	True
5	CCR_93	CCR	2	2	CCR	True

Figura 83: Tabla con predicciones de los 6 primeros CCRs del grupo de testeo en 3D, obtenida utilizando la *Clasificación según mayoría de predicciones*. En la última columna, se comparan las etiquetas (columna *Label*) con las clasificaciones (columna *Clasificación*). En las columnas *Fases_CCR* y *Fases_ONC*, se identifica la cantidad de fases de cada tumor clasificadas como CCR y ONC respectivamente.

- **Clasificación según predicción más certera:** Para cada tumor, se analizan las predicciones generadas a partir de las distintas fases. Se conserva únicamente la mejor predicción, es decir, la que posee menor diferencia con el extremo 0 o 1. Se considera una predicción más certera porque exhibe con mayor probabilidad la categoría a la que pertenece el dato de entrada. En la Figura 84, se esquematiza el proceso de búsqueda de la mejor predicción.

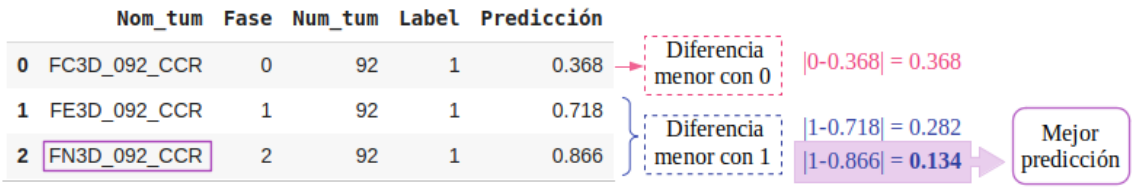

Figura 84: Ejemplo de hallazgo de la mejor predicción para el tumor CCR_92. En este caso, hay tomografías de las fases F_C, F_E y F_N; falta la F_SC. La menor diferencia es obtenida para el tumor en F_N, entonces se considera a esta predicción como la más certera.

Luego la mejor predicción se traslada a ONC o CCR mediante el uso del umbral y se clasifica al tumor dentro de dicha clase. En esta metodología el umbral posee un valor de 0.5. En un *dataframe* (Figura 85) se alistan las neoplasias del grupo de testeo con su etiqueta, la mejor predicción y la clasificación adquirida.

	Nom_tum	Label	Mejor_predicción	Clasificación	Comparación
0	CCR_88	CCR	0.675	CCR	True
1	CCR_89	CCR	0.251	ONC	False
2	CCR_90	CCR	0.718	CCR	True
3	CCR_91	CCR	0.770	CCR	True
4	CCR_92	CCR	0.866	CCR	True
5	CCR_93	CCR	0.766	CCR	True

Figura 85: Tabla con predicciones de los 6 primeros CCRs del grupo de testeo en 3D, obtenida utilizando la *Clasificación según predicción más certera*. Se observa que la mejor predicción del tumor CCR_92 coincide con la calculada en la Figura 84. Con un umbral de 0.5 se define la clasificación entre ONCs y CCRs.

5.5.3. Métricas de evaluación

Para evaluar los resultados del grupo de testeo, se usaron varias métricas. Como en el entrenamiento, se utilizó el *accuracy*, esta vez para determinar la cantidad de tumores del grupo de testeo bien clasificados respecto del total. Como ya fue mencionado si las clases están desbalanceadas, los valores de este índice pueden ser malinterpretados.

En consecuencia, también se implementaron métricas que se adaptan mejor a la situación de diferencia de cantidad de datos entre clases: se calculó sensibilidad (o *recall*), especificidad, VPP (o precisión) y VPN. En adición, se computa el Valor-F1 que combina las medidas de precisión y *recall* en un sólo valor.

Por otra parte, se cuantificó el AUC-ROC y AUC-PR. La curva PR considera más el desbalance entre las categorías, por lo que la métrica AUC-PR puede ser más representativa para este proyecto.

6. Resultados

Se realizaron múltiples pruebas aplicando distintas metodologías de preprocesamiento (generando tensores 2D, 3D o 3C) y variando los factores de diseño, las arquitecturas neuronales y los hiperparámetros de entrenamiento. La sección *Anexo 1* posee tres tablas que resumen los resultados obtenidos: en la primera tabla se indican los ensayos que usaron imágenes 2D; en la segunda, se identifican los casos que

utilizaron tensores 3D; por último, en la tercer lugar, se visualizan los resultados vinculados con el método 3C.

Cada tabla agrupa las pruebas en conjuntos y menciona las elecciones tomadas en cuanto al tamaño de las entradas y los factores de diseño. En referencia a las métricas registradas en las tablas, para cada grupo de ensayos se seleccionó el modelo que brindó el mejor resultado en el conjunto de testeo. Con dicho modelo, se calculó la sensibilidad y especificidad de la clasificación del grupo de testeo. Por otra parte, se anotó el mejor *accuracy* alcanzado en el entrenamiento del mismo, tanto para el conjunto de entrenamiento como para el de validación.

Por otra parte, dentro de esta sección se exhiben los dos modelos que brindaron los mejores resultados en el grupo de testeo. Una red neuronal fue entrenada con entradas 2D, mientras que la otra utilizó datos 3D. En primer lugar, se listan las decisiones de diseño que llevaron a la creación de cada modelo. Luego se muestran los gráficos de entrenamiento y por último los resultados del grupo de testeo: se visualizan las curvas ROC y PR, se explica la elección del umbral, se identifica la clasificación de este conjunto y se exponen las métricas de evaluación obtenidas.

Por último, en el *Anexo 2* se incluyen los mejores resultados obtenidos a partir del entrenamiento de una red neuronal con imágenes 3C (de tres canales). Esta información no se incluye en la sección principal de *Resultados* ya que las métricas alcanzadas presentan valores inferiores a las métricas conseguidas al emplear los modelos 2D y 3D. Además el preprocesamiento 3C excluye una mayor cantidad de tumores (debido a la falta de fases tomográficas), lo que limita el análisis de clasificación en dichas neoplasias.

6.1. Resultados 2D

Se realizaron las siguientes decisiones sobre la base de datos de entrada de la red, la estructura del modelo y los hiperparámetros de su entrenamiento:

- **Preprocesamiento:** Los cortes 2D se obtuvieron usando la metodología *Contorno por corte*, donde los límites de la matriz coinciden con los bordes del tumor en cada sección axial. Este método no conserva el tejido circundante a la neoplasia.

- **Factores de diseño de la base de datos en 2D:**

  1. Los datos se almacenaron como matrices numpy.
  2. Se excluyeron tumores con falla de cuadriculado y con *Porcentaje de píxeles con tumor/píxeles totales* menor a 3 %.
  3. Se seleccionaron sólo 9 cortes centrales de cada tumor, tanto para el grupo de entrenamiento y validación como para el grupo de testeo.
  4. Se utilizó el *dataframe* de la base de datos desbalanceado.

- **Factores de diseño del modelo:** La red presenta como única entrada los cortes 2D; no posee entradas adicionales. Solamente se emplearon datos de la F_N, en base a la cual se realizó la segmentación manual.

- **Estructura y entrenamiento del modelo:** En la sección *Modelos utilizados* se explica la estructura de la red neuronal creada y utilizada en esta prueba; en la Figura 79 se muestran las capas que conforman la CNN. Por otra parte, se menciona el uso de la entropía cruzada binaria como función de costo y el *accuracy* como métrica. En la Tabla 4 se establece la configuración de hiperparámetros realizada.

Dimensiones de entrada	**Optimizador**	**Learning Rate**	**Tamaño del batch**	**Número de epochs**	**Callbacks**
(50,50)	RMSprop	0,0001	30	50	Model Checkpoint

Tabla 4: Configuración de hiperparámetros de la prueba 2D que generó los mejores resultados del grupo de testeo.

- **Evaluación del modelo:** Se aplicó el *Método de evaluación 2D*. Los dos posibles umbrales a utilizar en el conjunto de testeo, se determinaron de diferente manera. El primero corresponde al umbral que brinda el mayor valor del índice de Youden a partir de la curva ROC del grupo de validación. El segundo umbral evaluado corresponde al valor que maximiza el *accuracy* en el conjunto de testeo.

6.1.1. Gráficos de entrenamiento

En cuanto a las variables de entrenamiento, se analizan la función de costo y la métrica *accuracy* obtenidas para cada *epoch*; cabe aclarar que dichos valores se calculan en base a cada corte 2D, no a nivel paciente. Se elaboran respectivamente las gráficas de las Figuras 86 y 87. En ambas imágenes, a la izquierda se muestran las variables tanto para el grupo de entrenamiento como de validación, omitiendo *outliers* para visualizar mejor la variación entre valores. A la derecha, se indica los datos únicamente del grupo de entrenamiento, en todos los *epochs*; de esta manera se puede examinar la gráfica a mayor escala.

En el gráfico de función de costo del conjunto de entrenamiento, se observa una tendencia de disminución de los valores al aumentar el número de *epochs*. El máximo de la función es 0,634 (*epoch* 1) y el mínimo 0,542 (*epoch* 50); se distingue entonces que el decrecimiento es lento. Por otra parte, la gráfica de validación presenta medidas de la variable con mayor variabilidad. Además hasta el *epoch 35* (donde se alcanza un mínimo de 0,441) se muestra una tendencia de disminución, pero luego existen valores en aumento.

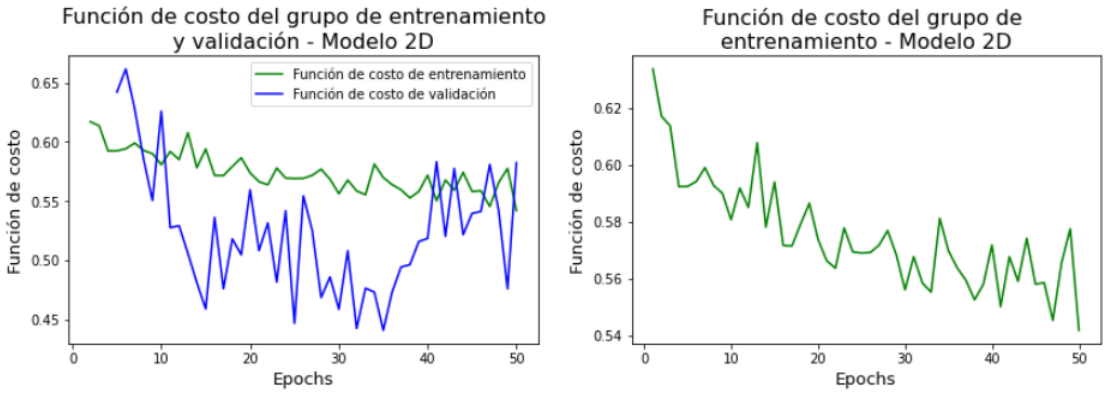

Figura 86: Gráficos de valores de función de costo adquiridos en el entrenamiento del modelo 2D. A la derecha se muestra la función de costo del grupo de entrenamiento (puntos verdes). A la izquierda también se incluyen los valores de esta variable que toma el grupo de validación (línea azul). Los *epochs* que presentaron *outliers* y fueron eliminados de la gráfica son el *epoch* 1 para el grupo de entrenamiento y los *epochs* entre 1 y 4 para el grupo de validación.

