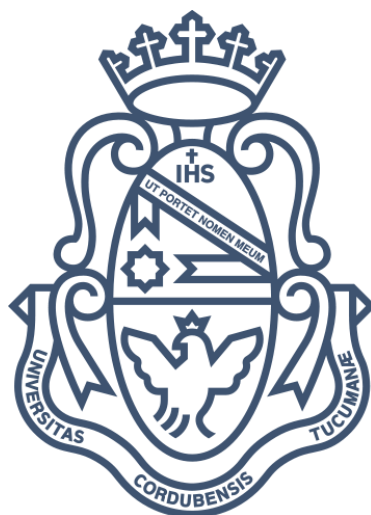


Universidad Nacional de Córdoba
Facultad de Ciencias Exactas, Físicas y Naturales
Facultad de Ciencias Médicas
Ingeniería Biomédica

**Predicción de distribución de dosis para
tratamientos de radioterapia utilizando redes
neuronales convolucionales: Aplicación en cáncer de
próstata.**



Alumnos:

Sadir, Inés – 42035104

Asesores:

Garrigó, Edgardo

Descamps, Caroline

Rulloni, Valeria Soledad

Córdoba, Abril 2023



Índice

1	Agradecimientos	9
2	Resumen	10
2.1	Palabras Clave	10
3	Abstract	11
3.1	Keywords	11
4	Glosario de abreviaturas	12
5	Introducción	13
5.1	Presentación de la problemática/caso o situación. Necesidades detectadas.	15
5.2	Motivación	15
6	Objetivos	16
6.1	Objetivos generales	16
7	Marco Teórico	17
7.1	Cáncer	17
7.1.1	Cáncer de próstata	17
7.2	Radioterapia (RT)	19
7.2.1	Principio de funcionamiento	19
7.2.1.1	Fraccionamiento	20
7.2.2	Tipos de RT	20
7.3	Radiación de haz externo (EBRT)	20
7.3.1	Técnicas	20
7.3.1.1	Radioterapia conformada tridimensional (3D-CRT)	20
7.3.1.2	Radioterapia de intensidad modulada (IMRT)	21
7.3.1.3	Radioterapia guiada por imágenes (IGRT)	21
7.3.1.4	Radioterapia de arco volumétrico modulado (VMAT)	22
7.3.1.5	Radioterapia estereotáctica	22
7.3.1.5.1	Radiocirugía (RC)	23
7.3.1.5.2	Radioterapia corporal estereotáctica (SBRT)	23
7.3.2	Volúmenes: ICRU Report 62	23
7.3.2.1	Volumen Tumor Macroscópico (GTV)	24
7.3.2.2	Volumen Blanco Clínico (CTV)	25
7.3.2.3	Volumen Blanco de Planificación (PTV)	25
7.3.2.4	Órganos de Riesgo (OAR)	25
7.3.3	Simulación	26
7.3.3.1	Inmovilización	27
7.3.4	Planificación de tratamiento	27

7.4	Knowledge-Based Planning (KBP)	28
7.4.1	Características	29
7.4.1.1	Métodos basados en OVH	29
7.4.1.2	Métodos basados en proyecciones	29
7.4.1.3	Métodos basados en DTH	29
7.4.2	Problemática	30
7.5	Deep Learning (DL)	30
7.5.1	Conceptos básicos	33
7.5.1.1	Descenso por el gradiente	33
7.5.1.2	Underfitting y Overfitting	34
7.5.1.3	Minilotes o minibatches	35
7.5.1.4	Parámetros e hiperparámetros	35
7.5.1.5	Descenso por el gradiente estocástico (SGD)	35
7.5.1.6	Retropropagación	36
7.5.1.7	Momentum	36
7.5.1.8	Tasa de aprendizaje o Learning Rate (LR)	37
7.5.1.9	Funciones de optimización con LR adaptativo	37
7.5.1.9.1	AdaGrad	37
7.5.1.9.2	ADAM	37
7.5.1.10	Función de activación	37
7.5.1.10.1	ReLU	38
7.5.1.10.2	Sigmoide	38
7.5.1.10.3	Softmax	38
7.5.1.11	Función de pérdida	39
7.5.1.11.1	MSE Loss	39
7.5.1.11.2	Sharp Loss	39
7.5.1.12	Dropout	40
7.5.1.13	Data augmentation	41
7.5.1.14	Batch Normalization (BN)	42
7.5.1.15	Métricas	42
7.5.1.15.1	IoU – índice de Jaccard	42
7.5.1.15.2	F1 score	42
7.5.1.15.3	Coeficiente de Dice	43
7.5.2	Deep Learning para KBP	43
7.5.3	Redes Neuronales Convolucionales (CNN)	43

7.5.3.1	Convolución, kernels y Max Pooling	44
7.5.3.2	Convolución transpuesta y Up-Sample	46
7.5.4	U-Net	47
7.5.5	Transfer Learning (TL) y ResNet	48
8	Materiales y Métodos	50
8.1	Dataset	50
8.1.1	Simulación del tratamiento y demarcación	50
8.1.2	Planificación	50
8.1.3	Preprocesamiento	51
8.2	Software y hardware	54
9	Implementación de las redes U-Net	55
9.1	Datos	55
9.2	Diseño de las redes	56
9.2.1	U-Net Básica	56
9.2.2	U-Net mejorada	57
9.2.3	ResU-Net	59
9.3	Entrenamientos / validaciones	60
9.4	Testeos	62
10	Resultados	65
10.1	Desempeño de los modelos	65
10.1.1	Primera etapa de entrenamientos	65
10.1.2	Segunda etapa de entrenamientos	68
10.2	Comparación con la planificación tradicional	71
10.2.1	Análisis de resultados para ejemplos particulares del set de testeo	71
10.2.2	Análisis global del desempeño de las predicciones	82
11	Discusiones	83
12	Conclusiones	86
13	Trabajos futuros	87
14	Bibliografía y Referencias	88
15	Anexos	91
15.1	Anexo I: Resumen de arquitecturas	91
15.1.1	U-Net Básica	91
15.1.2	U-Net mejorada	92
15.1.3	ResU-Net	94
15.2	Anexo II: Transfer learning: ResNet18 3D	96
15.3	Anexo III: Nabucodonosor vs Mendieta	99
15.4	Anexo IV: Códigos	100

