UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Diseño y desarrollo de un sistema de ayuda al diagnóstico para el cáncer de próstata en imágenes de resonancia magnética con adaptación inter-dominio no supervisada.

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

AUTOR/A: Camacho Sanchez, Miguel

Tutor/a: Naranjo Ornedo, Valeriana

Cotutor/a: Silva Rodríguez, Julio José

CURSO ACADÉMICO: 2021/2022

Abstract

Prostate cancer is globally the second most prevalent type of cancer. In 2018, 1.3 million patients were diagnosed and the number of new annual cases is estimated to increase by 40.2 % by 2030. This pathology is diagnosed from visual analysis of biopsies by the pathologist and classification of tissue differentiation according to the Gleason scale. However, obtaining biopsies is an invasive process, with clinical risks, which only covers a specific region of the organ. For this reason, in recent years, the clinical need has arisen to obtain the diagnosis according to the Gleason scale by means of non-invasive imaging tests, such as magnetic resonance imaging. To automate this process, the development of deep learning-based diagnostic aid systems has recently been proposed. However, the patterns observable in biopsies do not have a clear differentiation in magnetic resonance imaging, and the variability of image acquisition systems pose a challenge for the use of these diagnostic aid systems.

Therefore, the objective of this TFG is the development of deep learning models capable of aiding in the diagnosis of magnetic resonance imaging. For this purpose, use will be made of a public database of magnetic resonance images whose patients have been diagnosed by means of biopsies. Methodologically, it is intended to segment in first instance the glandular area, and to prepare an inter-domain calibration system by means of adversarial learning. In second instance, the glandular volumes will be classified to obtain the ISUP grade diagnosed by experts, by means of 3D convolutional neural networks. The work will include database preparation, algorithm development and both quantitative and qualitative validation of the results obtained, which have obtained a DICE of 0.8 for segmentation, a Kappa coefficient of 0.23 for 5 classes and an F1 score of 0.55 for the binary case, demonstrating that there is still a long way to go but that significant progress is possible.

Resumen

El cáncer de próstata es a nivel mundial el segundo tipo de cáncer con mayor prevalencia. En 2018 se diagnosticaron 1.3 millones de pacientes y se estima que el número de casos anuales nuevos aumente en un 40.2 % en 2030. Esta patología es diagnosticada a partir del análisis visual de biopsias por medio del patólogo y la clasificación de la diferenciación del tejido según la escala Gleason. Sin embargo, la obtención de biopsias es un proceso invasivo, con riesgos clínicos, que únicamente abarca una región concreta del órgano. Por ello, en los últimos años, ha surgido la necesidad clínica de obtener el diagnóstico según la escala Gleason por medio de pruebas de imagen no invasiva, como es el caso de la resonancia magnética. Para automatizar este proceso, recientemente se ha propuesto el desarrollo de sistemas de ayuda al diagnóstico basados en deep learning. Sin embargo, los patrones observables en las biopsias no tienen una diferenciación clara en la imagen de resonancia magnética, y la variabilidad de los sistemas de adquisición de imagen suponen un desafío para el uso de estos sistemas de ayuda al diagnóstico.

Por ello, el objetivo de este TFG es el desarrollo de modelos de deep learning capaces de ayudar en el diagnóstico de resonancias magnéticas. Para ello, se hará uso de una base de datos pública de imágenes de resonancia magnética cuyos pacientes han sido diagnosticados por medio de biopsias. Metodológicamente, se pretende segmentar en primera instancia la zona glandular, y preparar un sistema de calibración inter-dominio por medio de aprendizaje adversarial. En segunda instancia, los volúmenes glandulares serán clasificados para obtener el grado ISUP diagnosticado por expertos, mediante redes neuronales convolucionales 3D. El trabajo incluirá la preparación de la base de datos, desarrollo de algoritmos y validación tanto cuantitativa como cualitativa de los resultados obtenidos, los cuáles han obtenido un DICE de 0.8 para la segmentación, un coeficiente Kappa de 0.23 para 5 clases y un F1 score de 0.55 para el caso binario, demostrando que todavía queda mucho camino por recorrer pero que es posible lograr avances significativos.

Resum

El càncer de pròstata és a nivell mundial el segon tipus de càncer amb major prevalença. En 2018 es van diagnosticar 1.3 milions de pacients i s'estima que el nombre de casos anuals nous augmente en un 40.2 % en 2030. Aquesta patologia és diagnosticada a partir de l'anàlisi visual de biòpsies per mitjà del patòleg i la classificació de la diferenciació del teixit segons l'escala Gleason. No obstant això, l'obtenció de biòpsies és un procés invasiu, amb riscos clínics, que únicament abasta una regió concreta de l'òrgan. Per això, en els últims anys, ha sorgit la necessitat clínica d'obtindre el diagnòstic segons l'escala Gleason per mitjà de proves d'imatge no invasiva, com és el cas de la ressonància magnètica. Per a automatitzar aquest procés, recentment s'ha proposat el desenvolupament de sistemes d'ajuda al diagnòstic basats en deep learning. No obstant això, els patrons observables en les biòpsies no tenen una diferenciació clara en la imatge de ressonància magnètica, i la variabilitat dels sistemes d'adquisició d'imatge suposen un desafiament per a l'ús d'aquests sistemes d'ajuda al diagnòstic.

Per això, l'objectiu d'aquest TFG és el desenvolupament de models de deep learning capaços d'ajudar en el diagnòstic de ressonàncies magnètiques. Per a això, es farà ús d'una base de dades pública d'imatges de ressonància magnètica els pacients de la qual han sigut diagnosticats per mitjà de biòpsies. Metodològicament, es pretén segmentar en primera instància la zona glandular, i preparar un sistema de calibratge inter-domini per mitjà d'aprenentatge adversarial. En segona instància, els volums glandulars seran classificats per a obtindre el grau ISUP diagnosticat per experts, mitjançant xarxes neuronals convolucionals 3D. El treball inclourà la preparació de la base de dades, desenvolupament d'algorismes i validació tant quantitativa com qualitativa dels resultats obtinguts, els quals han obtingut un DIU de 0.8 per a la segmentació, un coeficient Kappa de 0.23 per a 5 classes i un F1 score de 0.55 per al cas binari, demostrant que encara queda molt camí per recórrer però que és possible aconseguir avanços significatius.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

El término *Inteligencia Artificial* (IA) fue acuñado por primera vez en la Conferencia de Dartmouth por el científico John McCarthy, quien lo definió como la ciencia de hacer actuar a una máquina de manera que pueda ser considerada como inteligente si lo hiciera un ser humano [1]. Sin embargo, hay discrepancias con respecto a la definición formal del término.

Aunque actualmente el campo de la inteligencia artificial esté en auge, no siempre fue así. Desde su definición, hubo varios periodos de décadas en los que apenas se financiaban investigaciones y su popularidad estaba en declive, es lo que se conoce como el *invierno de la IA*. Afortunadamente, y sobre todo en las dos últimas décadas, el crecimiento de la IA ha sido exponencial, en parte gracias a la aparición de las redes neuronales y de los algoritmos de Deep Learning [2], pero también gracias al desarrollo tecnológico de los procesadores que permiten la implantación de esos algoritmos. Así como las redes convolucionales o, más recientemente, los *Transformers* y las *GANs*, con arquitecturas como GPT-3 o DALL-E, están haciendo avances que hace décadas parecían impensables.

Sin emargo, el campo que aquí nos ocupa es el de la *Visión por Computador*, el cual trata de simular ciertos procesos cognitivos del ser humano a la hora de analizar imágenes digitales. Es un campo muy extenso y con numerosas aplicaciones, como, por ejemplo, la detección de caras u objetos o, la que trata este trabajo, la detección de enfermedades en imágenes médicas.

1.1. Cáncer de próstata

La próstata es una glándula del sistema reproductor masculino, ubicada a la salida de la vejiga urinaria. Su función es la de segregar sustancias que protegen al semen, así como ejercer presión para que este sea expulsado por la uretra. A día de hoy, no se conocen con suficiente claridad las causas que pueden degenerar en cáncer de próstata, pero hay factores que se sabe que aumentan el riesgo, como pueden ser la genética, como la etnia africana, o una dieta alta en grasas y con escasa fruta o verdura.

Actualmente, el cáncer de próstata (PCa) es el segundo tipo de cáncer más común entre los varones. Se estima que, en todo el mundo, en 2020 fueron diagnosticadas con cáncer de próstata 1.5 millones de personas, siendo el cuarto cáncer más diagnosticado en el mundo y el primero entre los hombres en España. Afortunadamente, la tasa de supervivencia a 5 y 10 años de ser detectado es del 98 %. Sin embargo, es la segunda causa de muerte por cáncer entre los varones en Estados Unidos [3].

El cáncer se produce por la rápida y descontrolada expansión de las células de tejidos u órganos, fuera de su ciclo normal de crecimiento, formando tumores. Esta expansión incontrolada esta provocada por una mutación en el material genético de las células. Además, estás células cancerígenas, pueden colonizar y afectar a otros tejidos u órganos, distintos de donde se originaron, lo que se conoce como metástasis, y es la causa de muerte del 92 % de los cánceres detectados.

El cáncer de próstata es mucho más común a partir de los 65 años, y rara vez se produce antes de los 50. Por esto, para garantizar una detección temprana, ya que es un cáncer que se desarrola sin síntomas hasta que esta muy avanzado, existen pruebas que se realizan a los varones a partir de los 50 años [4]. La manera más común de hacer una detección precoz es con la prueba del antígeno prostático específico (PSA) en sangre, sin embargo, la PSA no presenta un diagnostico definitivo, ya que puede dar falsos positivos. Asimismo, se necesita la biopsia para saber la severidad de la patología. Otra técnica es mediante una exploración física o tacto rectal, que aun siendo menos efectiva, se una en conjunto con la primera. Otra forma de detección es mediante una resonancia magnética (RM), que es la que nos ocupa aquí y se explica en la siguiente sección.

1.2. Imágenes de resonancias magnéticas

Las primeras *Imágenes de Resonancias Magnéticas* (IRM) las realizó Herman Carr en 1952 [5]. Esta técnica tiene como finalidad conocer la composición del cuerpo a analizar. Consiste en alinear los momentos de los núcleos de los átomos de hidrógeno mediante un campo magnético intenso. A continuación, se emiten pulsos de radiofrecuencia (RF) que hacen resonar los núcleos alineados, produciendo estos un campo magnético rotacional detectable.

Hoy en día es una técnica ampliamente utilizada, y juega un papel fundamental en el diagnóstico de tumores y cáncer, sobre todo para el de próstata. También sirve de ayuda para guiar una biopsia, identificando que área de la próstata presenta tejidos que serían de interés extirparlos para analizarlos bajo microscopio [6].