En el gráfico de *accuracy* se visualiza una tendencia de crecimiento de la variable, aunque existe variaciones entre medidas correspondientes a *epochs* continuos. El máximo valor obtenido de la métrica es 0,718 (*epoch* 41), entonces se considera que no se alcanzó *overfitting*. Para los datos de validación, en un comienzo la variable presenta un valor de 0,3 (*outlier* del *epoch* 1), luego aumenta y a partir del *epoch* 12,

se modifica entre 0,608 y 0,783, sin presentar una tendencia de aumento marcada.

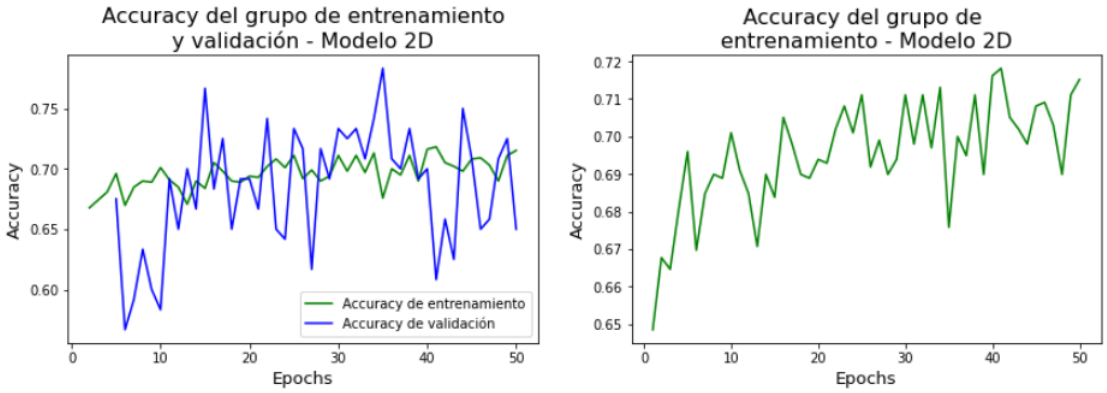

Figura 87: Gráficos de valores de *accuracy* obtenidos en el entrenamiento del modelo 2D. A la derecha gráfica con datos del grupo de entrenamiento (puntos verdes). A la izquierda además se muestran los resultados de dicha métrica para el grupo de validación (línea azul). En este último gráfico se excluyen *outliers*: para el conjunto de entrenamiento ocurre en los *epochs* 1 y 3; en el grupo de validación existen *outliers* entre los *epochs* 1 y 4.

6.1.2. Curvas ROC

Al trabajar con una base de datos 2D, existe más de un corte correspondiente a un mismo tumor y, por lo tanto, más de una predicción. Para disponer de una única probabilidad por masa tumoral, se calcula el promedio de las predicciones de cada caso. Con estos valores se obtiene la curva ROC de determinado grupo de datos.

En primer lugar se identificó el umbral que brindaba el mayor índice de Youden a partir de la curva ROC del grupo de validación. La gráfica se muestra en la Figura 88 (izquierda); el valor de AUC es 0,7. Se observa que ningún umbral provoca el acercamiento de la curva a la esquina superior izquierda, es decir, no se obtiene un buen resultado tanto de sensibilidad como de especificidad. El índice de Youden máximo en validación corresponde a un valor de 0,5 y se consigue con el umbral 0,615. Al aplicar dicho umbral para la clasificación del conjunto de testeo, se produce una curva ROC binaria con un AUC de 0,612 (Figura 88 - derecha).

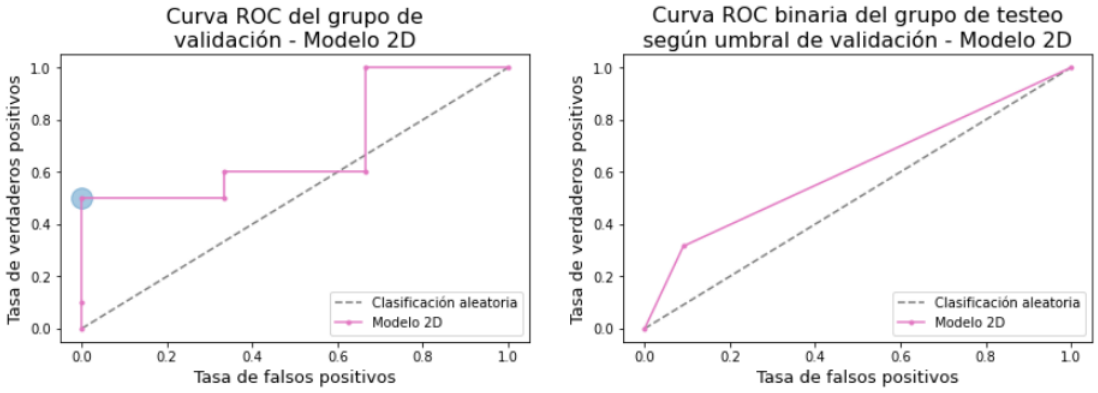

Figura 88: A la izquierda: Curva ROC del conjunto de validación, en base al promedio por tumor de las predicciones obtenidas a partir de los cortes 2D; el círculo azul marca el punto de la gráfica para el cual se obtiene el mayor índice de Youden. A la derecha: Curva ROC binaria del grupo de testeo, en base a la clasificación conseguida con el umbral 0,615. En ambos gráficos, la línea punteada gris representa una clasificación arbitraria entre las dos clases.

Debido a que el grupo de validación no generó resultados óptimos en los gráficos de entrenamiento ni en la curva ROC, se consideró que el mismo podría no ser representativo del conjunto total de datos. En consecuencia, se decidió no utilizar el umbral que deriva del análisis de los resultados del grupo validación.

Se evaluó entonces un segundo umbral dependiente del conjunto de testeo, a pesar del riesgo de provocar *overfitting* respecto a este grupo de datos. La curva ROC obtenida se muestra en la Figura 89 (izquierda) y presenta un AUC de 0,813. Según dicho gráfico, el umbral que genera una mejor sensibilidad y especificidad para la clasificación es el 0,343.

Sin embargo, si se consideran las probabilidades predecidas en base a los cortes 2D por separado (sin calcular el promedio por neoplasia) para la categorización del tumor utilizando el *Método de evaluación 2D*, se determina que el umbral 0,310 provoca mejores resultados para el grupo de testeo, ya que maximiza la métrica de *accuracy* en dicho conjunto. En la Figura 89 (derecha) se muestra la curva ROC binaria de los datos de testeo, resultado de la utilización del umbral 0,310 para la clasificación de los cortes 2D; el AUC obtenido es 0,856.

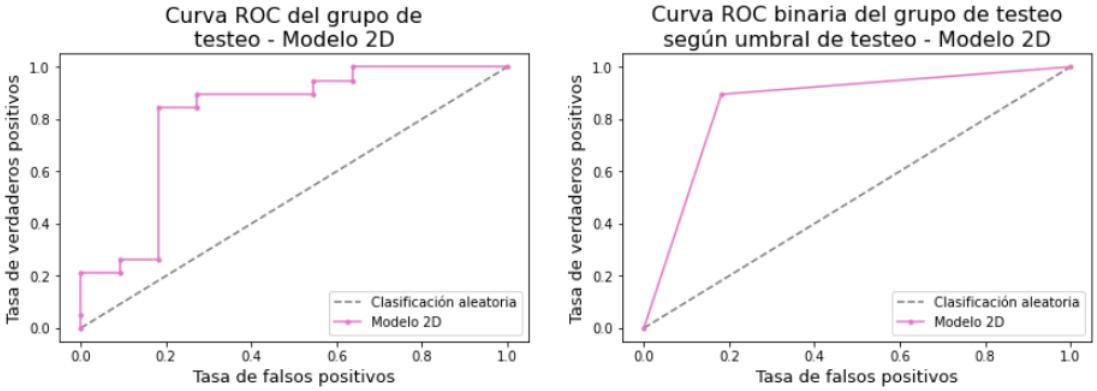

Figura 89: A la izquierda: Curva ROC del grupo de testeo, en base a las predicciones promedio del modelo 2D; para cada tumor, esta probabilidad se calcula como la media de las predicciones generadas por el modelo a partir de sus cortes 2D. A la derecha: Curva ROC binaria del grupo de testeo, en base a la clasificación obtenida con el umbral 0,310. En ambos gráficos, la línea punteada gris representa una clasificación arbitraria entre las dos clases.

Debido al hallazgo de buenos valores de AUC para el grupo de testeo, se determina que la distinción de clases propuesta por el modelo entrenado es aplicable a este conjunto de datos. Por lo tanto, en las siguientes secciones de análisis de resultados se utiliza el umbral 0,310 para evaluar la clasificación de los tumores de testeo y los valores del resto de las métricas.

6.1.3. Curva PR y Curva VPN-Especificidad

Para obtener los gráficos de esta sección también se precisa, para cada tumor, la media de las predicciones obtenidas a partir de sus cortes 2D. De esta manera se posee un único valor de probabilidad por neoplasia y se puede adquirir las curvas a nivel tumor.

En la Figura 90 (izquierda) se expone la curva PR, que se focaliza en el análisis de la clasificación de la clase con etiqueta positiva, es decir los CCRs, ya que muestra la relación entre la sensibilidad (*recall*) y el VPP (precisión). El AUC es 0,855 indicando un buen resultado para la categorización de tumores malignos.

La curva PR binaria representada en la Figura 90 (derecha), se obtiene al aplicar el umbral 0,310 para la clasificación de tumores con el *Método de evaluación 2D*. En este caso, el AUC es 0,928 superior al alcanzado con la curva PR general.