Índice de figuras

Ilustración 1: Red U-Net desarrollada por Ronneberger O, Fischer P y Brox T. [7]	14
Ilustración 2: Sistema urogenital masculino [2]	17
Ilustración 3: Incidencia del tipo de cáncer más común en 2020 en cada país entre hombres [2]	18
Ilustración 4: Equipo acelerador lineal Trilogy del Centro de Radioterapia Dean Funes con el que se realizan los tratamientos con tecnología RapidArc™	22
Ilustración 5: Esquema de la relación entre los distintos volúmenes (GTV, CTV y PTV) en tres situaciones clínicas (A, B y C)[18]	24
Ilustración 6: Vista sagital, axial y coronal de TAC de abdomen con los contornos de GTV, PTV y los órganos de riesgo (vejiga, recto, cabezas femorales y bulbo peniano) para tratamiento de cáncer de próstata [19]	26
Ilustración 7: Histograma Dosis-Volumen relativo acumulativo de una planificación para tratamiento de cáncer de próstata y una predicción con una CNN [22]	28
Ilustración 8: Diagrama de Venn que muestra cómo el Deep Learning es un tipo de Machine Learning, los cuales se utilizan para muchos de los enfoques de la IA [23]	31
Ilustración 9: Neurona/nodo/unidad unitaria [24]	32
Ilustración 10: Ejemplo de una red neuronal [5]	33
Ilustración 11: Ejemplos de valores de $f(x)$ donde $f'(x) = 0$ [23]	34
Ilustración 12: Ejemplos de situaciones de ajuste insuficiente, ajuste apropiado y sobreajuste [23]	34
Ilustración 13: Representación de la acción del momentum. Las líneas de contorno representan una función de pérdida cuadrática condicionada [23]	36
Ilustración 14: Función ReLU [24]	38
Ilustración 15: Función sigmoidea [24]	38
Ilustración 16: Capa de salida con Softmax [24]	39
Ilustración 17: Gráfico de las funciones, en azul la función sigmoide original, en naranja la función con $\gamma = 25$ y en verde la función con $\gamma = 25$ y corrida 0.03 a la derecha [27]	40
Ilustración 18: Ejemplo de todas las variantes de una red que se entrenan cuando se le aplica una capa de dropout [23]	41
Ilustración 19: Representación de una imagen objetivo (1) y una imagen generada (2). En (3) se indica con el tono más oscuro la intersección de las dos figuras, $A \cap B$, y el total de la superficie gris indica la unión, $A \cup B$ [28]	42
Ilustración 20: Ejemplo de una red neuronal convolucional [5]	44
Ilustración 21: Ejemplo de una convolución en una imagen de 5x5 con un kernel 3x3, con stride de 1 y sin relleno de ceros [34]	45
Ilustración 22: Ejemplo de Max pooling 3x3 en una imagen de 5x5 con stride 1 [34]	46
Ilustración 23: Ejemplo de una convolución transpuesta donde a una imagen 3x3 (píxeles azules) se la convierte en una imagen de 5x5 (verde) con un kernel 3x3 (sombreado) [34]	47
Ilustración 24: Representación de una U-Net 3D para predicción de dosis [38]	47
Ilustración 25: Representación de una capa residual [41]	49
Ilustración 26: Representación de la ResNet18 [33]	49
Ilustración 27: Ejemplo de cortes en los tres ejes y gráfico 3D de los datos de un paciente en el software 3DSlicer, en verde el cuerpo del paciente, en rojo la concentración de dosis, en azul el PTV, naranja la vejiga y amarillo el recto.	51

Ilustración 28: Ejemplo de un corte transversal de un volumen de estructuras (en morado el recto, en amarillo la próstata y en rosado la vejiga) (izquierda), distribución de dosis correspondiente (centro) y superposición de ambas imágenes (derecha).	52
Ilustración 29: Representación 3D vista desde el frente de un volumen del conjunto original y su versión “espejada”	52
Ilustración 30: Representación de los cortes centrales en cada uno de los tres ejes, de un ejemplo aleatorio del dataset inicial, su versión del grupo “espejado” y ambos superpuestos con la distribución de dosis correspondientes	53
Ilustración 31: Ejemplificación de la rotación en la orientación de los gráficos generadas con la librería “matplotlib.pyplot”	53
Ilustración 32: Representación de la 4 ^o dimensión un corte de un volumen con One-hot encoding	55
Ilustración 33: Representación del aumento de dimensión de los tensores de datos	55
Ilustración 34: Representación conceptual de la arquitectura de la primera red implementada	56
Ilustración 35: Representación de un corte transversal de las estructuras (izquierda), el objetivo (centro) y la predicción (derecha) obtenida con el entrenamiento de 5000 épocas para un solo ejemplo con la red U-Net Básica	57
Ilustración 36: A: Representación de una operación ReflectionPad donde se genera un padding de 1 para convolucionar con kernels 3x3. B: Representación de zero padding para la misma situación	58
Ilustración 37: Representación del trabajo de Ronneberger et al. en donde se puede ver la utilización de la operación “ReflectionPad” a gran escala [7]	58
Ilustración 38: Representación de un corte transversal del objetivo (izquierda) y de la predicción (derecha) obtenida con el entrenamiento de 5000 épocas para un solo ejemplo con la red U-Net mejorada	59
Ilustración 39: Representación de un corte transversal del objetivo (izquierda) y de la predicción(derecha) obtenida con el entrenamiento de 5000 épocas para un solo ejemplo con la red ResU-Net	60
Ilustración 40: Representación de un corte transversal de un ejemplo de distribución de dosis con los valores de vóxel originales y el mismo corte con los valores redondeados para calcular las métricas Dice e IoU	61
Ilustración 41: Representación de un corte transversal del ejemplo ubicado en la posición 6 de las cuatro carpetas (Dosis, UB, UM y RU)	62
Ilustración 42: Gráfico del paper de Sumida et al., “Valores medios de índice de Dice de 16 pacientes evaluados entre la dosis-volumen predicha en los dos modelos con respecto al volumen de dosis de referencia” [44]	63
Ilustración 43: Gráfico DVH del volumen objetivo y predicción del paper de Muerakami et al [22]	64
Ilustración 44: Corte transversal aproximadamente a la mitad de la distribución de dosis y las predicciones de cada red para el volumen en la posición número 16 dentro las carpetas	71
Ilustración 45: Gráfico de índice de Dice para los intervalos porcentuales de dosis del ejemplo 16	72
Ilustración 46: One-hot encoding de los órganos del ejemplo 16 del set de testeo. De izquierda a derecha se tiene la máscara correspondiente al fondo, luego a la vejiga (que a esta altura de corte no presenta ningún pixel), el recto y al último la próstata.	72
Ilustración 47: Histogramas de las distribuciones de las dosis relativas vs la cantidad relativa de píxeles del objetivo (primera columna) y las predicciones para cada órgano (desde arriba recto, PTV y vejiga) para el ejemplo 16	73