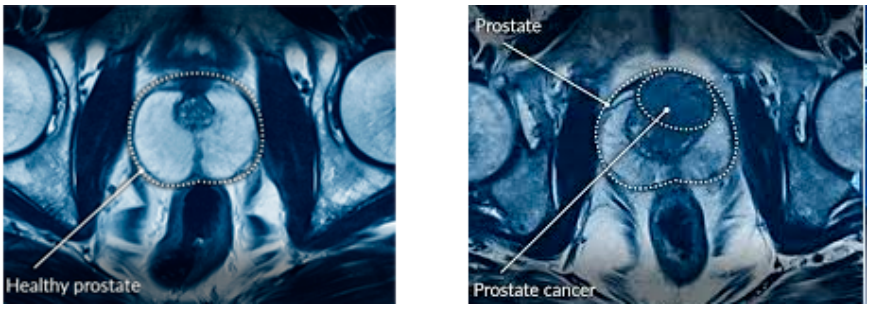


Figura 1.1: Comparación de una próstata sana con una con cáncer por medio de una resonancia

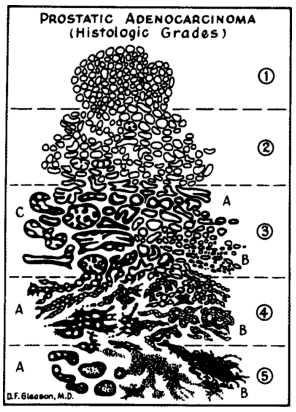
La biopsia es una técnica invasiva, ya que requiere extraer el tejido. Por ello, los médicos están muy interesados en poder utilizar técnicas no invasivas para el diagnostico del cáncer de próstata. Sin embargo, no está claro que patrones en la RM corresponde a cada nivel de severidad. Por ello, conseguir mediante IA acertar el diagnóstico obtenido mediante biopsias con los patrones presentes en la RM sería un gran paso a nivel de diagnóstico en este campo. Sin embargo, es muy difícil, ya que ni los propios urólogos son capaces de interpretar bien las RM.

1.3. Escala Gleason

Como se gradúa el cáncer es un aspecto importante del diagnostico. En esta sección se explica como funciona el sistema de graduación *Gleason*. Este sistema fue desarrollado por el patólogo Donald F Gleason entre 1960 y 1970 junto con el grupo de investigación urológica VACURG [7], y es el sistema más usado para determinar el grado cáncer de próstata.

En primer lugar se realiza una biopsia al paciente y se examina con microscopio una muestra del tejido extirpado. A continuación, el médico asigna un grado a las células cancerígenas dependiendo cuanto se parezcan a las células de próstata normales. En la figura 1.2a se muestra como el grado 1 significa que las células cancerígenas son muy parecidas a las normales, mientras que en el grado 5 son muy diferentes. Cuanto más alto sea este número, mas propenso a expandirse será el tumor [8].

Lo siguiente es reconocer cuales son los dos grupos de células más comunes y sumar sus grados, por ejemplo, si el grado más común en las células es el 3 seguido del 4, la puntuación Gleason seria de 3+4 = 7 [9]. Sin embargo, al haber una ambigüedad con la puntación 7, ya que puede venir tanto de 3+4 como de 4+3, y siendo este último más probable a que avance, se ha creado un nuevo sistema de 5 grados que si que resuelve este problema, figura 1.2b.



(a) Grados de Gleason

Risk Group*	Grade Group	Gleason Score
Low/Very Low	Grade Group 1	Gleason Score ≤ 6
Intermediate (Favorable/Unfavorable)	Grade Group 2	Gleason Score 7 (3 + 4)
	Grade Group 3	Gleason Score 7 (4 + 3)
High/Very High	Grade Group 4	Gleason Score 8
	Grade Group 5	Gleason Score 9-10

(b) Escala de Gleason

Figura 1.2: Ilustración de como funciona el sistema de graduación Gleason

Esta escala de 5 clases se conoce como *ISUP*, y será la que usaremos en la red para clasficar el grado de cáncer.

1.4. Objetivos y Estado del arte

El objetivo final de este trabajo es desarrollar un algoritmo, basado en el *Deep Learning*, que sirva de ayuda para el diagnostico de cáncer de próstata en imágenes de resonancias magnéticas.

Podemos dividir el trabajo en tres objetivos principales:

1. En primer lugar, con una primera base de datos, se entrenará a una red para que consiga extraer únicamente la próstata de toda la imagen de la resonancia.

2. En segundo lugar, y con ayuda de una segunda base de datos, se entrenará a una red adversarial con el fin de adaptar la red al dominio de la segunda base de datos y estudiar la capacidad de generalización en escenario de uso reales.

3. Por último, se usará una red entrenada en los apartados anteriores para segmentar la próstata de las imágenes de una tercera base de datos. Con estas imágenes segmentadas se entrenará a una red con el fin de que consiga diferenciar entre 5 o 2 grados de cáncer en escala *ISUP*.

En cuanto al estado del arte, los datos que usaremos para trabajar en este proyecto fueron liberados a raíz de un *Gran Desafío*, en concreto, PROSTATEx. Al terminar este reto, se publicó un artículo exponiendo los resultados obtenidos [10], donde podemos observar como, a la hora de graduar el cáncer en escala Gleason, los resultados no fueron muy prometedores, habiendo únicamente dos modelos que actuaron mejor que el azar.

Por otro lado, tenemos una revisión del 2021 sobre los algoritmos basados en IA para la clasificación del PCa con un total de 59 estudios [11]. En él se clasifica la eficacia de los estudios en una escala del 1 al 5, dependiendo de los métodos y las métricas usadas. Estando la mayoría de estudios en el nivel 2 concluyó que todavía falta evidencia del impacto que puede tener la IA en un escenario real, remarcando que el trabajo futuro debería ir en la linea de la generalización de los modelos.

Capítulo 2

Marco Teórico

Este capítulo pretende ser un resumen de los conocimientos necesarios de Deep Learning para entender como funcionan las redes que se han usado en este trabajo y que se explican en las siguientes secciones.

2.1. Perceptron Multicapa

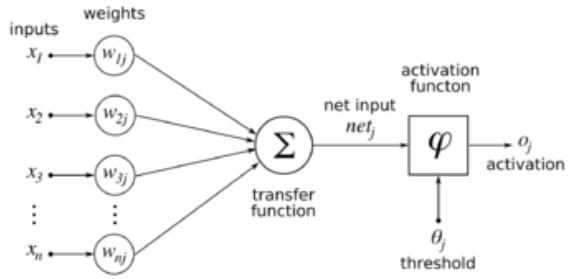


Figura 2.1: Perceptron de una capa

El *Perceptron* es la arquitectura de red más útilizada. Su potencial reside en el elemento base de todas la redes neuronales, *la neurona*. Una neurona realiza los siguientes pasos. Primero se hace una combinación lineal entre los pesos de entrada, w_{kj}, y la señal de entrada, x_j. El resultado de esta operación se pasa por una función de activación, que elimina la linealidad. Por último, a esta salida se le coloca un *threshold* para decidir si es de una clase o de otra. Este esquema se muestra en la figura 2.1, también se conoce como *Single-Layer Perceptron* y fue inventado por Frank Rosenblatt en 1958 [12].

Si juntamos varias neuronas en varias capas, uniendo la salida de cada neurona de una capa con la entrada de todas la neuronas de la capa siguiente, tenemos la arquitectura del Perceptron Multicapa, MLP, de la figura 2.2.

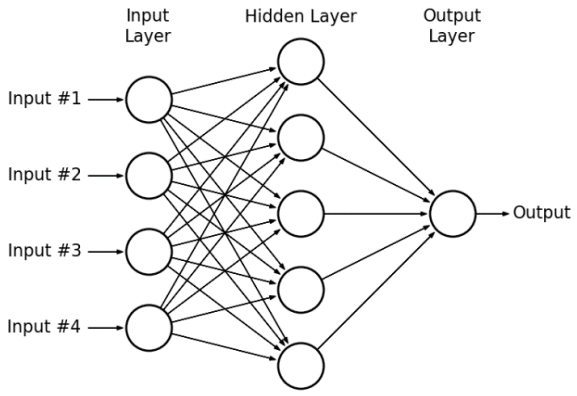


Figura 2.2: Perceptron multicapa

2.2. Algoritmos de aprendizaje

En esta sección se explica una parte vital de las redes neuronales, su proceso de *aprendizaje*. Este proceso de entrenamiento consiste en un algoritmo que actualiza los pesos de la red, w_{kj}, iterativamente, mejorando el desempeño de la red en cada iteración.

El entrenamiento puede ser de dos formas, supervisado o no supervisado. En el primero se tienen las etiquetas de los datos y la red se entrena comparando su salida con dichas etiquetas, para que vaya ajustando sus pesos poco a poco. Por el contrario, en el segundo no se tienen las etiquetas y los procesos de aprendizaje son muy variados.

El proceso de entrenamiento supervisado se divide en dos partes:

- **Forward propagation**: consiste en mostrarle un ejemplo a la red y pasar hacia adelante realizando el proceso descrito en 2.1, pero en todas las capas y neuronas. Este proceso genera un nuevo resultado que se compara con la etiqueta real mediante una función que nos cuantifica lo mucho o poco que se parezcan. Esta función se conoce como función de pérdidas, las más usadas son el error cuadrático medio (MSE) y la Binary Cross Entropy (BCE).

- **Back Propagation**: es un algoritmo que propaga el error producido en la salida de la red hacia atrás, lo que nos permite obtener los gradientes de todos los pesos, es decir, como afecta a la función de pérdidas una pequeña variación del peso w_{kj}. Por tanto, dando un pequeño paso en la dirección contraria al gradiente, conseguimos reducir las pérdidas y ajustarnos mejor a las etiquetas.

Sin embargo, existen unos parámetros que no se actualizan durante el entrenamiento, pero si que influyen en el resultado final de este, son los conocidos como *Hiper-parámetros*. Los más conocidos son:

- **Learning rate** (LR): sirve para controlar el tamaño del paso que damos en la dirección contraria al gradiente. Un LR alto, puede acabar divergiendo pero si converge lo hace más rápido. Mientras que un LR bajo, aunque necesite más iteraciones para converger, nos aseguramos de que encuentra el mínimo.

- **Bacth Size** (BS): es el número de ejemplos que le pasamos a la vez a la red. En lugar de enseñarle un ejemplo en cada iteración, le mostramos varios ejemplos a la vez en una nueva dimensión. Sirve para optimizar el tiempo de entrenamiento.

- **Épocas**: es el número de veces que le pasamos todos los datos a la red. Es decir, cuando le hayamos mostrados todos los datos se los volvemos a mostrar, así, tantas veces como épocas.