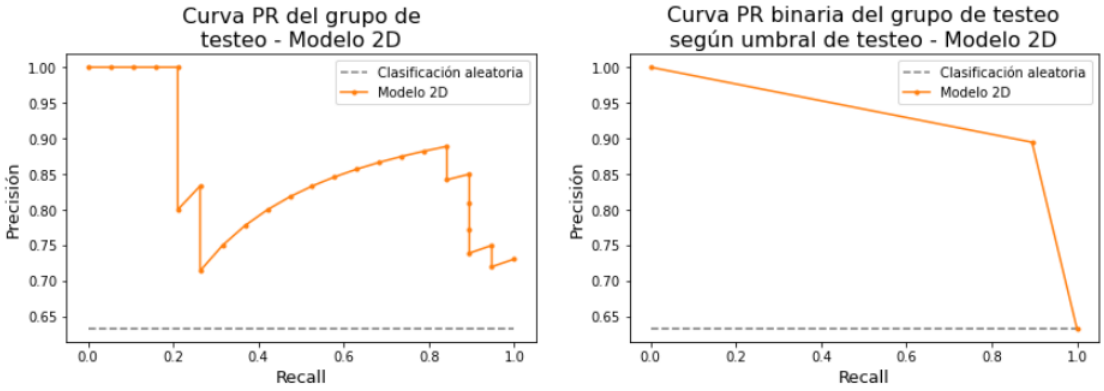

Figura 90: A la izquierda: Curva PR del grupo de testeo, en base a las predicciones promedio del modelo 2D para cada tumor. A la derecha: Curva PR binaria del grupo de testeo, según la clasificación adquirida con el umbral 0,310. En ambas imágenes, la clasificación arbitraria genera la línea punteada gris; al tratarse de la clase de tumor maligno presenta un valor de 0,633.

Por otra parte, se crea la gráfica equivalente para los ONCs (clase con etiqueta negativa). Dicha curva se observa en la Figura 91 (izquierda); analiza la relación entre la especificidad y el VPN, sin involucrar los VP. En consecuencia, examina únicamente los resultados de clasificación de los tumores benignos, categoría desfavorecida en cantidad de datos debido al desbalance entre las clases. El AUC de esta gráfica es 0,787.

Al usar el umbral 0,310 para la clasificación de tumores, se obtiene la curva VPN-Especificidad binaria mostrada en la Figura 91 (derecha). La misma presenta un AUC de 0,852, lo que evidencia un buena distinción de ONCs.

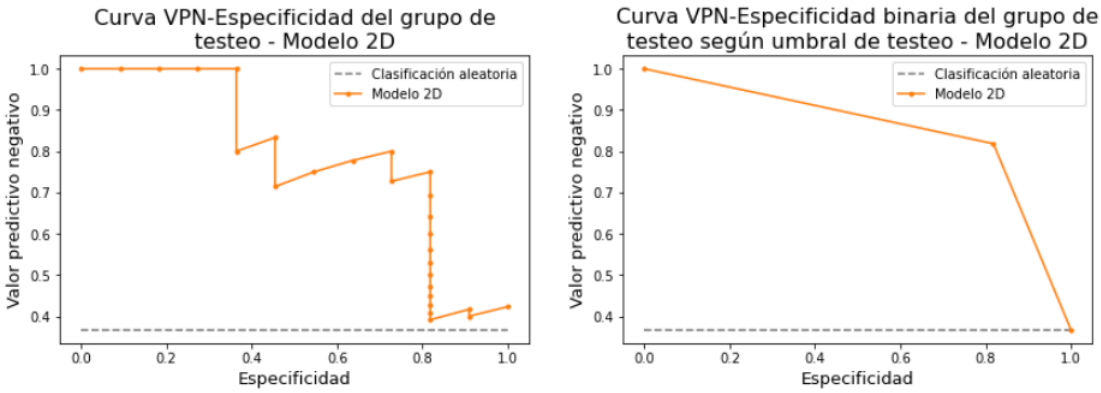

Figura 91: A la izquierda: Curva VPN-Especificidad del grupo de testeo, en base a las predicciones promedio del modelo 2D para cada tumor. A la derecha: Curva VPN-Especificidad binaria del grupo de testeo, según la clasificación adquirida con el umbral 0,310. En ambas imágenes, la clasificación arbitraria genera la línea punteada gris; en el caso de la clase de tumor benigno presenta un valor de 0,367.

6.1.4. Clasificación y métricas obtenidas

En primer lugar, es importante mencionar que al considerar únicamente la F_N, no existe información tomográfica de 3 CCRs del grupo de testeo (CCR_91, CCR_97 y CCR_98), ya que en dichos casos no se adquirió esta fase de estudio. Por lo tanto, el conjunto de datos está integrado por 11 ONCs y 19 CCRs.

En la Figura 92 se exponen los resultados de clasificación obtenidos en el grupo de testeo, al usar el umbral 0,310. Se clasifican correctamente 9 ONCs y 17 CCRs, mientras que 2 tumores de cada clase se categorizan de manera incorrecta. Estos resultados se trasladan a la matriz de confusión, mostrada en la Tabla 5.

	Nom_tum	Label	Predicción	Cortes_CCR	Cortes_ONC	Comparación
0	088_CCR	CCR	CCR	9	0	True
1	089_CCR	CCR	CCR	8	1	True
2	090_CCR	CCR	CCR	6	3	True
3	092_CCR	CCR	CCR	9	0	True
4	093_CCR	CCR	ONC	1	8	False
5	094_CCR	CCR	CCR	8	1	True
6	095_CCR	CCR	CCR	9	0	True
7	096_CCR	CCR	CCR	9	0	True
8	099_CCR	CCR	CCR	7	2	True
9	100_CCR	CCR	CCR	9	0	True
10	101_CCR	CCR	CCR	9	0	True
11	102_CCR	CCR	ONC	0	9	False
12	103_CCR	CCR	CCR	9	0	True
13	104_CCR	CCR	CCR	8	1	True
14	105_CCR	CCR	CCR	5	4	True
15	106_CCR	CCR	CCR	7	2	True
16	107_CCR	CCR	CCR	9	0	True
17	108_CCR	CCR	CCR	9	0	True
18	109_CCR	CCR	CCR	9	0	True

	Nom_tum	Label	Predicción	Cortes_CCR	Cortes_ONC	Comparación
19	046_ONC	ONC	ONC	0	9	True
20	047_ONC	ONC	ONC	3	6	True
21	048_ONC	ONC	ONC	2	7	True
22	049_ONC	ONC	ONC	1	8	True
23	050_ONC	ONC	ONC	0	9	True
24	051_ONC	ONC	ONC	1	8	True
25	052_ONC	ONC	CCR	9	0	False
26	053_ONC	ONC	ONC	0	9	True
27	054_ONC	ONC	ONC	4	5	True
28	055_ONC	ONC	ONC	3	6	True
29	056_ONC	ONC	CCR	9	0	False

Figura 92: Resultado de la clasificación del mejor modelo 2D, aplicando el umbral 0,310. A la izquierda, tabla con las predicciones de los CCRs: sólo el CCR_93 y el CCR_102 están mal clasificados (recuadros azules). A la derecha, tabla de los ONCs; 2 tumores fueron incorrectamente clasificados como CCR: el ONC_52 y el ONC_56 (recuadros azules).

		Clasificación	
		Positiva	*Negativa*
Etiqueta	*Positiva*	17	2
	Negativa	2	9

Tabla 5: Matriz de confusión del grupo de testeo en base a los resultados de clasificación obtenidos usando el modelo 2D y el umbral 0,310. Se recuerda que se considera positiva a la etiqueta de CCR y negativa a ONC. Existen 17 *VP*, 9 *VN*, 2 *FP* y 2 *FN*.

A partir de la tabla de confusión, se cuantifican las métricas mencionadas en la sección *Métricas de evaluación*. Las mismas se indican en la Tabla 6. La obtención de valores cercanos a la unidad indican buenos resultados.

Métrica	**Valor**
Accuracy	0,867
Sensibilidad (recall)	0,895
Especificidad	0,818
VPP (precisión)	0,895
VPN	0,818
Valor-F1	0,895
AUC-ROC	0,856
AUC-PR	0,928

Tabla 6: Métricas de evaluación calculadas en base a la clasificación implementada con el modelo 2D y el umbral 0,310.

6.2. Resultados 3D

En este caso se utilizaron los tensores 3D como entrada. A continuación se explican las decisiones de diseño que se eligieron en la prueba que presentó finalmente el mejor desempeño.

- **Preprocesamiento** Para obtener tensores 3D de entrada de la red, existe un único método de preprocesamiento. Se utilizaron las nuevas máscaras corregidas con operaciones morfológicas, para no perder información de tumores que presentaban la falla de cuadriculado.
- **Factores de diseño del modelo:** La CNN posee sólo una entrada con los tensores 3D. En la Figura 80 se representa la estructura de la red. Nuevamente se emplearon únicamente los datos de la F_N, para evitar posibles errores de registración entre las máscaras y la tomografía.

- **Entrenamiento del modelo:** Se utilizó la misma métrica y función de costo. En la Tabla 7 se mencionan los hiperparámetros elegidos.

Dimensiones de entrada	Optimizador	Learning Rate	Tamaño del batch	Número de epochs	Callbacks
(50,50,50)	RMSprop	0,00001	10	150	Model Checkpoint

Tabla 7: Configuración de hiperparámetros de la prueba 3D que generó los mejores resultados del grupo de testeo.

- **Evaluación del modelo:** Se usó el método de *Clasificación según mayoría de predicciones*. Sin embargo, al trabajar con una sola fase, el mismo equivale simplemente a clasificar el tumor en base a la predicción de la red y la aplicación del umbral.

  Se probaron dos umbrales: uno corresponde al valor que consigue el mayor índice de Youden a partir de la curva ROC del grupo de validación, y el otro se obtiene en base a la curva ROC del conjunto de testeo.

6.2.1. Gráficos de entrenamiento

Los resultados de la función de costo y el *accuracy* adquiridos en el entrenamiento, se muestran en las Figuras 93 y 94. A la izquierda se visualizan, a la vez, las variables del grupo de entrenamiento y de validación y se excluyen los *outliers*. A la derecha, se representa solamente los valores del grupo de entrenamiento en los 150 *epochs*.

En el gráfico de función de costo se observa una mayor variabilidad para los datos de validación que los correspondientes al conjunto de entrenamiento. Los valores de validación (considerando *outliers*) están en el rango 0,430-0,835, mientras que los resultados obtenidos para el grupo de entrenamiento varían entre 0,515 (mínimo en *epoch* 145) y 0,684 (máximo en *epoch* 3). En este último conjunto se ve una tendencia de disminución de los valores al aumentar el número de *epochs*.