Ilustración 48: Histogramas superpuestos de las predicciones y el objetivo para cada órgano del ejemplo 16	74
Ilustración 49: Gráfica DVH acumulado de todas las estructuras de todas las predicciones y el objetivo para el ejemplo 16	75
Ilustración 50: Corte transversal del one-hot encoding de los órganos del ejemplo 7 del set de testeo, de izquierda a derecha el fondo, el recto, la próstata y la vejiga	75
Ilustración 51: Corte transversal de la distribución de dosis objetivo y las predicciones de cada red para el ejemplo 7 del set de testeo	76
Ilustración 52: Gráfico de índice de Dice para los intervalos porcentuales de dosis del ejemplo 7	76
Ilustración 53: Histogramas superpuestos de las predicciones y el objetivo para cada órgano del ejemplo 7	77
Ilustración 54: Gráfica DVH acumulado de todas las estructuras para todas las predicciones y el objetivo para el ejemplo 7	78
Ilustración 55: Corte transversal del one-hot encoding del ejemplo 24 a la altura de corte 20 de 64	78
Ilustración 56: Corte transversal de la distribución de dosis objetivo y las predicciones de cada red para el ejemplo 24 del set de testeo, a la altura de corte 20 de 64	78
Ilustración 57: Corte transversal del one-hot encoding del ejemplo 24 a la altura de corte 24 de 64	79
Ilustración 58: Histogramas de distribución de dosis del objetivo y las predicciones para el recto del ejemplo 24	79
Ilustración 59: Gráfica DVH acumulado de todas las estructuras para todas las predicciones y el objetivo para el ejemplo 24	79
Ilustración 60: Corte transversal del one-hot encoding del ejemplo 6 a la altura de corte 32 de 64	80
Ilustración 61: Corte transversal de la distribución de dosis objetivo y las predicciones de cada red para el ejemplo 6 del set de testeo, a la altura de corte 32 de 64	80
Ilustración 62: Gráfico de índice de Dice para los intervalos porcentuales de dosis del ejemplo 6	80
Ilustración 63: Histogramas superpuestos de las predicciones y el objetivo para cada órgano del ejemplo 6	81
Ilustración 64: Gráfica DVH acumulado de todas las estructuras para todas las predicciones y el objetivo para el ejemplo 6	81
Ilustración 65: Gráfico de índice de Dice promedio para los intervalos porcentuales de dosis de los 30 ejemplos del set de testeo	82
Ilustración 66: Ejemplificación de un entrenamiento de 100 épocas de la red U-Net Básica con función de pérdida MSELoss y batch size de 2 en Nabucodonosor.	99
Ilustración 67: Ejemplificación de un entrenamiento de 100 épocas de la red U-Net Básica con función de pérdida MSELoss y batch size de 16 en la computadora Mendieta.	99
Ilustración 68: Gráficos de la evolución del valor de pérdida en entrenamiento y en validación de la res U-Net Básica con MSELoss en 100 épocas, con Mendieta y con Nabu respectivamente.	100

Índice de cuadros

Tabla 1: Características de cada entrenamiento y valores obtenidos	65
Tabla 2: Métricas de la segunda etapa de entrenamiento	69
Tabla 3: Summary de la primera U-Net implementada	91
Tabla 4: Summary de la segunda U-Net implementada	92
Tabla 5: Summary de la tercera U-Net implementada	94

1 Agradecimientos

Sería imposible nombrar a todas las personas con las que me cruce estos últimos años y les estoy agradecida por haber influido en esta etapa de mi vida. Muchos fueron los compañeros, profesores, familiares y amigos que, de alguna forma, participaron y dieron lugar a que llegara a este punto en mi desarrollo personal.

A Dios, por permitirme cumplir mis metas y por todas las personas, aventuras y oportunidades que puso en mi camino.

A Belén, porque risas no faltaron. Por hacerme espectadora de su eterna tragicomedia, ser la principal constante en mi vida y lo mejor que me llevo de estos últimos 5 años.

A mis compañeros, especialmente a Miguel y Cristian, por siempre darme una mano y hacer más fácil superar los muchos problemas a los que me enfrente en el desarrollo de este proyecto. Y a Belén, por todos los años de aguante y haber puesto “sobre la mesa” la posibilidad de estudiar esta carrera.

A mis asesores, presentarme este desafío y permitirme entrar a este inmenso mundo de la inteligencia artificial.

Al Centro de Computación de Alto Desempeño de la Universidad Nacional de Córdoba, por permitirme usar sus recursos computacionales para entrenar los modelos y estar disponibles para cualquier consulta.

A las Patris y Marce, por toda su dedicación y apoyo durante todos estos años, el cual, a fin de cuantas, me permitió dedicarme completamente a mis estudios y cumplir mis metas.

Y finalmente a mis padres, porque todo lo que soy es gracias a ellos.

2 Resumen

El propósito particular de este trabajo es construir y probar métodos de predicción de distribución 3D de dosis para tratamiento de cáncer de próstata con SBRT, utilizando modelos de aprendizaje profundo entrenados con datos históricos de pacientes tratados en el Centro de Radioterapia Dean Funes. La finalidad general es abrir paso al desarrollo de una herramienta capaz de predecir la dosis a partir de la anatomía del paciente, que se pueda integrar al proceso de creación de planes clínicos, y así, se exija menos tiempo y trabajo manual por parte de los físicos médicos.

Se crearon tres modelos de aprendizaje profundo basados en la arquitectura U-Net, para predecir las distribuciones 3D de dosis para planes de tratamiento de SBRT. Se incluyeron 135 casos de cáncer de próstata, de los cuales 120 fueron seleccionados aleatoriamente como conjunto de entrenamiento-validación y el resto como conjunto de prueba. Los datos de cada paciente eran 3 contornos: PTV, vejiga y recto. La base de datos inicial se duplicó generando copias con "espejado" sagital. Las redes propuestas parten de la arquitectura de Ronneberger et al. reemplazando las operaciones 2D con su versión 3D, agregando estrategias para mejorar la predicción (como dropout, reflection-pad and batch-normalization) y aplicando transfer learning con 3D-ResNet18. En todos los modelos, las entradas fueron cuatro canales de arreglos 3D de estructuras contorneadas, uno para PTV (próstata) y los otros para OAR (vejiga y recto) y fondo, y la salida el arreglo 3D de distribuciones de dosis. La precisión se evaluó con el índice de Dice y el coeficiente Intersection over Union (IoU) de los diferentes volúmenes. Para cada arquitectura se realizaron varios entrenamientos, modificando el número de épocas y la función de pérdida, para seleccionar una versión final y realizar un entrenamiento largo.

La U-Net Mejorada superó ampliamente a los otros dos modelos construidos, logró un índice de Dice para los intervalos de dosis hasta un 5% más altos que U-Net Básica y alrededor de un 10% mayor que el modelo de aprendizaje por transferencia, la ResU-Net. Todos los modelos dan los mejores resultados para vóxeles en las regiones de dosis baja y alta, sin embargo, las tres redes fracasan en predecir las dosis muy altas y son regulares para dosis intermedias. Finalmente, la U-Net Mejorada mostró un comportamiento consistente en todo el dataset de prueba, mientras que la U-Net Básica funcionó muy bien (incluso mejor que la Mejorada) con los ejemplos que tenían una "forma de próstata promedio", pero dio resultados muy pobres con ejemplos atípicos.