- **Profundidad de la arquitectura**: siguiendo con el MLP, estos parámetros serían el numero de capas o el número de neuronas en cada capa. Cuanto más profunda sea nuestra red, podrá solucionar problemas más complejos, pero si la hacemos demasiado profunda sufrirá de *Overfitting*, el cual se explica más adelante.

2.3. Redes Neuronales Convolucionales

Las redes neuronales convolucionales (CNNs) han supuesto un antes y un después en el campo de la visión por ordenador. El elemento más importante que caracteriza a las CNNs, es el uso de *Kernels*. Estos realizan una convolución a cada grupo de píxeles de la imagen de entrada con el fin de detectar patrones. Dicho grupo de píxeles esta definido por el tamaño del kernel.

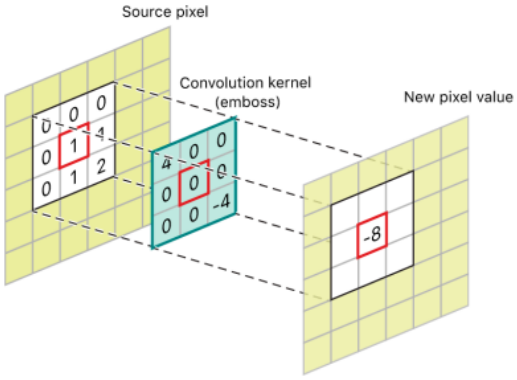


Figura 2.3: Filtros Kernel

Varios kernels juntos y conectados forman filtros, y es lo que se conoce como una capa convolucional. Para entender mejor el funcionamiento de estas redes, a continuación se describen las partes que la forman [13].

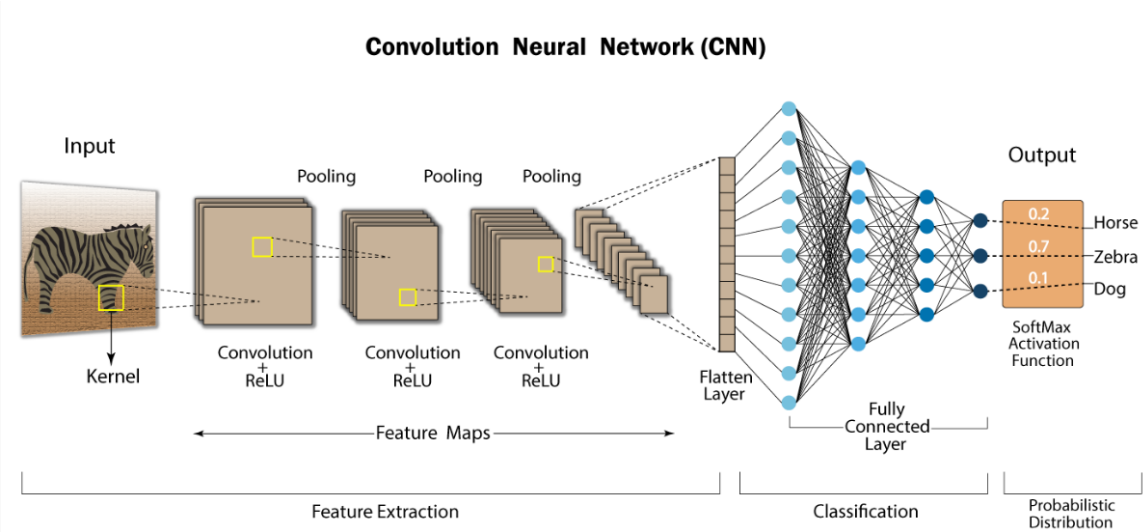


Figura 2.4: Arquitectura de una CNN

- **Convolutional layer**: esta capa aplica uno o varios filtros a la imagen de entrada. Mediante Kernels se va recorriendo la imagen por parches del tamaño del kernel. Si hay varios filtros, la salida del primero será la imagen de entrada del siguiente.

  Al final de cada capa convolucional se sitúa la función de activación. Las más comunes son la función Sigmoide y la Rectified Linear Unit (ReLU).

- **Pooling Layer**: esta capa realiza un *downampling* a la imagen, es decir, reduce las dimensiones de alto y ancho. Aunque esto conlleva un pérdida de información, reduce el coste computacional acelerando mucho el entrenamiento. Hay dos maneras de realizar un downsampling, *Max Pooling* o *Average Pooling*. La primera coge el valor máximo de un parche de píxeles, mientras que la segunda toma la media de los valores de los píxeles de dicho parche.

- **Fully Connected Layer**: antes de realizar la clasificación estiramos los valores de la última capa y usamos ese vector como entrada de un MLP.

Capítulo 3

Estructura y Metodología

Como se ha explicado en el apartado 1.4 , el trabajo se estructura en tres fases. Aunque se profundizará en cada una de ellas más adelante, aquí se explica brevemente en que consisten.

La primera fase trata sobre *segmentación de imágenes*. Este es un campo muy extenso dentro del Deep Learning, el cual consiste en dividir la imagen de entrada en conjuntos de píxeles, o regiones, de tal manera que ese conjunto de píxeles concuerde con lo que deseamos extraer de la imagen original. Se realiza una clasificación a nivel de píxel, es decir, si este píxel forma parte de lo que se quiere segmentar, se activará. Para lograr esto, se usará una arquitectura de red conocida como *UNET*, explicada más adelante.

En la segunda fase se desarrolla la *adaptación de dominio* o *Domain Adaptation* (DA). La motivación de esta fase reside en intentar subsanar un problema presente en todos los proyectos de Machine Learning, el *Overfitting*. Todas la redes se entrenan para ajustarse a los datos de entrenamiento, pero es labor de investigador vigilar que la red sea capaz de generalizar a los datos que no ha visto, o si por el contrario, se ajusta únicamente a los datos con los que se ha entrenado. Al entrenar la red con una sola base de datos (BBDD), es de esperar que su desempeño baje notablemente al probar imágenes de otro dominio. Es por esto, por lo que modificará la arquitectura de la red de la primera fase, para que, mediante unas pocas imágenes de una segunda BBDD, mejore sus resultados en este segundo dominio. A esta modificación de la arquitectura se le conoce como *entrenamiento adversarial*, el cual se explica más adelante.

En la tercera y última fase se realiza la *clasificación de imágenes*. Basicamente, esta tarea consiste en clasificar el grado de cáncer en una escala de 5 clases, *escala ISUP*. Se usará una *red convolucional 3D (CNN3D)*, y los datos de entrada serán las imágenes segmentadas y recortadas de la salida de la mejor red de las anteriores fases. Debido a que una CNN3D presenta problemas de memoria, las imágenes se deben segmentar y recortar para disminuir el tamaño de estas a únicamente la próstata.

3.1. Bases de Datos

Una de las etapas fundamentales de todo proyecto de Deep Learning es el tratamiento de los datos, ya que, y sobre todo en imágenes médicas, estos suelen ser escasos. Además, como hemos visto, las redes se entrenan mediante ejemplos, ajustándo sus parámetros únicamente a estos, por lo que dentro de este algoritmo de aprendizaje no hay nada que nos asegure que la red vaya a actuar de la misma forma con ejemplos que no ha visto.

Una estrategia que nos ayuda a vigilar el comportamiento de la red es la de dividir los datos en 2 o 3 grupos. Con uno de ellos entrenaremos a la red, mientras que con los otros controlaremos si está a aprendiendo a generalizar o si, por el contrario, se ajusta únicamente a los datos de entrenamiento.

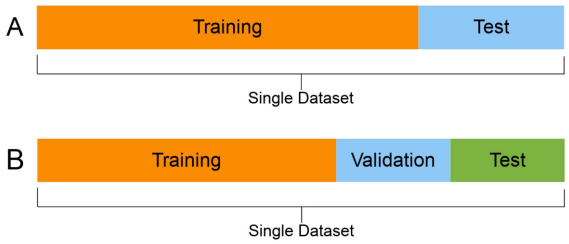
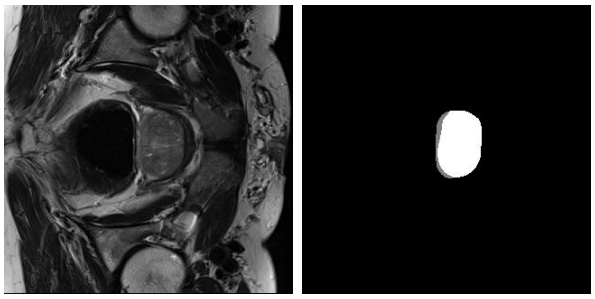


Figura 3.1: Ejemplo de partición

Como ya se ha mencionado anteriormente, se dispone de 3 bases de datos, *MSD*, *PROSTATEx* y *PI-CAI*, todas ellas han sido liberadas recientemente.

- **MSD**: fue liberada a partir de un reto de segmentación de imágenes médicas llamado *The Medical Segmentation Decathlon* [14], de ahí las siglas. Consta de un total de 32 volúmenes para el entrenamiento. Como se explicará más adelante, deberemos realizar unos cortes transversales a los volúmenes. Al hacer los cortes, obtenemos un total de 602 imágenes para el entrenamiento, de las cuales usaremos un 20 % para validación. En figura 3.2 podemos ver un ejemplo de un corte y su máscara. Esta BBDD será utilizada en la primera fase para realizar la segmentación.

- **PROSTATEx** [15]: tambien proviene de un reto y fue liberada por *The Cancer Imaging Archive*. Consta de 204 volúmenes, que al hacerle los cortes transversales, obtenemos 4167 imágenes. Esta base de datos la usaremos para la adaptación de dominio, por lo que no será necesario realizar una partición, basta con tomar unas pocas imágenes.



(a) Imagen (b) Máscara

Figura 3.2: Corte transversal de un volumen de próstata y su máscara

- **PI-CAI**[16]: es una extensión de PROSTATEx ya que repite algunos casos de esta. Las características del dataset se muestran en la tabla 3.1.

Nº pacientes	1476
Nº.casos	1500
Casos benignos	1075
Casos malignos ($ISUP \geq 2$)	425
Edad media	67
Nº de lesiones ISUP	775
ISUP 1	310
ISUP 2	260
ISUP 3	109
ISUP 4	41
ISUP 5	55

Cuadro 3.1: Descripción original del dataset PI-CAI

Sin embargo, al realizar el proceso de segmentación y recorte, el dataset con el que entrenaremos se reduce ligeramente. Usaremos el dataset del cuadro 3.2 para entrenar a una red que diferencie entre 5 clases. Y usaremos el dataset del cuadro 3.3 para entrenar a una red que diferencie entre 2 clases.

La partición la haremos del siguiente modo: 70 % para entrenamiento, 20 % para test y 10 % para validación, procurando que la distribución de las clases objetivo se mantenga constante en todas las particiones.