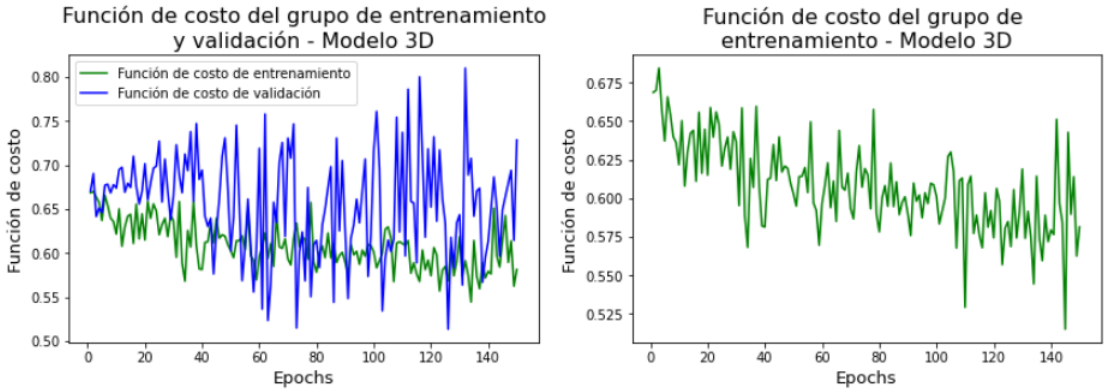

Figura 93: Gráficos de valores de la función de costo adquiridos en el entrenamiento del modelo 3D. A la derecha, valores de función de costo del grupo de entrenamiento (puntos verdes). A la izquierda, se agregan los datos del grupo de validación (línea azul). Los *epochs* que presentaron *outliers* y fueron eliminados de la gráfica son el 3, 110 y 145 para el grupo de entrenamiento. Además para el conjunto de validación son: 53, 57, 84, 96, 141, 146 y 147.

En el gráfico de la métrica *accuracy* del grupo de entrenamiento, se muestran resultados similares en distintos *epochs*, mayormente entre el rango de 0,65 y 0,70; el máximo conseguido es 0,755 sin generar *overfitting* del modelo. En el conjunto de validación existe mayor variabilidad de los valores de la métrica, incluyendo mínimos de 0.3 y máximos de 0.9 (ambos casos *outliers*). En este caso no se observa una tendencia de crecimiento de *accuracy*.

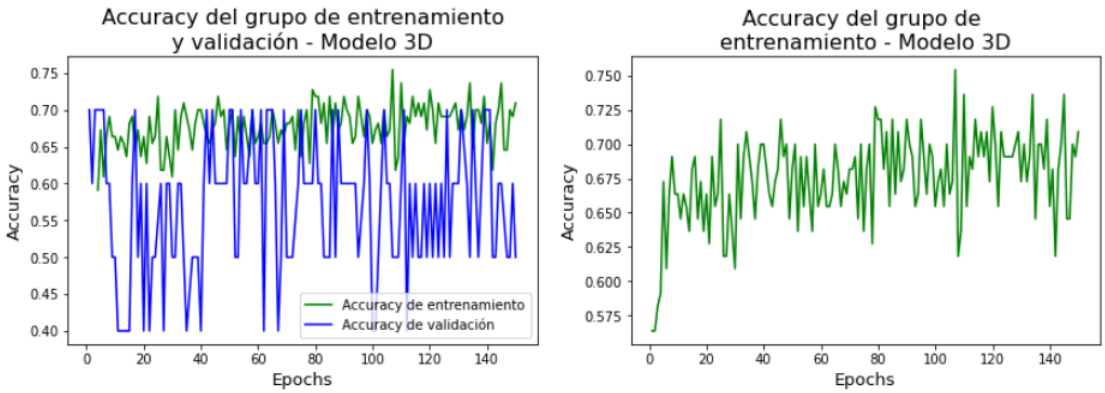

Figura 94: Gráficos del *accuracy* adquiridos en el entrenamiento del modelo 3D. A la derecha, métricas obtenidas a partir del conjunto de entrenamiento (puntos verdes). A la izquierda, también se muestran los resultados del *accuracy* para el grupo de validación (línea azul) y se excluyen *outliers*. Los *outliers* del conjunto de entrenamiento se encuentran entre los *epochs* 1 y 3. En cambio, los *epochs* con *outliers* del grupo de validación son: 4, 5, 25, 36, 38, 57, 73, 78, 84, 96, 132, 146.

En comparación con los gráficos de la métrica alcanzados al entrenar el modelo 2D, existe una menor cantidad de posibles medidas de *accuracy* debido al menor número de imágenes de entrada. Por esta razón el gráfico se visualiza con valores más repetitivos y se dificulta la observación de un crecimiento de la variable en el conjunto de entrenamiento.

6.2.2. Curvas ROC

Se calculó el umbral que maximiza el índice de Youden en el grupo de validación. La curva ROC de este conjunto se muestra en la Figura 95 (izquierda) y presenta un AUC igual a 0,652. No se halla un umbral que genere buenos resultados: con el valor 0,734 se consigue el mayor índice de Youden correspondiente a 0,379. Utilizando dicho umbral para la clasificación del grupo de testeo, se produce la curva ROC binaria que se expone en la Figura 95 (derecha), con un AUC de 0,541, valor bajo similar al AUC de la clasificación arbitraria (igual a 0,5).

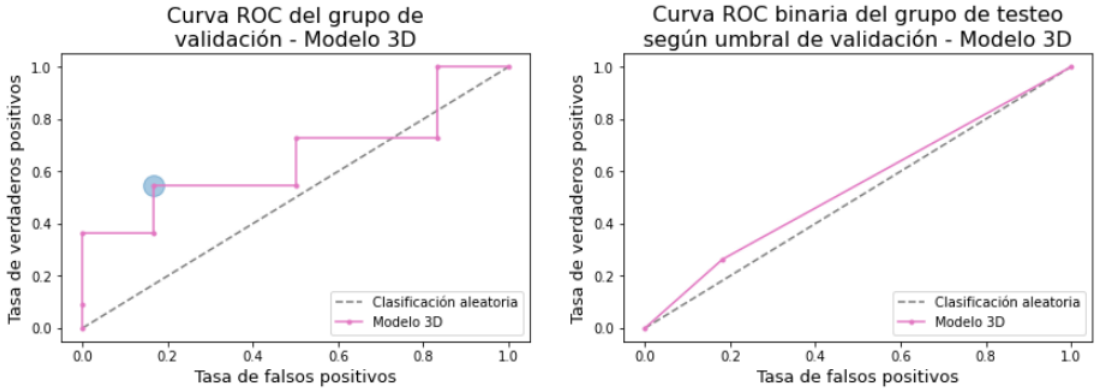

Figura 95: A la izquierda: Curva ROC del conjunto de validación, en base a las predicciones del modelo 3D; el círculo azul indica el punto de la gráfica para el cual se obtiene el mayor índice de Youden. A la derecha: Curva ROC binaria del grupo de testeo, en base a la clasificación obtenida con el umbral 0,734. En ambos gráficos, la línea punteada gris representa una clasificación arbitraria entre las dos clases.

Al igual que ocurre con los resultados del modelo 2D, no se produce una curva ROC óptima en base a las predicciones generadas para el grupo de validación: el gráfico presenta un bajo AUC e índice de Youden. Nuevamente se consideró que este conjunto de datos podría no representar la población total y se prosiguió analizando un umbral vinculado con el grupo de testeo.

Para conseguir el segundo umbral se graficó la curva ROC del conjunto de testeo y se detectó el valor que genera un máximo del índice de Youden. La curva ROC de este conjunto de datos se puede ver en la Figura 96 (izquierda). Posee un AUC de 0,766 y un índice de Youden máximo igual a 0,622 cuando el umbral es 0,391.

Se aplicó entonces este último umbral a los datos de testeo y se obtuvo la curva ROC binaria que se observa en la Figura 96 (derecha) con AUC 0,811. Por los resultados positivos, se determinó que la categorización de clases a partir del modelo 3D es realizable en el grupo de testeo. Se decidió utilizar el umbral 0,391 en las pruebas de clasificación de tumores presentadas en las siguientes secciones.

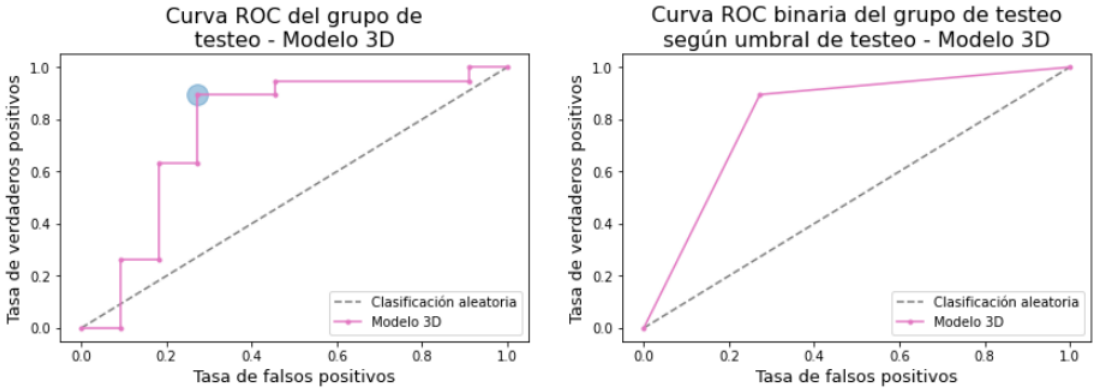

Figura 96: A la izquierda: Curva ROC del grupo de testeo, en base a las predicciones del modelo 3D. A la derecha: Curva ROC binaria del grupo de testeo, en base a la clasificación obtenida con el umbral 0,391. En ambos gráficos, la línea punteada gris representa una clasificación arbitraria entre las dos clases.

6.2.3. Curva PR y Curva VPN-Especificidad

En la Figura 97 (izquierda) se presenta la curva PR y su AUC es igual a 0,753. Este valor puede interpretarse como un resultado óptimo de clasificación de CCRs. Luego se implementa, con el método de *Clasificación según mayoría de predicciones*, la categorización de los tumores de testeo usando el umbral 0,391. La curva PR binaria se observa en la Figura 97 (derecha) y brinda un AUC de 0,906.