En conclusión, se desarrollaron y compararon tres modelos de Deep learning y se logró una predicción 3D de dosis en cáncer de próstata tratado con SBRT. Sin embargo, para obtener una potencial implementación clínica, estas tecnologías necesitan un mayor desarrollo para mejorar los resultados y finalmente obtener una planificación del tratamiento más precisa y eficiente.

Este trabajo utilizó recursos computacionales del CCAD de la Universidad Nacional de Córdoba (<https://ccad.unc.edu.ar/>), que forman parte del SNCAD del MinCyT de la República Argentina.

2.1 Palabras Clave

Aprendizaje profundo – Redes neuronales convolucionales – U-Net – Planificación de radioterapia – Cáncer de próstata

3 Abstract

The particular purpose of this work is to build and test methods to predict 3D-voxel-wise dose distributions for prostate cancer treatment with SBRT using deep learning models trained on historical data from patients treated at the Centro de Radioterapia Dean Funes. The overall goal is to pave the way for the development of a tool capable of predicting dose from patient anatomy, which can be used to create clinically acceptable plans that require less time and manual labor on the part of medical physicists.

Three deep learning models based on the U-Net architecture were created to predict 3D dose distributions for SBRT treatment plans. A total of 137 cases of prostate cancer were included, of which 120 cases were randomly selected as a training-validation set and the rest as a test set. Each patient's data contains 3 contours: PTV, bladder and rectum. The initial dataset was duplicated by data augmentation with sagittal reflections. The proposed networks extend the architecture from Ronneberger et al. by replacing 2D operations with their 3D counterparts, adding strategies to improve the network (as dropout, reflection-pad and batch normalization) and introducing transfer learning with the pre-trained 3D-ResNet18. The inputs were four channels of 3D arrays of contoured structures, one for PTV (prostate) and the others for OARs (bladder and rectum) and background, and the 3D array of dose distributions was taken as output for training in all the models. The predicted precision was evaluated with the Dice coefficient and the Intersection over Union coefficient (IoU) of different isodose volumes. For each architecture several training processes were performed, modifying the number of epochs and the loss functions, in order to select a final version and perform a long training.

The U-Net Mejorada model outperformed the other two models, it achieved Dice scores for isodose volumes as much as 5% higher than the U-Net Básica and about 10% than the transfer learning model, the ResU-Net. All the models give the best results for voxels in the low and the high dose regions however the three networks fall through predict the very high doses. Finally, the U-Net Mejorada showed consistent behavior across the entire dataset test, while the U-Net Básica performed very well (even better than the Mejorada) with the "regular prostate shape" samples but gave very poor results with atypical examples.

In conclusion, three deep learning models were developed and compared, achieving a dose prediction in 3D voxels for SBRT-treated prostate cancer. Nevertheless, to get a potentially clinical implementation, these technologies need further development to improve outcomes and ultimately obtain more accurate and efficient treatment planning.

This work used computational resources from CCDA – Universidad Nacional de Córdoba (<https://ccad.unc.edu.ar/>), which are part of SNCAD MinCyT, República Argentina.

3.1 Keywords

Deep learning – Convolutional Neural Networks – U-Net – Treatment Planning System – Prostate cancer

4 Glosario de abreviaturas

3D-CRT	Three-Dimensional Conformal Radiation – Radiación conformada 3D
AI / IA	Artificial Intelligence – Inteligencia Artificial
BN	Batch Normalization – Normalización por lotes
CNN	Convolutional Neural Network – Redes Neuronales Convolucionales
CTV	Clinical Target Volume – Volumen de objetivo clínico
DL	Deep Learning – Aprendizaje profundo
DVH	Dose-Volume Histogram – Histograma volumen-dosis
EBRT	External Beam Radiation Therapy – Radioterapia con haz externo
FCN	Fully Convolutional Network – Redes neuronales totalmente convolucionales
GPU	Graphics Processing Unit – Unidad de procesamiento gráfico
GTV	Gross Tumor Volume – Volumen tumoral macroscópico
ICRU	International Commission on Radiotherapy Units – Comisión Internacional de Unidades y Medidas de Radiación
IGRT	Image Guided Radiation Therapy – Radioterapia guiada por imágenes
IMRT	Intensity modulated radiation therapy – Radioterapia de intensidad modulada
KBP	Knowledge-Based Planning – Planificación basada en el conocimiento
LET	Linear Energy Transfer – Transferencia lineal de energía
LR	Learning Rate – Tasa de aprendizaje
ML	Machine Learning – Aprendizaje automático
MLC	Multileaf Collimator – colimador multiláminas
MLP	Multilayer Perceptron – Perceptrón multicapa
MP	Max Pooling – Agrupación por máximos
MRI	Magnetic Resonance Imaging – Resonancia magnética nuclear
MSE	Mean squared error – Error cuadrático medio
OAR	Organs at risk – Órganos de riesgo
PET	Positrons Emission Tomography – Tomografía por emisión de positrones
PTV	Planning Target Volume – Volumen objetivo de planificación
RA	RapidArc™
RC	Radiocirugía
ReLU	Rectified Lineal Unit – Unidad lineal rectificada
RT	Radiotherapy – Radioterapia
SBRT	Stereotactic Beam Radiation Therapy – Radioterapia Estereotáctica Corporal
SGD	Stochastic Gradient Descent – Descenso por el gradiente estocástico
TBI	Total Body Irradiation – Irradiación corporal total
TC/TAC	Computerized Axial Tomography – Tomografía Axial Computarizada
TL	Transfer learning – Aprendizaje por transferencia
TPS	Treatment Planning System – Sistema de planificación de tratamiento
VMAT	Volumetric Modulated Arc Therapy – Arcoterapia Volumétrica de Intensidad Modulada

5 Introducción

El cáncer es un grupo de enfermedades que se caracterizan por el crecimiento anormal y descontrolado de las células. Puede ser causado tanto por factores externos (tabaco, organismos infecciosos, químicos y radiación) como por factores internos (mutaciones heredadas, hormonas, condiciones inmunológicas y mutaciones que ocurren a partir del metabolismo) [1]. El cáncer se ubica como una de las principales causas de muerte y es una gran barrera para aumentar la esperanza de vida en todos los países del mundo.

Con un estimado de casi 1,4 millones de casos nuevos y 375.000 muertes en todo el mundo en 2020, el cáncer de próstata es el segundo cáncer más frecuente y la quinta causa de muerte por cáncer entre los hombres [2]. Los únicos factores de riesgo bien establecidos para el cáncer de próstata son la edad, la raza/etnicidad y los antecedentes familiares de la enfermedad. Alrededor del 62 % de todos los casos de cáncer de próstata se diagnostican en hombres de 65 años o más, y el 97 % ocurre en hombres de 50 años o más, siendo la incidencia mayor en afroamericanos y menor en países asiáticos. Los estudios genéticos sugieren que una fuerte predisposición familiar puede ser responsable del 5% al 10% de los cánceres de próstata.