Nº casos	1463
ISUP 0 o 1	1048
ISUP 2	226
ISUP 3	98
ISUP 4	39
ISUP 5	52

Cuadro 3.2: Descripción del dataset PI-CAI para 5 clases

Nº casos	1463
ISUP 0 o 1	1048
$ISUP \geq 2$	415

Cuadro 3.3: Descripción del dataset PI-CAI para 2 clases

3.2. Trabajo previo

Antes de comenzar a utilizar las BBDD y empezar a entrenar debemos realizar algunas tareas previas con el fin de facilitar y acelerar el entrenamiento. Estas tareas son: la conversión de imágenes 3D a 2D y la creación de los DataSets y DataGenerators.

Tanto en la primera como en la segunda fase, para evitar problemas de memoria, las imágenes de entrada a la red serán en 2D, sin embargo, las imágenes de resonancias magnéticas son en 3D. Por ello, se crearan dos nuevas bases de datos, *MSD_2D* y *PROSTATEx_2D*, con los cortes transversales de las imágenes de las respectivas BBDD.

Una vez se tienen las nuevas imágenes, el siguiente paso es la crear los correspondientes *DataSets* y *DataGenerators*, los cuales nos ayudarán a acelerar el entrenamiento. Por un lado, la función de un *dataset* es la de leer las imágenes y preprocesarlas, convirtiéndolas en un archivo más óptimo para la red. Por otro lado, la función de un *datagenerator* es la de controlar el número de imágenes del dataset que se van a pasar simultáneamente a la red.

3.3. Segmentación

Como ya se ha mencionando al principio de este capítulo, la primera fase trata sobre la *segmentación de imágenes*, es decir, extraer o dividir una imagen en regiones de píxeles. Esta técnica tiene numerosas aplicaciones en muchas áreas de la ingeniería. Por ejemplo, como podemos observar en la figura 3.3, en la conducción autónoma se usan algoritmos de segmentación para identificar en que parte de la imagen se encuentra la carretera, o donde estarían los peatones, etc. Otro ejemplo práctico serían las imágenes por satélite, donde se intenta identificar que parte de la imagen es un bosque, un lago o una casa. Sin embargo, la aplicación que nos interesa aquí sería la de identificar en que parte de la imagen se encuentra la próstata.

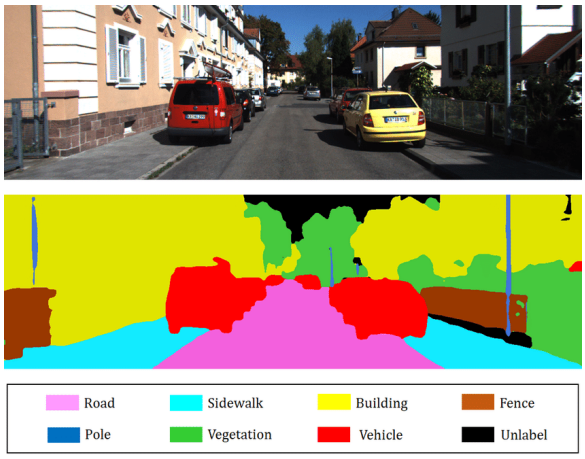


Figura 3.3: Ejemplo de segmentación para conducción autónoma

3.3.1. AutoEncoders

Como lo que se desea es crear una nueva imagen, debemos hacer uso de redes generativas y, en concreto, de los conocidos *AutoEncoders*. Diederik P. Kingma y Max Welling publicaron en 2013 un artículo donde se describían las bases de estas nuevas redes neuronales [17]. Para poder entender cómo funcionan, primero debemos entender las partes que la forman.

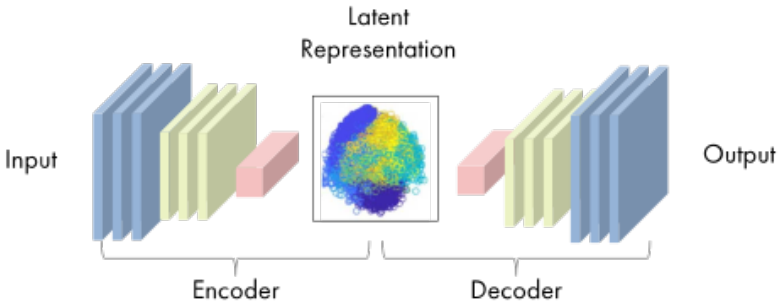


Figura 3.4: AutoEncoder

Un AutoEncoder se divide en dos partes, el *Encoder* y el *Decoder*. El primero se encarga de tomar la imagen de entrada, y reducirla a una matriz de características conocida como *espacio latente*. Por su parte, el Decoder, toma de entrada ese espacio latente, y genera una imagen a partir de él.

Una analogía de este proceso sería lo que ocurre cuando nos intentan describir a una persona. Digamos que una persona, A, intenta describirnos a nosotros, B, como es otra persona, C. Aquí, el encoder sería A, reduciendo a C, la imagen de entrada, a una matriz de características (pelo rubio, alto, ojos azules, etc). Y el decoder seriamos nosotros, generando una imagen mental a partir de las características.

3.3.2. UNET

Sin embargo, el fin de los AutoEncoders no era la segmentación, sino el de generar nuevas imágenes que no existieran, pero que se parecieran al conjunto de imágenes originales. Es por esto que se modifica la arquitectura anterior para crear un nuevo tipo de red generativa cuyo fin sea la segmentación. A esta nueva red se le conoce por el nombre de *UNET*, y su arquitectura se muestra en la figura 3.5.

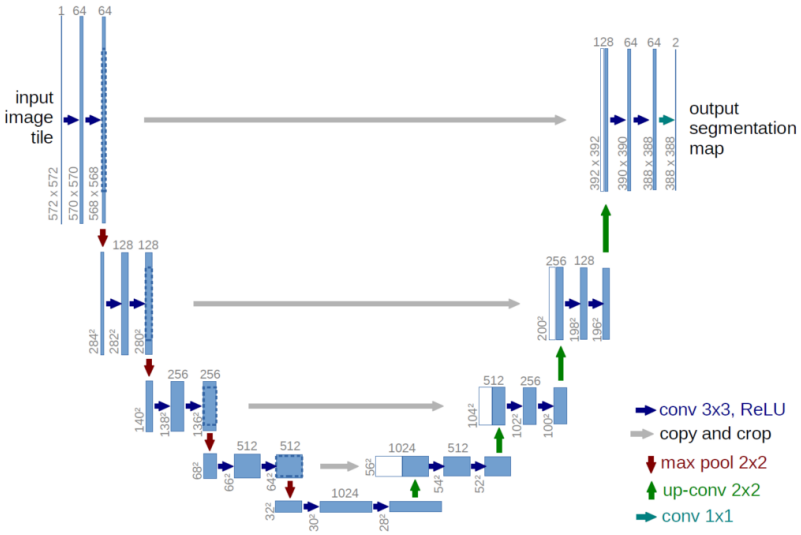


Figura 3.5: UNET

Esta arquitectura fue publicada en 2015 por Ronneberger [18], y el fin era la segmentación de imágenes médicas. La diferencia fundamental de esta arquitectura con la del AutoEncoder reside en las *Skips Connections*. Lo que hacen estas conexiones es pasar información posicional del Encoder al Decoder, por lo que, a la hora de generar la nueva imagen, el Decoder tiene más información sobre la posición del contenido de la imagen original, lo cual es muy útil si el objetivo es conocer la posición que ocupa el objeto de interés.

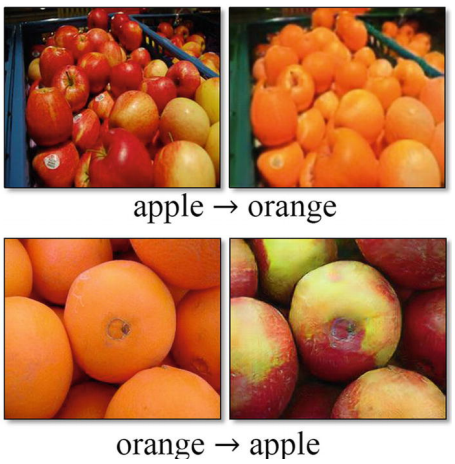
Estas redes no se entrenan con una etiqueta como tal, sino con una máscara que muestra donde está realmente lo que se desea segmentar. Esta máscara se compara con la salida de la red y se evalúa el error que se está cometiendo mediante una función de pérdidas, en concreto, usaremos *DICE* como función a optimizar, la cual se explica más detalladamente en la sección 4.1.

3.4. Domain Adaptation

En esta sección se explica en que consiste la *adaptación de dominio* y que redes se usan para llevarla acabo. En primer lugar se explicará qué es la *transferencia de estilo* y, en segundo lugar, cómo se adapta esta técnica a nuestro problema original, que era mejorar el desempeño de las redes anteriores en otra BBDD. En ambos casos se usan el mismo tipo de redes, las conocidas como *Generative Adversarial Network*, por su acrónimo, GAN.

3.4.1. Transferencia de estilo

Originalmente esta técnica consistía en, dada una imagen original, generar un nueva imagen que tuviera el mismo contenido pero con un estilo diferente [19]. Como vemos en el ejemplo de la figura 3.6a, se pretende generar una imagen que mantenga el contenido, es decir, que no pierda la posición de las manzanas, pero que sí cambie el estilo de estas para que se parezcan a unas naranjas. Otro ejemplo práctico sería el de poder cambiar el estilo de una foto para que parezca que ha sido pintada por Monet o Van Gogh, como vemos en en ejemplo de la figura 3.6b.



(a) Ejemplo 1 Transferencia de estilo

(b) Ejemplo 2 Transferencia de estilo

Figura 3.6: Ejemplos sobre como funciona la transferencia de estilo

3.4.2. Redes Generativas Adversarias

Como se ha mencionado al principio, las redes que usaremos para realizar esta técnica se conocen como GANs, las cuales fueron presentadas en 2016 en una conferencia en Barcelona por Ian Goodfellow investigador de Google Brain [20]. En la figura 3.7 podemos ver la arquitectura de una GAN, la cual se compone de dos redes. Por un lado, tenemos al *generador*, que puede ser una UNET o cualquier otra red generativa, y por otro lado tenemos al *discriminador*, que realiza una tarea de clasificación.