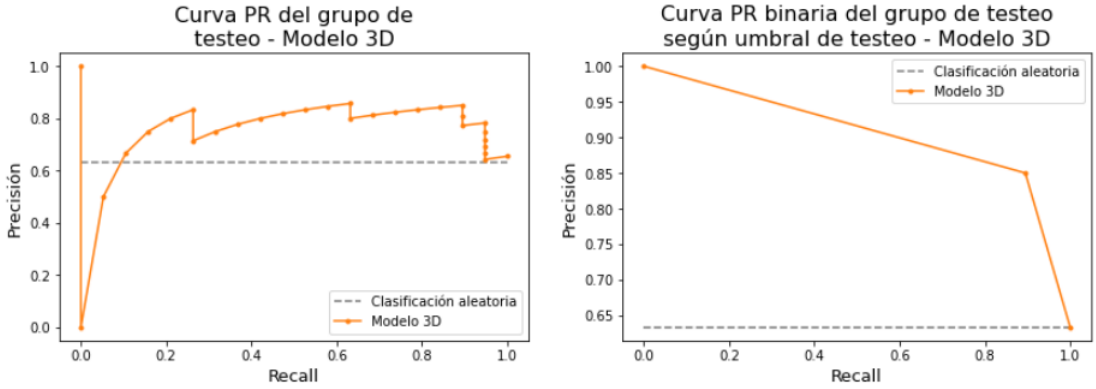

Figura 97: A la izquierda: Curva PR del grupo de testeo, en base a las predicciones realizadas con el modelo 3D. A la derecha: Curva PR binaria del grupo de testeo, según la clasificación utilizando el umbral 0,391. En ambas imágenes, la clasificación arbitraria genera la línea punteada gris; al tratarse de la clase de tumor maligno presenta un valor de 0,633.

Además se crea la curva VPN-Especificidad, expuesta en la Figura 98 (izquierda);

el AUC de este gráfico es 0,690. Después de realizar la clasificación con el umbral mencionado, esta métrica mejora a un valor de 0,814 que representa buenos resultados de categorización de tumores benignos. La curva binaria se expone en la Figura 98 (derecha).

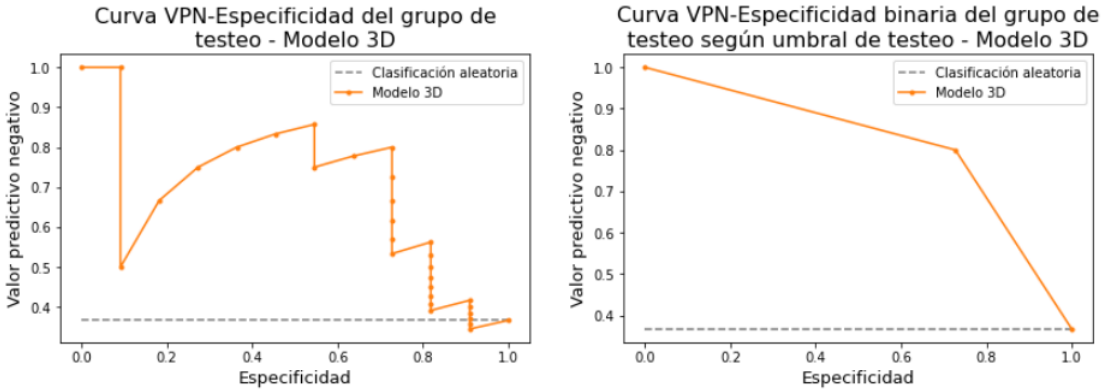

Figura 98: A la izquierda: Curva VPN-Especificidad del grupo de testeo, en base a las predicciones del modelo 3D. A la derecha: Curva VPN-Especificidad binaria del grupo de testeo, según la clasificación adquirida con el umbral 0,391. En ambas imágenes, la clasificación arbitraria genera la línea punteada gris; en el caso de la clase de tumor benigno presenta un valor de 0,367.

6.2.4. Clasificación y métricas obtenidas

Se muestra en la Figura 99 las predicciones binarias del grupo de testeo, obtenidas al utilizar el modelo 3D entrenado y el umbral 0,391. De un total de 19 CCRs, 17 fueron clasificados correctamente; mientras que se reconocieron 8 ONCs de los 11 que conforman el grupo de testeo. En la Tabla 8 se indica la matriz de confusión obtenida en esta prueba.

	Nom_tum	Label	Predicción	Comparación
0	088_CCR	CCR	CCR	True
1	089_CCR	CCR	CCR	True
2	090_CCR	CCR	CCR	True
3	092_CCR	CCR	CCR	True
4	093_CCR	CCR	CCR	True
5	094_CCR	CCR	CCR	True
6	095_CCR	CCR	CCR	True
7	096_CCR	CCR	ONC	False
8	099_CCR	CCR	CCR	True
9	100_CCR	CCR	CCR	True
10	101_CCR	CCR	CCR	True
11	102_CCR	CCR	ONC	False
12	103_CCR	CCR	CCR	True
13	104_CCR	CCR	CCR	True
14	105_CCR	CCR	CCR	True
15	106_CCR	CCR	CCR	True
16	107_CCR	CCR	CCR	True
17	108_CCR	CCR	CCR	True
18	109_CCR	CCR	CCR	True

	Nom_tum	Label	Predicción	Comparación
19	046_ONC	ONC	ONC	True
20	047_ONC	ONC	ONC	True
21	048_ONC	ONC	CCR	False
22	049_ONC	ONC	CCR	False
23	050_ONC	ONC	ONC	True
24	051_ONC	ONC	ONC	True
25	052_ONC	ONC	ONC	True
26	053_ONC	ONC	ONC	True
27	054_ONC	ONC	ONC	True
28	055_ONC	ONC	ONC	True
29	056_ONC	ONC	CCR	False

Figura 99: Resultado de la clasificación del mejor modelo 3D. A la izquierda, tabla con resultados de los CCRs; se clasificaron incorrectamente el CCR_96 y el CCR_102. A la derecha, tabla con las clasificaciones de los ONCs: 3 casos (ONC_48, ONC_49 y ONC_56) fueron identificados como CCRs.

		Clasificación	
		Positiva	*Negativa*
Etiqueta	*Positiva*	17	2
	Negativa	3	8

Tabla 8: Matriz de confusión del grupo de testeo en base a los resultados de clasificación obtenidos usando el modelo 3D. La etiqueta positiva es CCR y la negativa ONC. Los VP, FN, FP y VN presentes en esta matriz, sirven para calcular las métricas.

Por último, en la Tabla 9 se observan los valores de las métricas obtenidas para el grupo de testeo. En este caso se clasificaron de manera incorrecta 3 ONCs, mientras que con el modelo 2D existían sólo 2 ONCs en dicha situación. Esta diferencia ocasiona una pequeña disminución de los valores de las métricas. El índice más afectado es la especificidad, ya que se vincula directamente con la clasificación de ONCs (VN y FP).

Métrica	**Valor**
Accuracy	0,833
Sensibilidad (recall)	0,895
Especificidad	0,727
VPP (precisión)	0,850
VPN	0,800
Valor-F1	0,872
AUC-ROC	0,811
AUC-PR	0,906

Tabla 9: Métricas de evaluación calculadas en base a la clasificación implementada por el modelo 3D.

7. Discusión

En el informe, se describen los múltiples ensayos desarrollados con el fin de determinar la mejor opción para la clasificación de tumores sólidos renales a partir del uso de redes neuronales. El empleo de *Deep Learning* representa un cambio respecto a las propuestas anteriormente ejecutadas por el Departamento de Informática en Salud del Hospital Italiano de Buenos Aires, donde se aplicaron otras técnicas de *Machine Learning* combinadas con *Radiomics*. El uso de aprendizaje profundo posee el beneficio de identificar de forma automática las características significativas de la imagen, aunque esto también genera una limitación de pérdida de interpretabilidad.

Las redes neuronales se entrenaron a través del método de aprendizaje supervisado, mediante la utilización de una base de datos brindada por el Hospital Italiano de Buenos Aires. Dicha base de datos presentó variados desafíos. En primer lugar, disponía de un número bajo de tumores para dividir entre los tres conjuntos de datos (entrenamiento, validación y testeo). El trabajo con cortes 2D aumentó el número de observaciones, sin embargo la cantidad de imágenes de entrada permaneció escasa en comparación con el volumen de datos que normalmente se maneja en algoritmos de aprendizaje profundo.

En segundo lugar, cada neoplasia poseía distinto tamaño, por lo tanto la base de datos se conformó con imágenes de diversas dimensiones; además existía una alta proporción de masas de reducido volumen. En tercer lugar, existió el desafío de poseer un conjunto de observaciones con desbalance en el porcentaje de casos pertenecientes a cada tipo de tumor.

Por último, se sumó las limitaciones asociadas al proceso de segmentación manual: por una cuestión de tiempo la misma se realizó mayoritariamente empleando una sola fase de la tomografía contrastada y se llevó a cabo por un único especialista por masa tumoral. Esto último no considera la variabilidad interobservador. También se desconoce el efecto del programa 3D Slicer en el desempeño del algoritmo y cómo afectaría a la clasificación el empleo de otro programa para la segmentación.

Para comenzar con el estudio, se probaron diferentes metodologías para preprocesar las imágenes, tanto en 2D, 3D como 3C. En cuanto a los dos subtipos de preprocesamiento 2D (sin y con tejido circundante), se considera que la conservación del entorno del tumor no modificó los resultados de manera considerable, es decir, se cree que la red no aprendió representaciones claves para la tarea de clasificación a partir de dicho tejido. Esto podría ser positivo porque es conveniente que la extracción de características se centralice principalmente en el tumor. Otro punto a destacar sobre el preprocesamiento 2D es la exclusión de imágenes con artefactos de cuadriculado para evitar un sesgo en el aprendizaje de la red.

Por otro lado, la obtención de imágenes 3D se desarrolló con la finalidad de conservar las relaciones entre vóxeles de los distintos cortes del eje Z y el preprocesamiento con tres canales (3C) se creó para vincular distintas fases en una misma imagen.

Los factores de diseño se fueron definiendo en el proceso de realización del trabajo, como respuesta a las problemáticas que se presentaron en la implementación de las distintas metodologías. De esta manera, se generaron nuevas posibilidades de pruebas a medida que la investigación progresaba.

Dentro de los factores de diseño de la base de datos 2D, se interpreta como importante la exclusión de cortes con información escasa de la neoplasia para eliminar casos que no propician el aprendizaje de la red. Además se destaca la selección de cortes centrales, la cual permitió que todas las masas aporten una misma cantidad de cortes como entrada de la red, evitando que los tumores de mayor volumen dominen el entrenamiento del modelo debido a su número superior de cortes axiales; la utilización de este factor de diseño mostró mejores resultados en la clasificación del grupo de testeo.