Las opciones de tratamiento varían según la edad, la etapa y el grado del cáncer, así como de otras afecciones médicas, y deben analizarse con el médico de la persona. Para tratar la enfermedad en etapa temprana se recurre a cirugía, radiación de haz externo, implantes de semillas radioactivas o braquiterapia de alta tasa de dosis. Para tratar la enfermedad más avanzada se usan la terapia hormonal, la hemoterapia, la radioterapia (RT) o una combinación de estos tratamientos [1].

Aproximadamente el 50% de todos los pacientes con cáncer se someten a RT durante el curso de su enfermedad. Las innovaciones tecnológicas de las últimas décadas han logrado mejoras significativas en el doble objetivo dosimétrico: preservar los órganos críticos en riesgo (OAR, Organ at Risk) al mismo tiempo que se optimiza la aplicación de las altas dosis en el órgano objetivo [3]. La llegada y avances de modalidades innovadoras, como la radioterapia de intensidad modulada (IMRT) y la terapia de arco modulado por volumen (VMAT), generaron que la calidad de los planes de tratamiento de RT mejorarán drásticamente. Sin embargo, tal desarrollo requiere el uso de algoritmos más complejos y tiene como costo el aumento de los ajustes iterativos por parte de los físicos médicos y el tiempo de planificación [4].

Para mejorar la eficiencia de la planificación del tratamiento, se han desarrollado enfoques de planificación del tratamiento basados en el conocimiento que se adquiere de casos anteriores para predecir el resultado de un nuevo caso. Este paradigma es utilizado por investigadores desde hace más de una década en forma de planificación basada en el conocimiento (KBP, por sus siglas en inglés) [3]. En esencia, el KBP es la utilización del conocimiento existente y los datos de planificación anteriores para automatizar la selección de parámetros y así mejorar la eficiencia y la calidad de la planificación del tratamiento [5].

El KBP tradicional, ampliamente utilizado en la actualidad, incluye métodos que utilizan características anatómicas y geométricas (distancia a las estructuras objetivo, volúmenes del objetivo y estructuras OAR, etc.) para construir un modelo estadístico que luego se usa para predecir las características dosimétricas para un nuevo caso. Un software rápido puede proporcionar una solución en segundos, sin embargo, el físico médico todavía tiene que ajustar

manualmente los objetivos hasta que se logra la distribución de dosis deseada [3]. Esto ocurre porque estos métodos requieren la enumeración manual de características para alimentar un modelo para la predicción de dosis y los histogramas dosis-volumen (DVH por sus siglas en inglés) [4].

Por otro lado, las tecnologías de planificación de tratamiento tradicionales (sin aplicación de KBP), que también son ampliamente utilizadas en la actualidad, adoptadas, por ejemplo, por los sistemas de planificación comerciales como Eclipse, se basan únicamente en los histogramas dosis-volumen (DVH) y volúmenes de dosis máximos/mínimos para OAR. Uno de los principales y más relevantes inconvenientes de estos enfoques tradicionales es la selección manual de características, que no llega a capturar verdaderamente los detalles geométricos en una anatomía tridimensional (3D) y requieren muchos ajustes y replanificación para garantizar que el plan resultante cumpla con el DVH prescrito. Obtener resultados aceptables con este método, puede generar una sobrecarga computacional significativa y hacer que sea inviable la planificación en tiempo real. Con este panorama, el Deep Learning (DL) se presenta como un enfoque muy prometedor [5].

En los últimos años, el Deep learning o aprendizaje profundo ha dado un salto cualitativo en el avance de muchas áreas. Una en particular fue la progresión de las arquitecturas de redes neuronales convolucionales (CNN) para procesamiento de imagen y visión artificial. En 2015, Shelhamer et al. propusieron las redes totalmente convolucionales (FCN, por sus siglas en inglés) [6] que superaron las técnicas existentes de la época en tareas de segmentación semántica. A partir de la idea de FCN, Ronneberger et al. generaron una arquitectura perfeccionada, el modelo llamado U-Net [7], que se centró en la segmentación semántica de imágenes biomédicas y fue el disparador de este y muchos otros trabajos que se desarrollaron desde entonces. En el diseño de la arquitectura U-net había tres ideas centrales: 1) una gran cantidad de operaciones de Max Pooling para permitir que los filtros de convolución encontrarán características globales, y no locales o puntuales en las imágenes (flechas rojas en la ilustración 1), 2) operaciones de convolución transpuestas (mal llamadas deconvolución), para devolver la imagen a su tamaño original (flechas verdes), y 3) copias de los mapas de características de la rama descendente de la red U en la rama ascendente, para preservar las características locales de nivel inferior (flechas grises) [4].

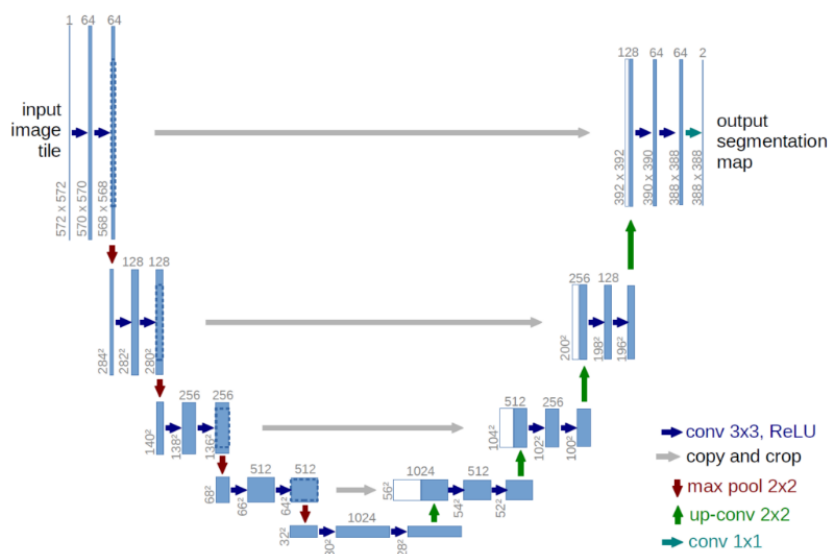


Ilustración 1: Red U-Net desarrollada por Ronneberger O, Fischer P y Brox T. [7]

Debido a la cantidad limitada de datos que se suele tener disponibles en el campo de la radioterapia, el uso de la experiencia y conocimiento particular del físico médico para planificación de tratamiento suele ser muy útil, pero recientemente las investigaciones apuestan fuertemente en el DL para reducir la dependencia de las funciones artesanales que se utilizan en la clínica actual y permitir que una red aprenda sus propias funciones para realizar las predicciones de tratamientos para nuevos casos.