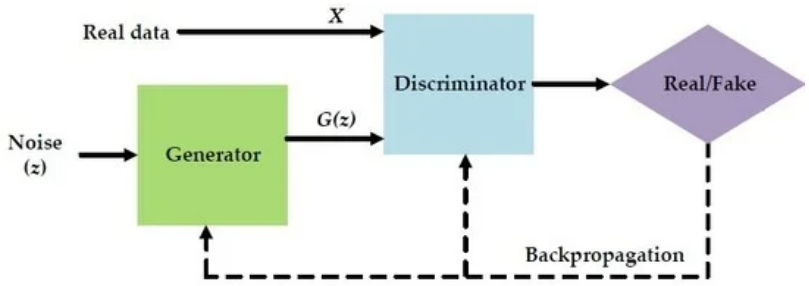


Figura 3.7: Arquitectura de una GAN

Siguiendo con el ejemplo de las naranjas y las manzanas, el generador se encarga de convertir una imagen de naranjas en una de manzanas, mientras que la labor del discriminador es la de diferenciar si lo que se ha generado son manzanas o naranjas.

Lo realmente interesante es como se entrenan estas redes. Para entender esto vamos a suponer que el discriminador ya ha sido entrenado y es capaz de diferenciar muy bien cuando una imagen es de naranjas o manzanas, entonces, el generador se entrenará de tal forma que sea capaz de burlar al discriminador, es decir, se premiará al generador cuando, a partir de una imagen de manzanas, genere una imagen que haga creer al discriminador que son naranjas. En la práctica, al principio ambas redes están sin entrenar y habrá que entrenarlas por separado.

Sin embargo, y aunque la técnica que se usa es la misma, esto no es exactamente lo que buscamos para nuestro problema. En nuestro caso, lo que se desea hacer es que el generador mejore la segmentación de imágenes de otra BBDD mediante unas pocas imágenes de esta. Para ello, como se observa en la figura 3.8, se modifica la arquitectura ligeramente.

Para adaptar la arquitectura a nuestro problema, hemos colocado el discriminador a la salida del espacio latente, la matriz de características mencionada en la sección 3.3. De esta forma, el discriminador aprendará a diferenciar si las características son de MSD o de PROSTATEx, mientras que el encoder deberá aprender a generar unas características que consigan confundir al discriminador, es decir, que cuando entre una imagen de PROSTATEx, el encoder deberá reducirla a un espacio latente que haga decidir al discriminador que la imagen que entró fue de MSD. Así, conseguimos reducir ambas BBDD a un espacio latente que el decoder comprenderá mejor a la hora de generar la máscara.

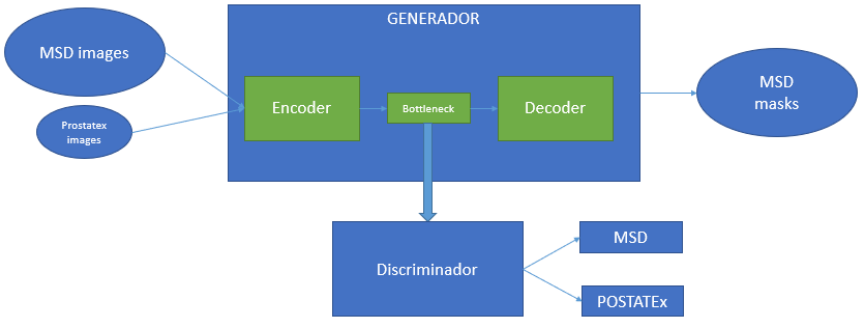


Figura 3.8: Arquitectura para adaptación de domio

Esta adaptación de domino nos será útil para la tercera y última fase, la *Clasificación*, donde usaraemos la tercera BBDD, PI-CAI, la cual contiene algunas imágenes del dominio de PROSTATEx. Véase que no se ha entrenado directamente la segmentación con PROSTATEx porque en la siguiente fase segmentaremos las imágenes de PI-CAI, y no sería justo que sean las mismas imágenes que con las que se ha entrenado la segmentación. En su lugar, se ha querido mostrar que unas pocas imágenes bastan para adaptar la red de un dominio a otro.

3.5. Clasificación

Esta última fase consiste en la *clasificación* de las imágenes de PI-CAI en 5 o 2 grados de cáncer. La red que usaremos para esto sera una convolucional, explicada en la sección 2.3, pero en 3 dimensiones, CNN3D.

Como se ha mencionado en la sección 3.2, la imágnes de las BBDD son en 3D, pero las redes han sido entrenadas con los cortes en 2D. Sin embargo, aquí, si que tomaremos las imágenes en 3D, pero para reducir su tamaño usaremos los modelos anteriores para recortar únicamente la próstata. Esto lo haremos siguiendo estos pasos:

1. Tomar una imagen y dividirla en cortes transversales.
2. Pasar los cortes por el mejor modelo de entre las fases 1 y 2.
3. Realizar morfología matemática, en concreto, una apertura, a cada uno de los cortes. Esto se hace son el fin de suavizar el resultado.
4. Reconstruir el volumen con todos los cortes.
5. Este volumen reconstruido puede tener pequeñas regiones de píxeles fuera de la próstata que no deberían estar. Por tanto, mediante un algoritmo, nos quedaremos con la región de píxeles que tenga el volumen más grande.

6. Con el volumen ya recortado, nos quedaremos con el volumen de la imagen que coincida con la máscara.
7. Guardaremos el volumen anterior y repetiremos el proceso con todas las imágenes.

El objetivo de los anteriores pasos es el de reducir el tamaño de entrada de la red, ya que, al ser redes en 3D, tienen un elevado coste computacional, que se aumenta al cubo cuanto más grandes sean lás imágenes.

Con esta nueva BBDD entrenaremos a una CNN3D como la que se presenta en la figura 3.9. El objetivo inicial es diferenciar entre 5 grados de cáncer, pero más adelante se intenta únicamente con 2 clases.

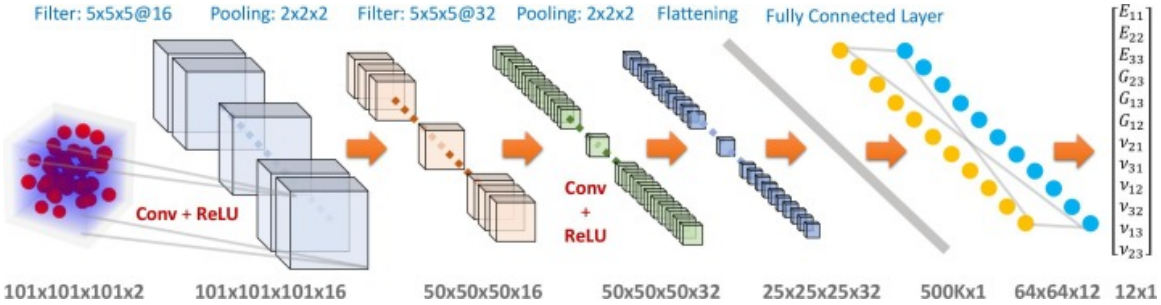


Figura 3.9: Arquitectura de una CNN3D [21]

Se trabaja de manera general para optimizar la función de pérdidas *Categorical Cross-Entropy* (CE), la cual es una manera de medir la diferencia en la cantidad de información entre dos variables aleatorias. Su expresión matmática está definida del siguiente modo:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * log(p_{ij}) \tag{3.1}$$

3.5.1. Morfología

Como se ha mencionando anteriormente, en el tercer paso usaremos morfología matemática para suavizar la máscara generada por la UNET. En esta sección se explica en consiste esta técnica y que operaciones se realizan.

La *morfología matemática* surge en 1964 de la mano de Jean Serra [22], se basa en la topología y en la teoría de conjuntos y se usa en el procesamiento de imágenes para manipular objetos. Se usa una estructura para transformar cada píxel dependiendo del valor de sus vecinos. Por ejemplo, tomemos como nuestra imagen original la que se muestra en la figura 3.10, esta imagen nos servirá para explicar en que consisten las 4 operaciones básicas de la morfología: erosión, dilatación, apertura y cierre.

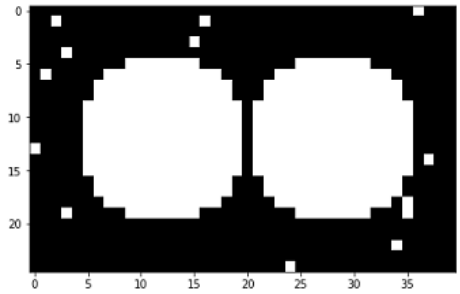


Figura 3.10: Imagen original de ejemplo

- **Erosión**: reduce el tamaño de los objetos y ayuda a eliminar islas de manera que solo el el objeto más grande prevalece. Solo se eliminan los objetos que sean más pequeños que la estructura.
- **Dilatación**: lo contrario que la erosión, agranda los objetos y ayuda a rellenar pequeños agujeros.

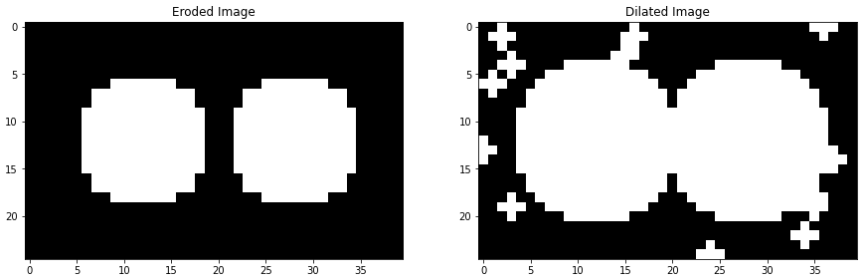


Figura 3.11: Erosión y Dilatación aplicada a la imagen original

- **Apertura**: consiste en realizar sucesivamente una erosión primero y una dilatación después.
- **Cierre**: lo contrario que la apertura, primero se realiza la dilatación y luego la erosión.

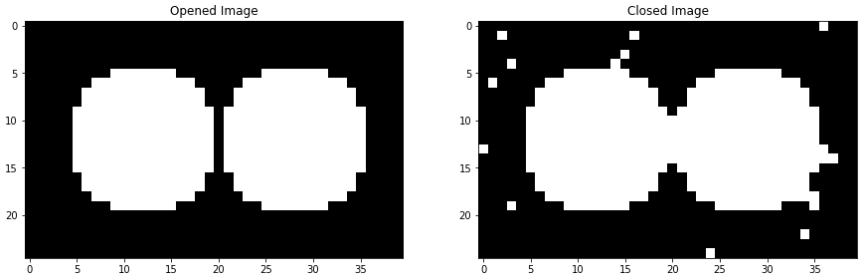


Figura 3.12: Apertura y cierre aplicado a la imagen original

Visto esto, una vez la UNET nos haya generado la máscara, le aplicaremos una apertura para eliminar pequeñas islas, rellenar huecos, y suavizar la imagen. Además, nos servirá para limpiar el ruido producido en la predicción de la red, introduciéndole restricciones de tamaño y forma sabiendo que, en nuestro problema, la próstata tiene una forma de disco y un tamaño estándar.