Por otra parte, el empleo de los datos como imagen o matriz numpy brindó resultados similares, ya que esta elección no implica cambios significativos en la aplicación de *Deep Learning*. El balanceo de datos tampoco provocó mejoras en la categorización de tumores, a pesar de ser implementado en estudios anteriores [40]-[41]. Es posible que esto se deba a que el desbalance de la base de datos del Hospital Italiano (66,06 % de CCRs y 33,94 % de ONCs) es menor que el desequilibrio de los conjuntos de datos de los trabajos mencionados.

Con respecto a los factores de diseño del modelo, las pruebas que brindaron los resultados más óptimos se entrenaron con un modelo por fase, empleando únicamente la F_N, y sin utilizar metadatos como entradas adicionales de la red (sólo se trabajó con las imágenes tomográficas de los tumores). Esta última decisión se debe a que, al trabajar con modelos con múltiples entradas, no se evidenciaron modificaciones relevantes en los resultados del grupo de testeo.

El factor de diseño de uso de *parches*, se aplicó para aumentar la cantidad de muestras y focalizarse en la textura del tumor. Sin embargo, no beneficiaron la clasificación; se cree que las dimensiones reducidas de estas estructuras pueden haber afectado la adquisición de jerarquías espaciales entre patrones de la red. Por último, un factor exclusivo para los datos 2D es *Modelos por tamaño del tumor*, donde se desarrollaron ensayos diferenciando en dos grupos los cortes de acuerdo a su área; en estas pruebas no se obtuvieron notables beneficios en el grupo de mayor dimensión.

También se determinaron posibles estructuras de la red neuronal y distintos métodos de evaluación. En trabajos previos [40],[41],[43] suelen emplearse arquitecturas pre-entrenadas para la clasificación de masas renales tumorales, por esta razón se decidió probar las mismas usando técnicas de *Feature extraction* y *Fine-tunning*. Sin embargo, en los ensayos efectuados con modelos con arquitectura propia se adquirieron resultados más favorables; posiblemente debido a que la red pre-entrenada posee más parámetros y con la limitada cantidad de datos disponibles existe un mayor riesgo de *overfitting*.

En el diseño de arquitecturas propias, se emplearon tanto modelos 2D como 3D, lo cual permitió comparar sus respectivos resultados en un mismo trabajo a partir de una base de datos compartida; esta información no fue encontrada en estudios previos. A continuación, se realiza el análisis de las pruebas que dieron los mejores resultados en el grupo de testeo (expuestos en la sección *Resultados*), obtenidas mediante el uso de las redes registradas en el apartado *Modelos utilizados*.

En cuanto al entrenamiento, el modelo 3D presentó una mayor variabilidad punto a punto tanto en la función de costo como en el *accuracy*, con una tendencia de mejora de las métricas más suave que la del modelo 2D. Esto podría deberse a un mayor

overfitting, ya que el trabajo en 3D conlleva el uso de menos datos para entrenar un modelo con mayor cantidad de parámetros (más del doble que los pesos del modelo 2D).

Al emplear el mejor modelo 2D y 3D, se hallaron resultados similares en el grupo de testeo. En el primer caso únicamente se clasificaron de manera incorrecta 2 CCRs y 2 ONCs; con el modelo 3D se suma otro tumor benigno interpretado como maligno. Existen dos neoplasias que fueron mal categorizadas en ambas pruebas (CCR_102 y ONC_56), el resto varía según el modelo utilizado.

Cabe mencionar que considerar un CCR como ONC es un error médico con mayor importancia, ya que pospone la extracción de un tumor maligno del paciente. En los casos analizados, se consiguen igual cantidad de CCRs mal clasificados. Si la situación fuese diferente, habría que examinar las consecuencias de cada categorización y decidir qué modelo provocaría mejores resultados médicos: una red neuronal que brinde mayor sensibilidad, aunque se vincule con una reducción de la especificidad, podría ser la mejor opción.

Todas las métricas calculadas para el conjunto de testeo al realizar las predicciones con el mejor modelo 2D y 3D, presentan valores mayores a 0,8 con excepción de la especificidad obtenida con la red 3D (igual a 0,727), lo que se consideran buenos resultados debido a su cercanía a la unidad. El *accuracy* elevado indica un número alto de masas tumorales clasificadas correctamente. En particular, las medidas próximas a 1 de la sensibilidad, el VPP y el Valor-F1 pueden interpretarse como una categorización satisfactoria de los CCRs; en cambio, los valores cercanos a 1 de la especificidad y el VPN hablan de una buena distinción de ONCs.

Por otra parte, las AUC de las gráficas corresponden a métricas generales de desempeño, cuyos valores cercanos a la unidad simbolizan una mayor efectividad del modelo para el cumplimiento de la tarea. Particularmente, el AUC-ROC elevado refleja una capacidad discriminativa óptima entre clases; el AUC-PR alto se interpreta como un buen resultado de clasificación de masas malignas (de importancia médica para la extracción de los mismos) y el AUC de la gráfica de VPN-Especificidad próximo a 1 representa una categorización correcta de ONCs (resultado relevante ya que esta clase es minoritaria).

Respecto a las pruebas que utilizan datos con preprocesamiento de tres canales (3C), el ensayo que dio mejores resultados se describe en el *Anexo 2*; en este caso se consiguió un *accuracy* de 0,815, una sensibilidad de 0,812 y una especificidad de 0,818. Si bien existen estudios previos, como los mencionados en la sección *Estado del arte* [40],[41],[43], donde los resultados muestran métricas más elevadas a las expuestas en este informe, se considera que los valores alcanzados en el presente

proyecto son óptimos.

Un problema alcanzado en el desarrollo de este proyecto es el empleo de un umbral basado en el conjunto de testeo. Lo ideal es lograr obtener el umbral en base al grupo de validación, pero se determinó que el mismo podría no ser representativo del conjunto total de datos, porque no brindó resultados positivos en los gráficos de entrenamiento ni en la curva ROC. Se considera que la limitada cantidad de datos presente en dicho grupo podría influir en esta situación.

La configuración de un umbral basado en el grupo de testeo puede implicar un sobreajuste a este conjunto de muestras en particular, es decir, que el desempeño diagnóstico reportado para este grupo podría no mantenerse en nuevas muestras. Sin embargo, este riesgo disminuye en la medida en que el grupo de testeo sea representativo de la población real.

Actualmente, el Hospital Italiano está ampliando la cantidad de imágenes tomográficas de casos de estudio con masas renales (segmentadas y etiquetadas), para realizar una validación prospectiva. A futuro, se podrían clasificar dichos datos con los dos mejores modelos entrenados y sus respectivos umbrales elegidos, para poder analizar las métricas de evaluación en un nuevo conjunto. Esto también podría ayudar a verificar la representatividad del grupo de testeo original respecto a la población total.

8. Conclusión

En este proyecto se implementó la aplicación de redes neuronales para la clasificación de tumores sólidos renales, en base a tomografías computadas con contraste. Dicho planteo difiere de propuestas anteriores realizadas por el Departamento de Informática en Salud del Hospital Italiano de Buenos Aires, en las cuales se empleaban técnicas de *Machine Learning* combinadas con *Radiomics*, método muy utilizado para la diferenciación entre tipos de tumores.

El uso de *Deep Learning* es novedoso dentro del entorno médico y ha aumentado en los últimos años. Este trabajo muestra que el empleo de redes neuronales puede ser útil para el desarrollo de la tarea de distinción entre masas tumorales malignas y benignas. Se considera que se cumple el objetivo de este estudio, la creación de un sistema automático de clasificación binaria de tumores renales mediante el empleo de una CNN entrenada con aprendizaje profundo, debido a los resultados óptimos conseguidos en el conjunto de testeo al utilizar el mejor modelo 2D y 3D.

La herramienta podría colaborar en la toma de decisión de los médicos, sin embargo al no obtenerse una sensibilidad del 100 % existe un riesgo de falta de identificación de CCRs. Antes de integrar la misma al curso de trabajo de la institución debe superar varias etapas de ensayos de funcionamiento; en primer lugar se evaluarán los resultados del modelo en un nuevo grupo de datos que está en proceso de elaboración.

Otro trabajo vinculado que se podría implementar a futuro, es el empleo de redes neuronales para adquirir la segmentación automática de los tumores. Esta función no es esencial pero facilitaría el trabajo de los radiólogos y además las máscaras resultantes podrían usarse para la tarea de clasificación.

Por último, es importante destacar que con el desarrollo del Proyecto Final de Carrera de Bioingeniería se enriquecieron ampliamente los conocimientos previamente adquiridos sobre la disciplina de redes neuronales. Además, se reforzó el aprendizaje mediante la implementación práctica de un caso de estudio complejo y aplicado al mundo real.

9. Anexo 1

En esta sección se visualizan tres tablas; cada una incluye información sobre distintos conjuntos de pruebas realizados a partir de datos adquiridos con un tipo de preprocesamiento: 2D, 3D y 3C. En cada grupo de ensayos, se selecciona el modelo que brinda el mejor resultado en el grupo de testeo (*test*) y se registra la sensibilidad y especificidad de la clasificación. Además, se menciona el mejor *accuracy* alcanzado en el entrenamiento de dicho modelo, tanto para el conjunto de entrenamiento (*train*) como para el de validación (*validation*).

Cada fila de una tabla, perteneciente a un conjunto de ensayos, puede incluir más de una prueba. Entre las mismas existe variaciones en la estructura del modelo, en la elección de hiperparámetros vinculados con *data augmentation* o el entrenamiento (por ejemplo: LR, tamaño del *batch*, cantidad de *epochs*, uso de *callbacks*). Esta información no se incluye en las tablas para brindar un resumen más claro y relevante. En verde se resaltan las filas con las características del modelo entrenado, a partir del cual proviene el mejor resultado de cada conjunto de pruebas.