En definitiva, aunque la U-Net y otras arquitecturas se idearon en un principio para la tarea de identificación de objetos y segmentación de imágenes, varios estudios han demostrado que, con algunas modificaciones, son capaces de predecir con precisión una distribución de dosis a nivel de vóxel (pixel volumétrico o tridimensional) a partir de los contornos del paciente, aprendiendo a abstraer sus propias características generales y locales de alto nivel [4]. Esto abre un nuevo horizonte de posibilidades para la investigación y avance en el área de planificación de radioterapia, dando lugar a las tecnologías de inteligencia artificial (IA) que en un futuro pueden llegar a ser métodos más eficientes de aplicación clínica de planificación de los tratamientos de radioterapia.

A partir de esta idea, se plantea como objetivo del siguiente proyecto integrador construir y entrenar unas redes U-Net para estudiar su aplicación en la predicción de la distribución de dosis en planificaciones de tratamiento de cáncer de próstata con datos propios de planificaciones de tratamiento de RapidArc™ (RA).

5.1 Presentación de la problemática/caso o situación. Necesidades detectadas.

Este proyecto parte de que la esencia en la creación de un plan de tratamiento de radioterapia es el manejo de los requisitos de diferentes estructuras que se encuentran en conflicto, es decir, la minimización de la dosis entregada a los órganos en riesgo al mismo tiempo que se deposita la dosis prescrita necesaria para tratar el tejido cancerígeno.

Hoy en día, en la clínica se utilizan los métodos tradicionales de planificación de RT, que requieren un diseño cuidadoso y la selección manual de características, que demandan mucho tiempo y experiencia de parte del físico médico. Se reconoce que este tipo de modelado está lejos de ser perfecto, y la determinación de un gran número de parámetros requiere bastante trabajo y reduce sustancialmente la eficiencia del proceso. Es por esto que, en la actualidad se busca investigar alternativas para optimizar esta tarea, y el foco principal está en las tecnologías de inteligencia artificial, más específicamente en los modelos de DL.

5.2 Motivación

La motivación comunitaria de esta línea de investigación, es desarrollar una herramienta capaz de predecir la dosis a partir de la anatomía del paciente (volúmenes dibujados) basada en inteligencia artificial, que se pueda utilizar para crear planes completos clínicamente aceptables que requieran menos tiempo y trabajo manual por parte de los físicos médicos.

6 Objetivos

6.1 Objetivos generales

Investigar y poner en práctica arquitecturas de CNN para la predicción de la distribución de dosis para la planificación del tratamiento con radioterapia para pacientes con cáncer de próstata tratados con RapidArc™.

6.2 Objetivos específicos

- Examinar, depurar, optimizar y aumentar la base de datos generada durante las prácticas profesionales supervisadas (PPS)
- Implementar una primera CNN con arquitectura U-Net
- Optimizar la CNN
- Implementar y comparar otras arquitecturas y técnicas
- Analizar los resultados

7 Marco Teórico

7.1 Cáncer

Según la Organización Mundial de la Salud, la palabra cáncer es un término genérico que abarca un amplio grupo de enfermedades que pueden afectar a cualquier parte del organismo. Su característica principal es la multiplicación rápida y agresiva de células alteradas que se extienden más allá de sus límites habituales y pueden propagarse a otros órganos dando lugar a las conocidas metástasis.

Las alteraciones celulares son el resultado de la interacción entre factores genéticos y tres categorías de agentes externos: carcinógenos físicos, como las radiaciones ionizantes; carcinógenos químicos, sustancias contenidas en el humo de tabaco, por ejemplo; carcinógenos biológicos, como algunos virus, bacterias y parásitos.

Hay más de cien tipos diferentes de cáncer y su incidencia aumenta considerablemente con la edad, probablemente porque se van acumulando factores de riesgo. Además, a esta acumulación global se suma la pérdida progresiva con la edad de eficacia de los mecanismos de reparación celular [8].

7.1.1 Cáncer de próstata

La próstata es una glándula que forma parte del sistema urogenital masculino (Ilustración 2). Tiene tamaño y forma aproximada de una nuez y se encarga de secretar un líquido blanquecino y viscoso que permite la movilidad de los espermatozoides en la eyaculación. Esta glándula se ubica justo debajo de la vejiga rodeando la uretra, frente al recto y cerca de la base del pene [9].

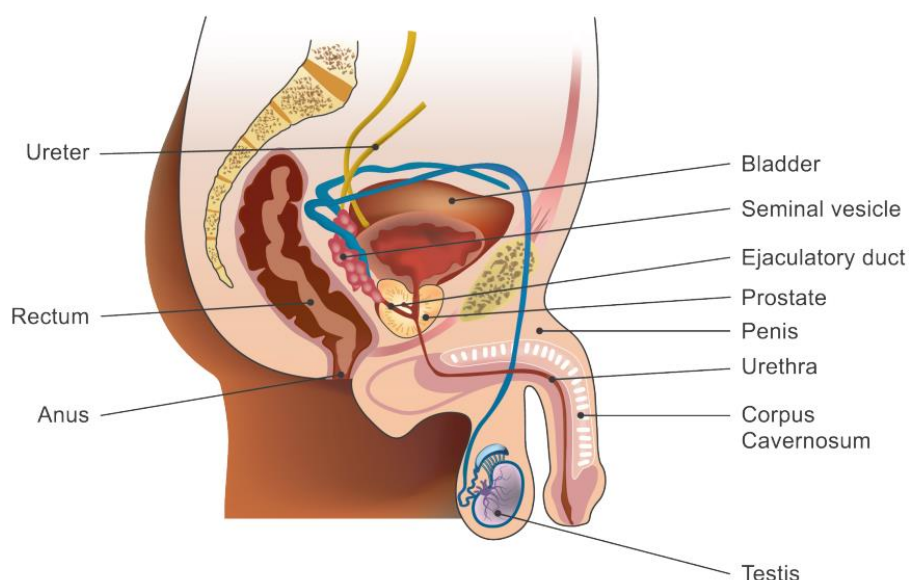


Ilustración 2: Sistema urogenital masculino [2]

El cáncer de próstata fue el segundo cáncer más frecuente y la quinta causa de muerte por cáncer entre hombres en 2020. Además, es el cáncer más diagnosticado en hombres en más de la mitad (112 de 185) de los países del mundo (Ilustración 3) [2]. La incidencia del cáncer de próstata en todo el mundo se correlaciona principalmente con el aumento de la edad, siendo la

edad promedio en el momento del diagnóstico de 66 años. La tasa de mortalidad también aumenta con la edad, casi el 55% de todas las muertes ocurren después de los 65 años [10].