Una vez ya realizada la morfología y hayamos reconstruido el volumen, es posible que todavía sigan quedando regiones de píxeles en 3D que la apertura no haya conseguido eliminar. Por tanto, lo que haremos será, mediante un algoritmo, detectar cuantas regiones de píxeles adyacentes tenemos en nuestra imagen, y que volumen tiene cada una para quedarnos únicamente con la región del volumen más grande.

Capítulo 4

Experimentos y resultados

En esta sección se exponen resultados obtenidos para cada una de las fases. Con el fin de contextualizar mejor los resultados, comenzaremos haciendo un breve resumen de los objetivos principales del proyecto.

Así, en la primera fase se utiliza una UNET para segmentar únicamente la próstata de toda la resonancia. En la segunda, se hace uso de unas pocas imágenes de PROSTATEx para adaptar la red a este dominio mediante el entrenamiento adversarial. Por último, en la tercera, usamos las imágenes de PI-CAI, segmentadas por las anteriores redes y procesadas para entrenar una CNN3D con el fin de que consiga diferenciar entre 5 o 2 grados de cáncer.

Visto esto, en la primera sección se explican las métricas usadas para evaluar y comparar los resultados, mientras que en la segunda sección se exponen y se describen los mejores resultados obtenidos de cada fase.

4.1. Métricas usadas

Con el fin de evaluar el rendimiento de los algoritmos desarrollados, tenemos a nuestra disposición diferentes métricas que nos ayudan en nuestra tarea. Escoger la métrica correcta es importante ya que cada una tiene sus ventajas y desventajas, las cuales debemos comprender si queremos ser rigurosos y críticos con los resultados.

Antes de comenzar con las métricas, vamos a explicar una herramienta que nos sirve para visualizar el comportamiento de nuestra red y nos ayuda a evaluar los resultados, la *Matriz de Confusión*.

En la figura 4.1 se muestra el caso binario de la matriz de confusión, en nuestro caso será una matriz de 5x5 ya que tenemos 5 clases. Al comparar los valores reales con los que predice la red, pueden ocurrir 4 casos:

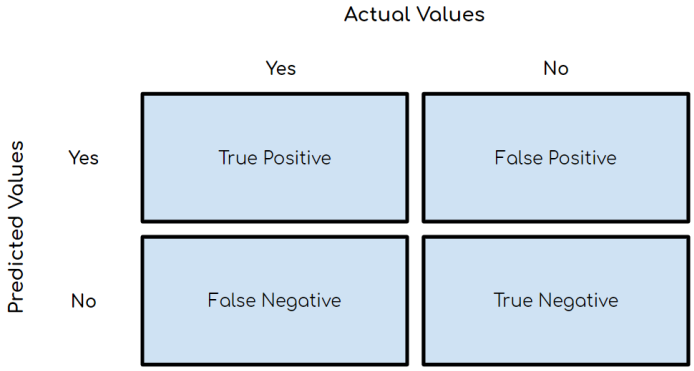


Figura 4.1: Matriz de Confusión

- True Positive (TP): cuando el valor real es positivo y la red lo predice positivo.
- True Negative (TN): cuando el valor real es negativo y la red lo predice negativo.
- False Positive (FP): cuando el valor real es negativo y la red lo predice positivo.
- False Negative (FN): cuando el valor real es positivo y la red lo predice como negativo.

Esta matriz cuenta el número de veces que se produce cada caso, por tanto, cuantos más elementos tengamos en la diagonal, mejor será la precisión de nuestra red. Visto esto, las métricas usadas son las siguientes:

- **Accuracy**(Acc): es la métrica mas básica y mide el ratio entre el número de aciertos y número total de casos. Siguiendo con las definiciones explicadas anteriormente, la *Accuracy* se define del siguiente modo:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{4.1}$$

  Sin embargo, con datasets desbalanceados como el nuestro, es decir, que hay una prevalencia por algunas clases más que otras, está métrica no proporciona información muy útil. Imaginemos que tenemos un dataset con el 95 % de los casos positivos y el 5 % negativos. Si nuestro modelo siempre predice positivo, su accuracy sería del 95 %. Es por esto que se usan otras métricas como F1 score o Cohen Kappa que si tienen en cuenta este desbalanceo.

- **Dice**: será usada como métrica y como función de pérdidas para las dos primeras fases. Sirve para comparar cuanto se parece la máscara real con la generada por la red del siguiente modo:

$$Dice = \frac{2\,|A \cap B|}{|A| + |B|} \tag{4.2}$$

Donde A serían los píxeles activos de la máscara, B los píxeles activos de la predicción y, $A \cap B$, los píxeles que comparten A y B. En la siguiente imagen se comprende mejor esta expresión.

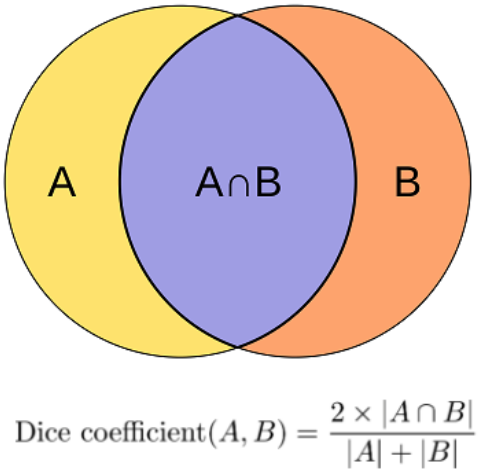


Figura 4.2: Métrica Dice

- **F1 score**: está métrica se define a partir de otros dos conceptos, *Precision y Recall*. La *Precision* es la relación en los positivos acertados (TP) y todos los positivos predichos. Mientras que *Recall* mide la relación entre los positivos acertados y todos los positivos.

$$Precision = \frac{TP}{TP + FP} \tag{4.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.4}$$

$$F1score = 2\frac{Precision \cdot Recall}{Precision + Recall} \tag{4.5}$$

- **Cohen Kappa**: al igual que el Dice, es usada tanto función de pérdidas como métrica. Se usa para problemas de clasificación y mide cuanto se parece la predicción al valor real, pero se diferencia del accuracy al tener en cuenta el acierto que puede haber por suerte. Es decir, si prevalece una clase sobre otra es más probable que, por suerte, se acierte más en esa clase, por tanto, se resta el acierto por suerte al total de aciertos.

Sin embargo, aquí no se tiene en cuenta lo difícil que resulta, incluso a los expertos, diferenciar el grado de cáncer en imágenes médicas, como biopsias o resonancias. Es decir, una misma imagen puede diferir en 1 o 2 grados de cáncer dependiendo del patólogo que evalúe la imagen. Para solucionar esto se define *Quadratic Weighted Kappa* (QWK) que minimiza el error que se produce en clases adyacentes y lo máximiza cuanto más distantes estén. Se expresa del siguiente modo:

$$\kappa = 1 - \frac{\sum_{ij} W_{ij} O_{ij}}{\sum_{ij} W_{ij} E_{ij}} \tag{4.6}$$

Donde, W es la matriz de pesos, O es la matriz de confusión y E es el producto vectorial entre el número de veces que un resultado ha ocurrido y el número de veces previsto para ese resultado.

La matriz de pesos W_{ij}, calcula la penalización por haber confundido la clase i con la j, y tiene la siguiente expresión, donde podemos observar que cuanto mayor sea la diferencia entre i y j, mayor será la penalización, siendo c es el número de clases:

$$W_{ij} = \frac{(i-j)^2}{(c-1)^2} \tag{4.7}$$

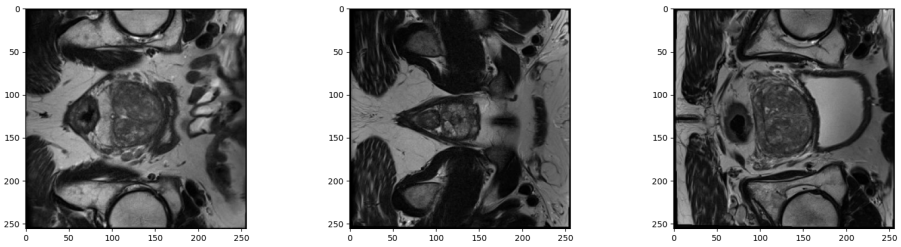
Sin embargo, una desventaja de usar este coeficiente es que no es tan fácil saber que considera como un resultado bueno o malo. Esta claro que el valor 1 significa que funciona a la perfección, y que el valor 0 o negativo es todo lo contrario. En el resto de casos, se suelen aceptar los valores comprendidos entre 0.3 y 0.4 como que el desempeño es mejor que el azar.

4.2. Discusión y exposición de resultados

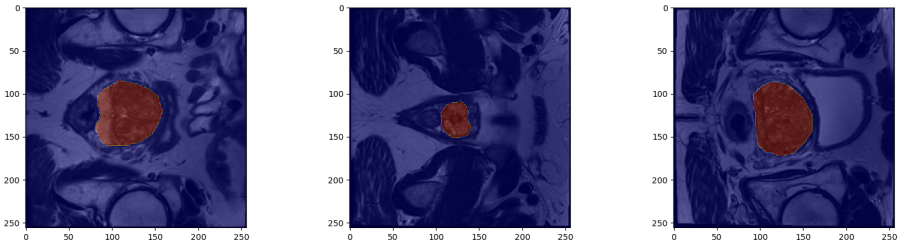
Aquí se muestran los mejores modelos de cada fase. Estos modelos son el resultado de un proceso de prueba y error para optimizar los hiper-parámetros de las redes, siendos los modelos muy sensibles a estos parámetros. Además, en la última fase, para cada configuración de hiper-parámetros se ha probado con diferentes funciones de pérdidas, aunque aquí solo se exponen las pruebas que han obtenido los mejores resultados.

4.2.1. Segmentación

En esta fase se han usado las 602 imágenes de MSD_2D para entrenar, de las cuales un 20 % son para validación. Para test se han usado las 4167 imágenes de PROSTATE_2D, y los mejores modelos son aquellos que mejor DICE hayan sacado para estas imágenes de test. Las imágenes que se muestran en la figura 4.4, las usaremos como referencia para mostrar algunos ejemplos de las máscaras que generan los modelos.