Tabla de pruebas empleando cortes 2D

Conjunto de pruebas	***Preprocesa-miento/ Imagen o numpy***	***Tamaño de entrada***	***Factores de diseño de la base de datos***			***Factores de diseño del modelo***				***Mejor resultado***	
			Exclusión de datos de entrada	*Selección de cortes centrales*	*Balanceado de base de datos*	*Parches como entradas*	*Entradas adicionales de la red*	*Modelos por fase del tumor*	*Modelos por tamaño del tumor*	*Mayor accuracy train y val.*	*Sensibilidad y especificidad en testeo*
1	Contorno por corte/ Imagen	***50x50***	No	No	**Sí**	No	No	No	No	Train: 0,802 Validation: 0,619	Test S: 0,864 E: 0,364
		50x50	No	No	No	No	No	No	No		
		50x50	No	No	**Sí**	No	**Fase**	No	No		
		25x25	No	No	**Sí**	No	No	No	No		
2	Contorno por tumor/ Imagen	***50x50***	No	No	**Sí**	No	No	No	No	Train: 0,811 Validation: 0,623	Test S: 0,818 E: 0,273
		50x50	No	No	**Sí**	No	**Fase**	No	No		
		50x50	No	No	No	No	No	No	No		
3	Entorno por corte/ Imagen	**50x50**	No	No	**Sí**	No	No	No	No	Train: 0,816 Validation: 0,579	Test S: 0,364 E: 0,636
		50x50	No	No	**Sí**	No	**Fase**	No	No		
4	Contorno por corte/ Imagen	***50x50***	No	No	No	No	No	**FE**	No	Train: 0,852 Validation: 0,639	Test S: 0,947 E: 0,364
		50x50	No	No	No	No	No	**FC**	No		
5	Contorno por corte/ Numpy	***50x50***	No	No	**Sí**	No	No	No	No	Train: 0,758 Validation: 0,637	Test S: 0,864 E: 0,273
		50x50	No	No	No	No	No	No	No		
		75x75	No	No	No	No	No	No	No		
		25x25	No	No	No	No	No	No	No		
6	Contorno por corte/ Numpy	**50x50**	No	No	No	No	No	**FC**	No	Train: 0,854 Validation: 0,682	Test S: 0,789 E: 0,727 Aclaración: para FC existen 3 CCRs faltantes
		50x50	No	No	No	No	No	**FE**	No		
		50x50	No	No	No	No	No	**FSC**	No		
7	Contorno por corte/ Numpy	**50x50**	**Por área (< 20x20)**	No	No	No	No	No	No	Train: 0,762 Validation: 0,471	Test S: 0,864 E: 0,091
		50x50	**Por área (< 20x20)**	No	No	No	**Área**	No	No		
		25x25	**Por área (< 20x20)**	No	No	No	**Área**	No	No		
8	Contorno por corte/ Numpy	**50x50**	**Por ubicación (> 50)**	No	No	No	No	No	No	Train: 0,873 Validation: 0,688	Testeo S: 0,864 E: 0,273
		50x50	**Por ubicación (> 20)**	No	No	No	No	No	No		
9	Contorno por corte/ Numpy	**20x20**	No	No	No	**Sí**	No	No	No	Train: 0,723 Validation: 0,556	Test S: 0,364 E: 0,727
		15x15	No	No	No	**Sí**	No	No	No		
10	Contorno por corte/ Numpy	**50x50**	No	**Sí**	No	No	No	**FN**	No	Train: 0,718 Validation: 0,783	**Modelo desarrollado en resultados** Test S: 0,895 E: 0,818 Aclaración: para FN existen 3 CCRs faltantes

Tabla de pruebas empleando tensores 3D

Conjunto de pruebas	***Tamaño de entrada***	***Factores de diseño de la base de datos***	***Factores de diseño del modelo***			***Mejor resultado***		
		Solución del error de cuadriculado	*Parches como entradas*	*Entradas adicionales*	*Modelos por fase del tumor*	*Mayor accuracy train/val*	*Sensibilidad y especificidad en testeo → mayoría de predicciones*	*Sensibilidad y especificidad en testeo → mejor predicción*
1	**50x50x50**	No	No	No	No	Train: 0,810 Validation: 0,700	Test S: 0,773 E: 0,454	Test S: 0,909 E: 0,545
	100x100x100	No	No	No	No			
	50x50x215	No	No	No	No			
2	**50x50x50**	No	No	**Fase**	No	Train: 0,824 Validation: 0,733	Test S: 0,909 E: 0,091	Test S: 0,954 E: 0,091
3	**5x5x5**	No	**Sí**	**Fase**	No	Train: 0,544 Validation: 0,620	Test S: 0,182 E: 0,909	No empleado al trabajar con parches
	8x8x5	No	**Sí**	**Fase**	No			
	10x10x5	No	**Sí**	**Fase**	No			
	9x9x9	No	**Sí**	**Fase**	No			
4	**5x5x5**	No	**Sí**	No	No	Train: 0,547 Validation: 0,667	Testeo S: 0,454 E: 0,727	No empleado al trabajar con parches
5	**50x50x50**	**Sí**	No	No	No	Train: 0,714 Validation: 0,717	Test S: 0,864 E: 0,364	Test S: 0,727 E: 0,454
6	**50x50x50**	**Sí**	No	No	**FN**	Train: 0,754 Validation: 0,900	**Modelo desarrollado en resultados** Test S: 0,895 E: 0,727 Aclaración: para FN existen 3 CCRs faltantes	No empleado al trabajar con una única fase
	40x40x40	**Sí**	No	No	**FN**			

Tabla de pruebas empleando imágenes de 3C

Conjunto de pruebas	***Tamaño de entrada***	***Número de puestos***	***Mejor resultado***	
			Mayor accuracy train/val	*Sensibilidad y especificidad en testeo*
1	**224x224x3**	**1**	Train: 0,992 Validation: 0,800	Testeo S: 0,750 E: 0,636
2	**224x224x3**	**5**	Train: 0,738 Validation: 0,737	Testeo S: 0,687 E: 0,636
3	**100x100x3**	**1**	Train: 0,858 Validation: 0,771	Testeo S: 0,812 E: 0,727
4	**100x100x3**	**5**	Train: 0,785 Validation: 0,737	Testeo S: 0,750 E: 0,636

10. Anexo 2

En este Anexo se evalúa el mejor resultado obtenido al trabajar con la base de datos con imágenes de tres canales (3C). Dicha información no fue incluida en la sección *Resultados* para no extender la misma, ya que en este caso las métricas son inferiores y se excluyen más tumores del grupo de testeo debido a falta de fases tomográficas.

10.0.1. Modelo 3C y entrenamiento

La arquitectura de la CNN contiene una capa de entrada (**Input**), dos bloques convolucionales y un bloque con capas densas. En la Figura 100 se esquematiza el modelo, el cual posee 53.377 parámetros entrenables.

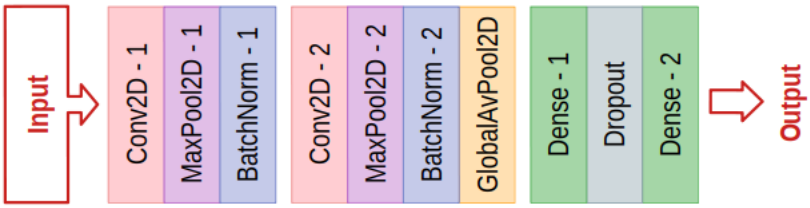

Figura 100: Esquema de la red neuronal con arquitectura propia, empleada en el entrenamiento con imágenes de 3C.

En la primera capa convolucional 2D se aplican 32 filtros y en la segunda, 64 filtros. En ambas, se utilizan *kernels* de tamaño 3x3 para la operación de convolución, con un *stride* igual a 1 y sin implementar *padding*. Luego se emplea la función de activación ReLU. Por otra parte, las capas *Max Pooling* emplean un *kernel* de tamaño 2x2. Tampoco se usa *padding* y el *stride* es de valor 2.

El bloque de clasificación posee una capa densa de 512 unidades y función de activación ReLU, seguida por una capa *Dropout* (con *dropout rate* del 30 %). Para finalizar, se utiliza una capa densa con 1 unidad y función de activación sigmoidea, que brinda la salida de la red.

Para el entrenamiento del modelo, nuevamente se usa la entropía cruzada binaria como función de costo y el *accuracy* como métrica. La configuración de los hiperparámetros se muestra en la Tabla 10. Cabe aclarar que en esta prueba únicamente se utilizan los cortes mayores del tumor para generar las imágenes 3C (es decir, eligiendo el puesto 1), en consecuencia la base de datos es escasa y se usan *batchs* de tamaño pequeño.

Dimensiones de entrada	Optimizador	Learning Rate	Tamaño del batch	Número de epochs	Callbacks
(100,100,3)	RMSprop	0,001	5	50	Model Checkpoint

Tabla 10: Configuración de hiperparámetros de la prueba 3C que causó los mejores resultados del grupo de testeo.

10.0.2. Definición de umbral y curva ROC

El grupo de validación no brinda un umbral que beneficia la clasificación. Entonces se emplea un umbral que provoca el mayor valor del índice de Youden a partir del conjunto de testeo. Esto ocurre cuando el umbral presenta un valor de 0,615, consiguiendo un índice de Youden de 0,539. En la Figura 101, se muestran la curva ROC (con AUC 0,716) y la curva ROC binarizada usando el umbral 0,615 (con AUC 0,770).

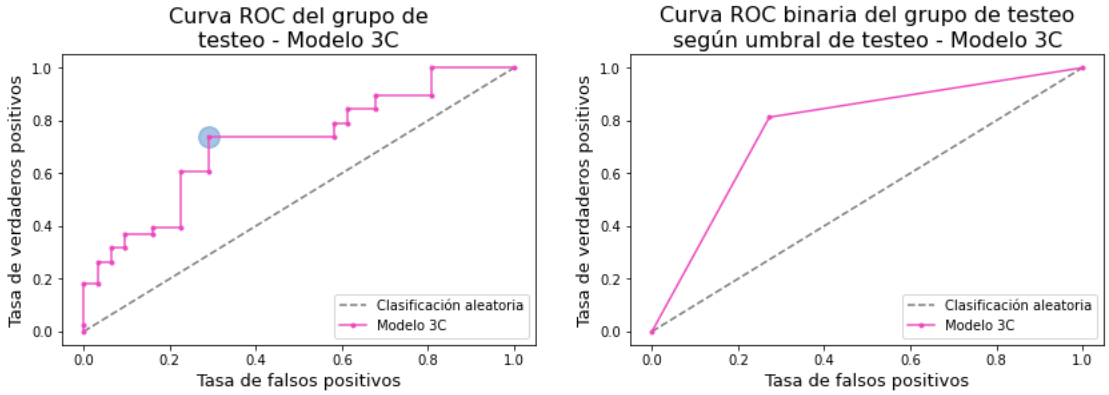

Figura 101: A la izquierda: Curva ROC del grupo de testeo, en base a las predicciones del modelo 3C; el círculo azul marca el punto de la gráfica para el cual se obtiene el mayor índice de Youden. A la derecha: Curva ROC binaria del grupo de testeo, en base a la clasificación obtenida con el umbral 0,615. En ambos gráficos, la línea punteada gris representa una clasificación arbitraria entre las dos clases.