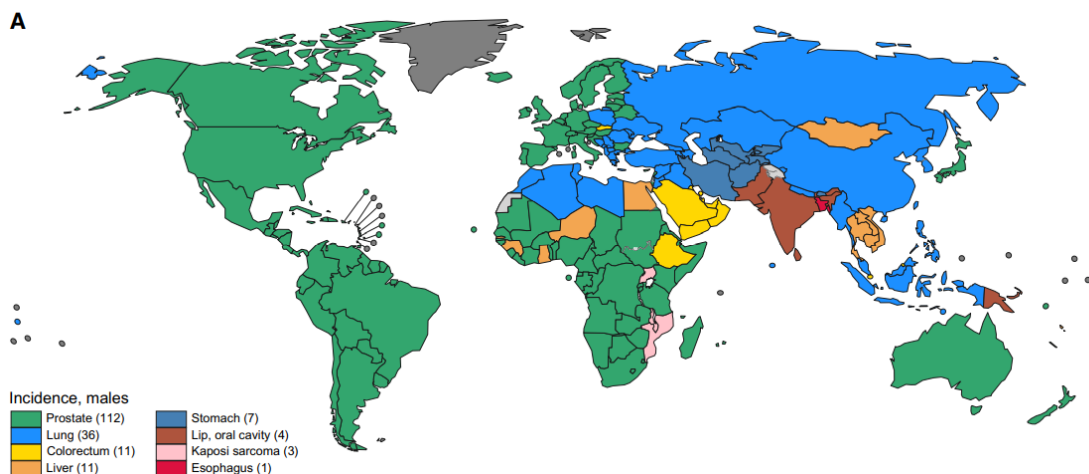


Ilustración 3: Incidencia del tipo de cáncer más común en 2020 en cada país entre hombres. El número de países representados en cada grupo de clasificación se incluye en la leyenda [2]

Aunque es una enfermedad bastante frecuente, se sabe relativamente poco sobre su etiología. Los factores de riesgo establecidos se limitan a la edad avanzada, los antecedentes familiares, ciertas mutaciones genéticas (por ejemplo, de BRCA1 y BRCA2) y afecciones (síndrome de Lynch). Se han identificado pocos factores ambientales y de estilo de vida para los cuales la evidencia es convincente, estos se pueden resumir a fumar y el exceso de peso corporal.

Las tasas de mortalidad han disminuido en la mayoría de los países de ingresos altos desde mediados de la década de 1990, incluidos los de América del Norte, Oceanía y el norte y el oeste de Europa. Esto se debe probablemente a los avances tecnológicos en el tratamiento y la detección más temprana a través de un mayor número de exámenes de detección [2]. La mayoría de los cánceres de próstata se detectan por niveles plasmáticos elevados de antígeno prostático específico (PSA > 4 ng/mL). Sin embargo, debido a que también se han encontrado hombres sin cáncer con PSA elevado, la biopsia de tejido es el estándar para confirmar la presencia de cáncer [9]. Más del 90 % de todos los cánceres de próstata se descubren en estadios locales, para los cuales la tasa de supervivencia relativa a 5 años se acerca al 100% [1].

El cáncer de próstata puede ser asintomático en la etapa inicial y, a menudo, puede requerir un tratamiento mínimo. Sin embargo, la queja más frecuente es la dificultad para orinar y la nicturia (micción nocturna frecuente), que pueden surgir de la hipertrofia prostática. La etapa más avanzada de la enfermedad puede presentarse con retención urinaria y dolor de espalda, ya que el esqueleto es el sitio más común de enfermedad metastásica ósea [10].

Las opciones de tratamiento varían según la edad, la etapa y el grado del cáncer, así como de otras afecciones médicas, que deben analizarse con el médico especialista. En etapa temprana de la enfermedad, se puede tratar con cirugía, radiación de haz externo (RT), implantes de semillas radioactivas o braquiterapia de alta tasa de dosis. Para tratar la enfermedad más avanzada, se utilizan la terapia hormonal, la quimioterapia, la RT o una combinación de estos tratamientos. Todos estos tratamientos pueden generar efectos secundarios, que incluyen dificultades urinarias y eréctiles, afectando la calidad de vida del paciente [1].

7.2 Radioterapia (RT)

Después del descubrimiento de los rayos X en 1895, por Wilhelm Conrad Röntgen, se comenzó a apreciar su utilidad clínica como medio de tratamiento del cáncer. Desde entonces, la radioterapia (RT) se ha convertido en una especialidad médica reconocida, la radio-oncología es una disciplina en la que los especialistas trabajan junto con otros profesionales de la salud [11]. El desarrollo de la irradiación de megavoltaje (unidades de cobalto y aceleradores lineales) hace 5 décadas, impulsó la RT como terapia principal para el cáncer temprano, con tasas de curación superiores al 70% en ciertos tipos de tumores, como carcinomas de cuello uterino, de próstata y de vejiga. La posterior introducción de la tomografía computarizada (TAC) como herramienta para la planificación del tratamiento de RT mejoró aún más la precisión de los campos de radiación y la reducción de su toxicidad [12].

Aproximadamente el 50% de todos los pacientes con cáncer recibirán RT durante el curso de su enfermedad. Esta terapia es actualmente un componente esencial en el manejo de los pacientes con cáncer, ya sea sola o en combinación con cirugía, quimioterapia u hormonoterapia. De los pacientes con cáncer que se curan, se estima que alrededor del 40% lo hace con RT sola o combinada con otras modalidades [13]. Además, la RT es muy eficaz en la paliación de los síntomas del cáncer como el dolor, el sangrado y la obstrucción de órganos [11].

7.2.1 Principio de funcionamiento

La radiación utilizada para destruir las células cancerosas se denomina radiación ionizante, porque elimina electrones de los átomos y forma iones (partículas cargadas eléctricamente). La mayoría de los tipos de partículas de radiación (electrones, protones, iones de carbono) son directamente ionizantes, pueden alterar directamente la estructura atómica del medio absorbente a través del cual pasan. Por otro lado, la radiación electromagnética (fotones con energía superior a 10 keV, los rayos X y γ), son indirectamente ionizantes porque no producen daños por sí mismos, sino que producen electrones secundarios que son los que generan los iones [13].

La energía depositada puede matar directamente las células cancerosas o puede inducir varios tipos de cambios genéticos que dañan el ADN (ácido desoxirribonucleico) y así, bloquear su capacidad para dividirse y proliferar aún más, también causando su muerte. Las células normales proliferan más lentamente en comparación a las células cancerosas y, por lo tanto, tienen tiempo para reparar el daño antes de la replicación. Las células cancerosas, en general, no son tan eficientes para reparar el daño causado por la radiación, lo que resulta en una destrucción diferencial.

Por otro lado, la RT no mata las células cancerosas de inmediato. Se necesitan horas, días o hasta semanas de tratamiento antes de que las células cancerosas comiencen a morir, y continúan muriendo durante semanas o meses después de que finaliza la RT. Su eficacia biológica depende de la transferencia lineal de energía (LET), la dosis total, la tasa de dosis, el fraccionamiento, y la radiosensibilidad del tejido objetivo. Si bien la radiación daña tanto tejidos normales como tejidos cancerosos, el objetivo de la RT es maximizar la dosis de radiación a los tejidos cancerosos y minimizar la exposición a los tejidos normales que se encuentran próximos o en el camino de la radiación [11].