(a) Imagen 1 (b) Imagen 2 (c) Imagen 3



(a) Máscara 1 (b) Máscara 2 (c) Máscara 3

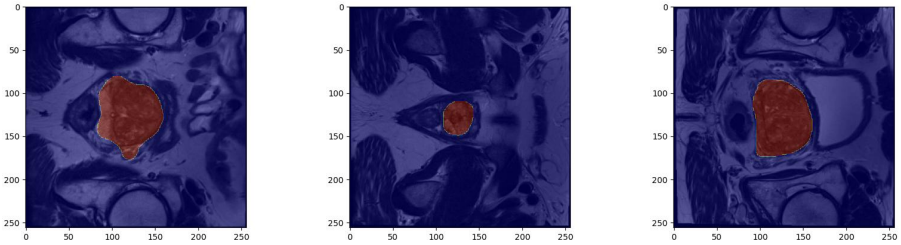
Figura 4.4: Imágenes de referencia para test

En la siguiente tabla se muestra el DICE alcanzado por los tres mejores modelos entrenados para la segmentación.

Modelo	LR	BS	Épocas	Entrenamiento	Validación	Test
1	5e-5	8	100	0.9419	0.8435	0.7932
2	1e-4	32	100	0.8888	0.8111	0.7775
3	5e-5	8	100	0.9409	0.8408	0.7331

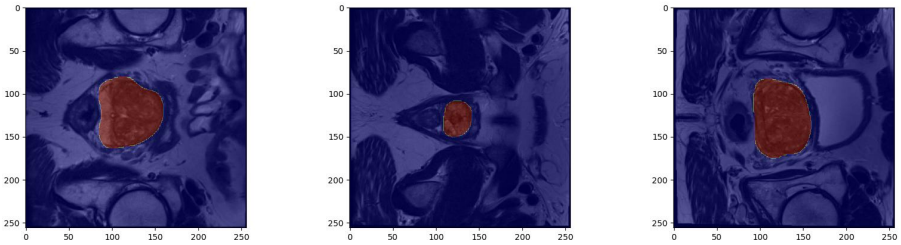
Cuadro 4.1: DICE de los mejores modelos de la primera fase

A continuación se muestran las predicciones de estos modelos para los ejemplos de la figura 4.4. Podemos observar que con un *DICE* de 0.8, aproximadamente, los modelos segmentan la próstata correctamente, excepto por algunas pequeñas discrepancias.



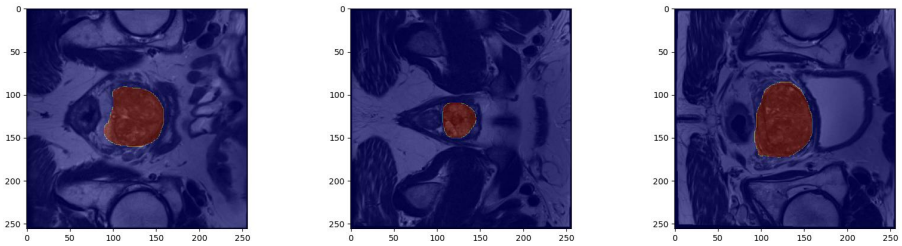
(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 4.5: Predicciones del modelo 1



(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 4.6: Predicciones del modelo 2



(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 4.7: Predicciones del modelo 3

4.2.2. Segmentación + DA

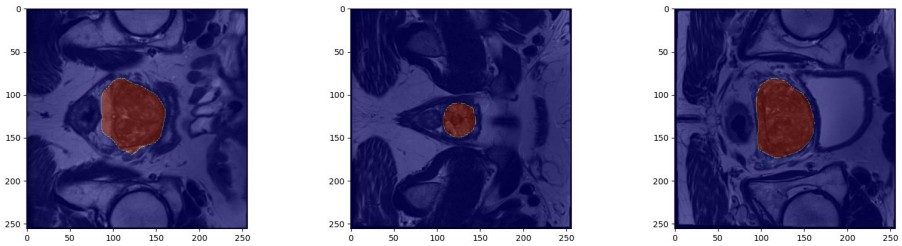
En esta fase se vuelven a usar las 602 imágenes de MSD_2D con el 20 % de validación y todas las de PROSTATEx como test, pero también se usan unas pocas imágenes de PROSTATEx_2D para entrenar. Usaremos las mismas imágenes de la figura 4.4 como referencia. En la siguiente tabla se muestran los resultados alcanzados por los tres mejores momentos, donde n^o *img_target* es el número de imagenes de PROSTATEx usadas.

Modelo	LR	BS	Épocas	nº img_target	Entrenamiento	Validación	Test
1	3e-5	8	50	8	0.8474	0.8378	0.8095
2	5e-5	8	100	16	0.9349	0.8539	0.8049
3	5e-5	8	100	8	0.9540	0.8321	0.8042

Cuadro 4.2: DICE de los mejores modelos de la segunda fase

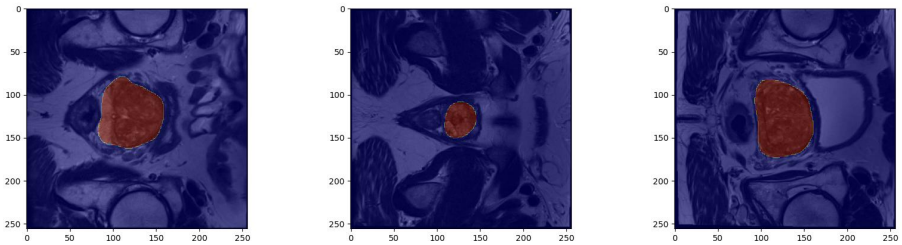
Con estos resultados conseguimos aumentar en un 2 % el desempeño de las redes anteriores en el dominio de PROSTATEx (véase Tabla 4.1). Aunque es justo preguntarse si este ligero aumento se debe a que ha aprendido a generalizar mejor las imágenes de la segunda base de datos, o si, por el contrario, se debe a que las pocas imágenes que ha visto para el entrenamiento se han repetido para test. Aun así, son pocas imágenes en comparación con el dataset total, y su entrenamiento ha sido no supervisado, por lo que no se tenían las máscaras de estas imágenes.

En las figuras 4.8, 4.9 y 4.10, se muestran las predicciones de los tres modelos anteriores para las imágenes de muestra. Al igual que en la fase anterior, un *DICE* de 0.8 es un buen resultado.



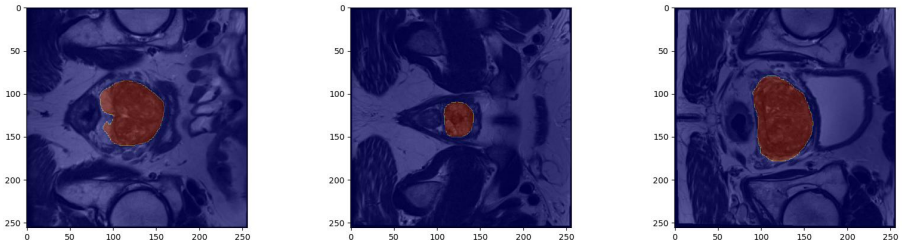
(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 4.8: Predicciones del modelo 1



(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 4.9: Predicciones del modelo 2

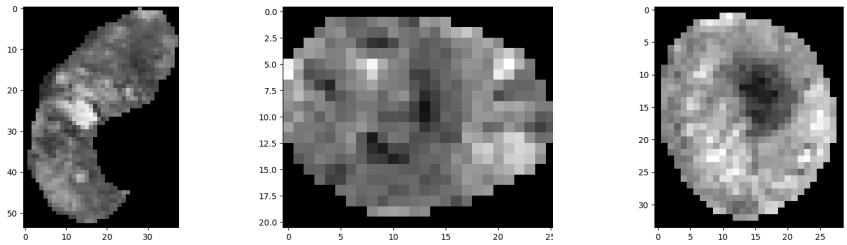


(a) Predicción 1 (b) Predicción 2 (c) Predicción 3

Figura 4.10: Predicciones del modelo 3

4.2.3. Segmentación + DA + Clasificación

Esta fase se divide en dos partes, en la primera se intenta diferenciar entre 5 clases mientras que en la segunda se realiza una clasificación binaria para detectar la presencia de patrones cancerosos. Esta fase es realmente complicada debido a la poca información que tienen las imágenes de PI-CAI tras haber sido segmentadas y recortadas. Para hacernos una idea de con que estamos trabajando, en la imagen 4.11 se muestran unos cortes de algunos volúmenes tras haberlos procesado. Como se ha explicado anteriormente, para esta fase se usa, de los 1463 volúmenes, un 70 % para entrenamiento, un 20 % para test y un 10 % para validación.



(a) Imagen 1 (b) Imagen 2 (c) Imagen 3

Figura 4.11: Cortes de muestra de los volúmenes segmentados de PI-CAI

Las métricas de los mejores modelos para la primera fase se muestran en el cuadro 4.3, ordenados por el valor del coeficiente kappa que hayan obtenido en test. Para la segunda parte, los mejores modelos son aquellos que han obtenido el F1 score más alto. Los tres mejores resultados se recogen en el cuadro 4.4.

Modelo	Train Acc	Val Acc	Test Acc	Val kappa	Test Kappa
1	0.4342	0.6224	0.6314	0.1731	0.2298
2	0.4131	0.5315	0.5529	0.1483	0.2018
3	0.7126	0.4825	0.4881	0.3381	0.1822

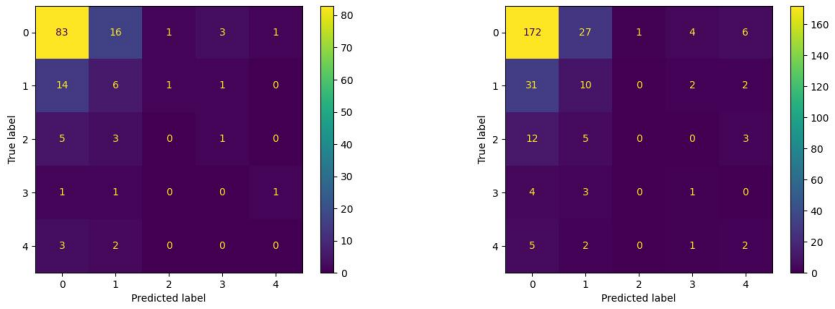
Cuadro 4.3: Métricas de los tres mejores modelos de la primera fase

Modelo	Train Acc	Val Acc	Test Acc	Val kappa	Test Kappa	Val F1	Test F1
1	0.6947	0.6966	0.7167	0.3397	0.3431	0.5600	0.5464
2	0.6744	0.6759	0.6724	0.3254	0.3046	0.5607	0.5428
3	0.7281	0.6690	0.6894	0.3237	0.3082	0.5636	0.5333

Cuadro 4.4: Métricas de los tres mejores modelos de la segunda fase

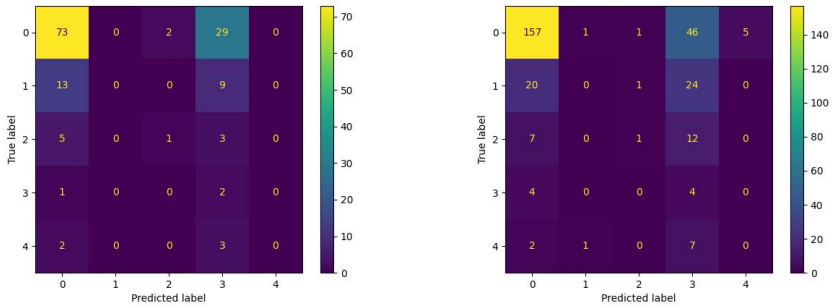
A continuación, las figuras 4.12, 4.13y 4.14, muestran las matrices de confusión de los modelos anteriores para validación y test de la primera parte. Vemos como la clase 0, que es ISUP 0 o 1, se acierta en la mayoría de casos. Sin embargo, con el resto de clases, ningún modelo ha aprendido a diferenciarlas, ya que ninguno ha obtenido un kappa mayor de 0.3, siguiendo la linea de los trabajos previos, donde el rango de kappa que obtuvieron era de entre -0.24 y 0.27. Es por esto, por lo que se intenta en la segunda parte con únicamente dos clases, aglutinando las etiquetas de 1 a 4 o ISUP mayor que 2.