10.0.3. Matriz de confusión y métricas

Es importante recordar que en el preprocesamiento 3C los casos con menos de tres fases son excluidos, debido a la imposibilidad de generar imágenes 3C. Por esta razón, existen 6 tumores malignos que no pueden representarse con este preprocesamiento.

Entonces, el conjunto de testeo está integrado por 11 ONCs y sólo 16 CCRs.

La matriz de confusión se muestra en la Tabla 11. Se emplea el umbral 0,615 para la clasificación en dos clases. Se identificaron de manera correcta 13 CCRs (VP) y 8 ONCs (VN). En cambio, 3 neoplasias malignas y 3 masas tumorales benignas se categorizaron incorrectamente (respectivamente FN y FP).

		Clasificación	
		Positiva	*Negativa*
Etiqueta	*Positiva*	13	3
	Negativa	3	8

Tabla 11: Matriz de confusión del grupo de testeo según los resultados de clasificación obtenidos usando el modelo entrenado con imágenes 3C y el umbral 0,615. Se recuerda que se considera positiva a la etiqueta de CCR y negativa a ONC.

Se computaron las métricas de evaluación para este ensayo. Las mismas se exponen en la Tabla 12. Todas las medidas presentan valores mayores que 0,7.

Métrica	**Valor**
Accuracy	0,778
Sensibilidad (recall)	0,812
Especificidad	0,727
VPP (precisión)	0,812
VPN	0,727
Valor-F1	0,812
AUC-ROC	0,770
AUC-PR	0,868

Tabla 12: Métricas de evaluación de la clasificación realizada con el modelo 3C y el umbral 0,615.

Referencias

[1] American Cancer Society. About Kidney Cancer

https://www.cancer.org/cancer/kidney-cancer/about/what-is-kidney-cancer.html

[2] Moch, Cubilla, Humphrey, Reuter & Ulbright (2016). *The 2016 World Health Organization classification of tumours of the urinary system and male genital organs-part A: renal, penile, and testicular tumours.* European Urology. Volumen 70 (páginas 93–105). DOI: 10.1016/j.eururo.2016.02.029

[3] American Cancer Society.

https://acsjournals.onlinelibrary.wiley.com/doi/epdf/10.3322/caac.21660

[4] International Agency for Research on Cancer

https://gco.iarc.fr/today/data/factsheets/populations/32-argentina-fact-sheets.pdf

[5] Sociedad Española de Oncología Médica

https://seom.org/informacion-sobre-el-cancer/que-es-el-cancer-y-como-se-desarrolla

[6] National Institutes of Health

https://www.cancer.gov/about-cancer/understanding/what-is-cancer

[7] UROCIR

http://www.urocir.com/cancer-de-rinon/

[8] Base de datos brindada por el Hospital Italiano de Buenos Aires - Departamento de Diagnóstico por imágenes - Urología

[9] Van Rossum, G., & Drake Jr, F. L. (1995). Python reference manual. Centrum voor Wiskunde en Informatica Amsterdam.

[10] Chollet, F., & others. (2015). Keras. GitHub. Retrieved from https://github.com/fchollet/keras

[11] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, & others. *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015. Software available from tensorflow.org.

[12] Francois Chollet (2018). *Deep Learning with Python.* Manning Shelter Island

[13] Candelaria Mosquera, María Agustina Ricci Lara, Facundo Nahuel Díaz (2021). *Inteligencia Artificial en Imágenes Médicas. De la teoría a la aplicación.* Magister

[14] TIBCO Software Inc.
https://www.tibco.com/reference-center/what-is-a-neural-network

[15] Keras. Functional API
https://keras.io/guides/functional_api/

[16] Keras. Layers
https://keras.io/api/layers/

[17] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei, *ImageNet: A large-scale hierarchical image database.* 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[18] Shiv Vignesh. *The Perfect Fit for a DNN.* Medium.
https://medium.com/analytics-vidhya/the-perfect-fit-for-a-dnn-596954c9ea39

[19] TensorFlow - Aumento de datos
https://www.tensorflow.org/tutorials/images/data_augmentation

[20] Documentación de Albumentations
https://albumentations.ai/

[21] Buslaev, Parinov, Khvedchenya, Iglovikov & Kalinin (2018). *Albumentations: fast and flexible image augmentations.* arXiv preprint: 1809.06839.

[22] McKinney, W. & others. (2010). *Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference* (Vol. 445, pp. 51–56).

[23] Sebastian Raschka. *Gradient Descent and Stochastic Gradient Descent.* Mlxtend. *http://rasbt.github.io/mlxtend/user_guide/general_concepts/gradient-optimization/*

[24] Carolina Bento. *Stochastic Gradient Descent explained in real life.* Medium. *https://towardsdatascience.com/stochastic-gradient-descent-explained-in-real-life*

[25] Andrew Ng. *Machine Learning course.* Stanford University.

[26] Keras. Callback Model Checkpoint
https://keras.io/api/callbacks/model_checkpoint/

[27] Keras. Callback Learning Rate Scheduler
https://keras.io/api/callbacks/learning_rate_scheduler/

[28] Prateek Joshi (2017). *Artificial Intelligence with Python.* Packt

[29] Singh, Elhoseny & Elngar (2021). *Machine Learning and the Internet of Medical Things in Healthcare.* Academic Press

[30] Griner, Mayewski, Mushlin & Greenland (1981). *Selection and interpretation of diagnostic tests and procedures. Principles and applications.* Ann Intern Med.

[31] Pita-Fernández & Pértegas-Díaz (2003). *Pruebas diagnósticas: Sensibilidad y especificidad.* Cad Aten Primaria.

[32] López de Ullibarri Galparsoro & Píta Fernández (1998). *Curvas ROC.* Cad Aten Primaria. Unidad de Epidemiología Clínica y Bioestadística.

[33] Cerda & Cifuentes (2012). *Uso de curvas ROC en investigación clínica. Aspectos teórico-prácticos.* Revista chilena de infectología. Volumen 29 (Número 2) http://dx.doi.org/10.4067/S0716-10182012000200003

[34] Alex Yartsev. *The receiver operating characteristic (ROC) curve. https://derangedphysiology.com/cicm-primary-exam/required-reading/research-methods-and-statistics/Chapter %203.0.5/receiver-operating-characteristic-roc-curve*

[35] Davis & Goadrich (2006). *The relationship between precision-recall and roc curves.* ICML '06: Proceedings of the 23rd International Conference on Machine learning (páginas 233–240). https://doi.org/10.1145/1143844.1143874

[36] Hsieh & Jiang (2003). *Computed Tomography: Principles, Design, Artifacts, and Recent Advances.* SPIE Press Books

[37] Mayo Clinic - Urografía por tomografía computarizada
https://www.mayoclinic.org/es-es/tests-procedures/ct-urogram/about/pac-20393602

[38] Yuh & Cohan (1999). *Different phases of renal enhancement: role in detecting and characterizing renal masses during helical CT.* American Journal of Roentgenology. Volumen 173 (Número 3).

[39] Sheth & Fishman (2004). *Multi-Detector Row CT of the Kidneys and Urinary Tract: Techniques and Applications in the Diagnosis of Benign Diseases.* Radiological Society of North America (RSNA) - RadioGraphics. https://doi.org/10.1148/rg.e20

[40] Pedersen, Andersen, Christiansen & Azawi (2020). *Classification of renal tumour using convolutional neural networks to detect oncocytoma.* European Journal of Radiology. Volumen 133 - Elsevier Wordmark. https://doi.org/10.1016/j.ejrad.2020.109343

[41] Pan, Yang, Wang, Lu, Zhou, Kong, Tang, Zhu, Dillenseger, Shu & Coatrieux (2019). *A multi-task convolutional neural network for renal tumor segmentation and classification using multi-phasic CT images.* IEEE (Institute of Electrical and Electronics Engineers) International Conference on Image Processing. DOI: 10.1109/ICIP.2019.8802924

[42] Bhatnagar, Gill & Ghosh (2020). *Drone Image Segmentation Using Machine and Deep Learning for Mapping Raised Bog Vegetation Communities.* Remote Sensing. Volumen 12. https://doi.org/10.3390/rs12162602

[43] Uhm, Jung, Choi, Shin, Yoo, Oh, Kim, Kim, Lee, Youn, Hong & Ko (2021). *Deep Learning for end-to-end kidney cancer diagnosis on multi-phase abdominal computed tomography.* NPJ Precision Oncology - Nature. https://doi.org/10.1038/s41698-021-00195-y

[44] Han, Hwang & Lee (2019). *The Classification of Renal Cancer in 3-Phase CT Images Using a Deep Learning Method.* Journal of Digital Imaging. Volumen 32 (páginas 638 – 643). https://doi.org/10.1007/s10278-019-00230-2

[45] Fedorov, Beichel, Kalpathy-Cramer, Finet, Fillion-Robin, Pujol, Bauer, Jennings, Fennessy, Sonka, Buatti, Aylward, Miller, Pieper S., Kikinis. *3D Slicer as an Image Computing Platform for the Quantitative Imaging Network.* Magn Reson Imaging. 2012 Nov; 30(9):1323-41. PMID: 22770690. PMCID: PMC3466397.

[46] Documentación Slicer

https://slicer.readthedocs.io/en/latest/user_guide/image_segmentation.html

[47] Nibabel

Brett, M. et al. nipy/nibabel: 3.0.1. (2020). Zenodo

https://doi.org/10.5281/zenodo.3628482

[48] Cox RW, Ashburner J, Breman H, Fissell K, Haselgrove C, Holmes CJ, Lancaster JL, Rex DE, Smith SM, Woodward JB, Strother SC (2004). *A (sort of) new image data format standard: NIfTI-1*. Presented at the 10th Annual Meeting of the Organization for Human Brain Mapping.

[49] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D. & Oliphant, T. E. (2020). *Array programming with NumPy.* Nature, 585, 357–362. https://doi.org/10.1038/s41586-020-2649-2

[50] Qingge, Jie, Wenjie & Yankui (2019). *Optimized Deep Convolutional Neural Networks for Identification of Macular Diseases from Optical Coherence Tomography Images.* Algorithms. Volumen 12. DOI:10.3390/a12030051