En las figuras 4.15, 4.16 y 4.17 se exponen las matrices de confusión para validación y test de la segunda parte. En este caso, los modelos clasifican a la mayoría de casos positivos como tal. Además, con un kappa mayor que 0.3, es justo decir que los modelos han aprendido en cierto grado a diferenciar entre ambos casos, y que su desempeño es mejor que el azar.



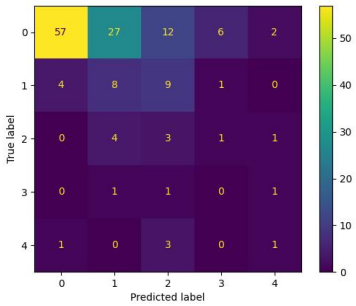
(a) Validación (b) Test

Figura 4.12: Matrices de confusión del primer modelo de 5 clases

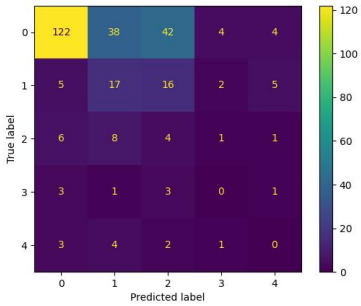


(a) Validación (b) Test

Figura 4.13: Matrices de confusión del segundo modelo de 5 clases

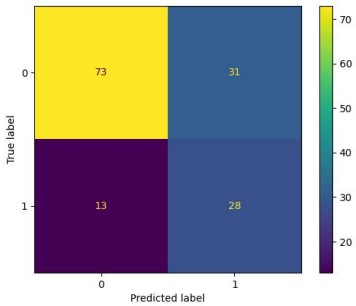


(a) Validación

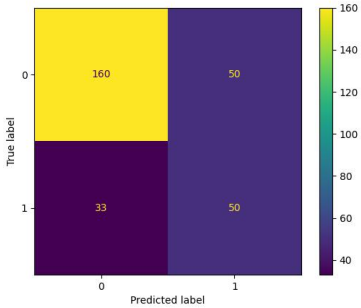


(b) Test

Figura 4.14: Matrices de confusión del tercer modelo de 5 clases

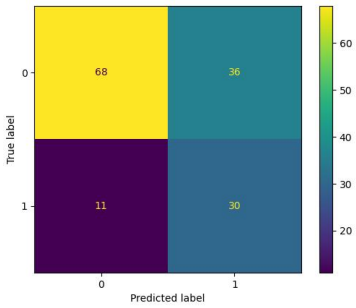


(a) Validación

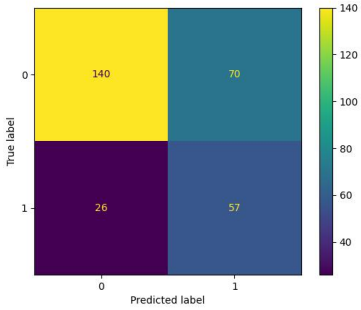


(b) Test

Figura 4.15: Matrices de confusión del primer modelo

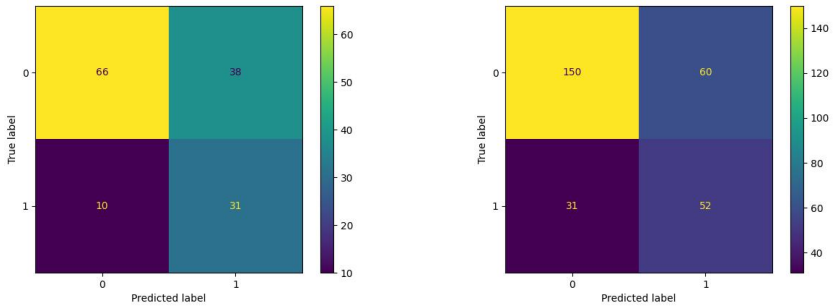


(a) Validación



(b) Test

Figura 4.16: Matrices de confusión del segundo modelo



(a) Validación (b) Test

Figura 4.17: Matrices de confusión del tercer modelo

Capítulo 5

Conclusiones y trabajo futuro

Como se ha ido explicando en las diferentes secciones, el trabajo ha consistido en 3 problemas distintos, en los que cada uno tiene un enfoque diferente a la hora de resolverlos. En un principio, el objetivo de este trabajo era únicamente la segmentación de las imágenes, pero al tener los datos de PI-CAI, se decidió continuar hasta la clasificación, con el fin de crear un algoritmo que pudiera servir de ayuda al diagnóstico del cáncer de próstata. Aunque este objetivo final se haya quedado lejos, podemos sacar algunas conclusiones.

En primer lugar, en la primera fase sí que se ha cumplido con el objetivo propuesto. Logrando segmentar la próstata con buena precisión y habiendo entrenado a la red con pocas imágenes, en concreto. En segundo lugar, tenemos la *adaptación de dominio*, donde se ha conseguido aumentar el *DICE* en un 2 % con respecto a la fase anterior, que puede parecer poco, pero teniendo en cuenta que la métrica se obtenía de la media de todas imágenes de PROSTATEx, un 2 % es reseñable, y más teniendo en cuenta que, una mejora en la generalización de imágenes que no ha visto, es un logro directo en el avance hacia su uso en un escenario real.

En tercer y último lugar, tenemos la fase sobre la *clasificación*. Aunque los resultados no hayan sido muy prometedores, no son malos resultados, y más teniendo en cuenta las dificultades con la que se ha trabajo, como puede ser la baja resolución de las imágenes, los datasets desbalanceados o las pocas imágenes disponibles. Con todo esto, los modelos no han sido capaces de diferenciar entre los 5 grados de cáncer. En el caso binario, aunque los resultados de los modelos han mejorado levemente, todavía están lejos de cumplir su cometido, siguiendo línea con la literatura previa.

Dicho esto, es un hecho que los algoritmos de *Deep Learning* están siendo muy útiles para la detección de enfermedades y se espera que su papel siga en aumento, convirtiéndose poco a poco en una herramienta indispensable para los expertos. Pero para ello, el trabajo futuro debe ir enfocado en mejorar la generalización.

Bibliografía

[1] McCarthy, J. (2007). What is artificial intelligence?.

[2] Minar, M. R., & Naher, J. (2018). Recent advances in deep learning: An overview. arXiv preprint arXiv:1807.08169.

[3] Siegel, R. L. (2022). Cancer statistics. URL: https://doi.org/10.3322/caac.21708

[4] Grönberg, H. (2003). Prostate cancer epidemiology. The Lancet, 361(9360), 859-864.

[5] Carr, Herman (1952) Free Precession Techniques in Nuclear Magnetic Resonance. URL: https://www.worldcat.org/es/title/76980558

[6] Heavey, S., Haider, A., Sridhar, A., Pye, H., Shaw, G., Freeman, A., & Whitaker, H. (2019). Use of magnetic resonance imaging and biopsy data to guide sampling procedures for prostate cancer biobanking. JoVE (Journal of Visualized Experiments), (152), e60216. DOI: 10.3791/60216

[7] Gleason, D. F. (1977). The Veterans Administration Cooperative Urological Research Group. Histological grading and clinical staging of prostatic carcinoma. Urologic pathology: the prostate, 171-198.

[8] Entrada de MedlinePlus: https://medlineplus.gov/spanish/ency/patientinstructions/000920.htm#:~:text=El%20sistema%20de%20puntuaci%C3%B3n%20de,crecimiento%20lento%20y%20no%20agresivo.

[9] Peter. A. Humphrey. "Gleason grading and prognostic factors in carcinoma of the prostate." In: Modern pathology : an official journal of the United States and Canadian Academy of Pathology 17(3), 292–306 (Mar. 2004). DOI: 10.1038/modpathol.3800054.

[10] Armato SG 3rd, Huisman H, Drukker K, Hadjiiski L, Kirby JS, Petrick N, Redmond G, Giger ML, Cha K, Mamonov A, Kalpathy-Cramer J, Farahani K. PROSTATEx Challenges for computerized classification of prostate lesions from multiparametric

magnetic resonance images. J Med Imaging (Bellingham). 2018 Oct;5(4):044501. doi: 10.1117/1.JMI.5.4.044501. Epub 2018 Nov 10. PMID: 30840739; PMCID: PMC6228312.

[11] Twilt, J. J., van Leeuwen, K. G., Huisman, H. J., Fütterer, J. J., & de Rooij, M. (2021). Artificial intelligence based algorithms for prostate cancer classification and detection on magnetic resonance imaging: a narrative review. Diagnostics, 11(6), 959.

[12] Rosenblatt, F. (1960). Perceptron simulation experiments. Proceedings of the IRE, 48(3), 301-309.

[13] Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning. arXiv preprint arXiv:1603.07285.

[14] Antonelli, M., Reinke, A., Bakas, S., Farahani, K., Kopp-Schneider, A., Landman, B. A., ... & Cardoso, M. J. (2022). The medical segmentation decathlon. Nature communications, 13(1), 1-13.

[15] Geert Litjens, Oscar Debats, Jelle Barentsz, Nico Karssemeijer, and Henkjan Huisman. "ProstateX Challenge data", The Cancer Imaging Archive (2017). DOI: 10.7937/K9TCIA.2017.MURS5CL

[16] https://pi-cai.grand-challenge.org/

[17] Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

[18] Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

[19] Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).

[20] Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160.

[21] Rao, C., & Liu, Y. (2020). Three-dimensional convolutional neural network (3D-CNN) for heterogeneous material homogenization. Computational Materials Science, 184, 109850.

[22] Serra, J., & Soille, P. (Eds.). (2012). Mathematical morphology and its applications to image processing (Vol. 2). Springer Science & Business Media.