7.2.1.1 Fraccionamiento

Se ha demostrado que el resultado general del tratamiento de RT mejora al fraccionar en varias sesiones el suministro de la dosis total. Los factores más importantes que intervienen en la eficacia de la RT fraccionada están basados en el concepto de las "cuatro R": reparación, redistribución, reoxigenación y repoblación, y en los últimos años, se ha agregado "radiosensibilidad" formando las "cinco R" [13].

La RT administrada en un régimen fraccionado se basa en la diferencia en las propiedades radiobiológicas del tejido maligno y los tejidos normales. Esta estrategia amplifica la ventaja de supervivencia de los tejidos normales sobre las células cancerosas, en gran parte basándose en su mayor capacidad de reparación del daño subletal por radiación en comparación con las células cancerosas. Las células normales proliferan más lentamente en comparación con las células cancerosas y, por lo tanto, tienen tiempo para reparar el daño antes de la replicación.

Las observaciones iniciales de los efectos de la radioterapia fraccionada en la década de 1920 finalmente llevaron al desarrollo de regímenes que comparaban diferentes esquemas de tratamiento según la dosis total, el número de fracciones y el tiempo total de tratamiento [11].

7.2.2 Tipos de RT

Hay dos formas de administrar radiación al tejido objetivo. La radiación interna o braquiterapia se administra desde el interior del cuerpo mediante fuentes radiactivas, selladas en catéteres o semillas directamente en el sitio del tumor. Y la radiación de haz externo se administra desde el exterior del cuerpo dirigiendo rayos de alta energía (fotones, electrones, protones o radiación de partículas) a la ubicación del tumor [11].

7.3 Radiación de haz externo (EBRT)

Según la definición del National Cancer Institute (NCI de EE. UU.), la RT de haz externo (EBRT) es una terapia en la que una fuente externa dirige la radiación hacia el tumor, desde el exterior del cuerpo. Si bien los rayos X y β (electrones) son las fuentes más utilizadas para la RT, existen algunos centros que operan programas que emplean haces de partículas más pesadas, los protones [14].

Los desarrollos en imágenes junto con los avances en la tecnología informática han cambiado fundamentalmente los procesos de localización de tumores y planificación de la RT. La capacidad de mostrar información anatómica en una selección infinita de vistas ha llevado al surgimiento de la radioterapia conformada tridimensional (3D-CRT) [13]. Posteriormente, las innovaciones tecnológicas de las últimas décadas, han impulsado la transición a la radioterapia de intensidad modulada (IMRT). Finalmente, los avances algorítmicos permitieron la extensión de la IMRT de campo estático a la terapia de arco volumétrico modulado (VMAT) [3].

7.3.1 Técnicas

7.3.1.1 Radioterapia conformada tridimensional (3D-CRT)

La introducción de las computadoras en la década de 1980 cambió la oncología radioterapéutica haciendo posibles cálculos matemáticos iterativos, brindando información sobre la anatomía real del paciente y permitiendo la realización de una dosimetría más confiable. Con todas estas mejoras, los cálculos de dosis se pudieron comenzar a realizar en todo un volumen, en lugar de solo en un punto como se venía trabajando, y esto condujo a la creación

de la radioterapia conformada tridimensional (3D-CRT) a principios de la década de 1990. Esta técnica se basa en la simulación del tratamiento con tomografía axial computarizada (TAC), que permite la localización precisa del tumor y las estructuras críticas de los órganos sanos. Luego con la transferencia en línea de los datos del paciente y elaboración con paquetes informáticos, se obtienen los campos de radiación basado en la vista de haz conformado y, finalmente, la reproducción de la planificación por un acelerador lineal, generalmente dotado con un colimador multiláminas (MLC). En resumidas cuentas, se consigue que el volumen tratado sea más estrechamente el volumen del tumor y que la toxicidad de la radiación se reduzca manteniendo los mismos niveles de dosis [13].

7.3.1.2 Radioterapia de intensidad modulada (IMRT)

Cuando un tumor no está bien separado de los órganos circundantes en riesgo y/o tiene una forma cóncava o irregular, puede ser imposible generar una combinación práctica de haces de intensidad uniforme que trate el tumor de forma segura y no afecte a los órganos sanos. En estos casos, agregar intensidad modulada a la configuración del haz permite una conformidad mucho más estricta con los objetivos.

La radioterapia de intensidad modulada (IMRT) es un tipo avanzado de 3D-CRT que utiliza varias aperturas de haz en el mismo ángulo y modula la "intensidad" (fluencia) de ese haz, es decir, asigna intensidades no uniformes a pequeñas subdivisiones de los haces. Esto consigue el diseño personalizado de distribuciones de dosis óptima, lo que ayuda a obtener un mejor control del tumor y una menor toxicidad en el tejido normal.

Hay varias técnicas para implementar la IMRT, miles de posibles soluciones de disposición de haces para cumplir con un objetivo de tratamiento específico, con una variedad de distribuciones de dosis diferentes para el OAR. Probarlos todos hasta encontrar el que mejor se adapte sería una tarea muy engorrosa y casi imposible sin la ayuda de un programa informático. Por lo tanto, esta tarea la lleva a cabo el software de planificación utilizando un enfoque de planificación inversa. La persona que planifica establece los objetivos del rango de dosis y las restricciones de dosis en los diferentes volúmenes planificados y OAR. El software, clasificando estos planes por funciones de costo, produce la "mejor" solución de plan posible [13].

7.3.1.3 Radioterapia guiada por imágenes (IGRT)

Con la radioterapia guiada por imágenes (IGRT) se logra el aumento de la precisión de la RT mediante la obtención de imágenes justo antes, durante y después de cada sesión, actuando sobre estas imágenes para adaptar el tratamiento en tiempo real. Existen varias opciones de guía de imágenes disponibles: TAC no integrada, imágenes de rayos X integradas (kV), marcadores implantados activos, ultrasonido, TAC de un solo corte, TAC convencional y TAC de haz cónico integrado. La disponibilidad de sistemas de imágenes de alta calidad y el registro automático de imágenes han dado lugar a muchas aplicaciones clínicas nuevas, como tratamientos hipofraccionados de alta precisión [13].

7.3.1.4 Radioterapia de arco volumétrico modulado (VMAT)

La terapia de arco volumétrico modulado (VMAT) es un avance de la IMRT, esta permite llegar a una distribución de dosis tridimensional esculpida con precisión con, generalmente, 1-3 vueltas completas o parciales del pórtico del acelerador lineal y el control electrónico de la posición de cada lámina del MLC. Esto es posible gracias a un algoritmo, como RapidArc™ (Varian Medical Systems, Palo Alto, CA), que cambia simultáneamente tres parámetros durante el tratamiento: la velocidad de rotación del pórtico, la forma de la apertura de las hojas de MLC y la tasa de suministro de dosis [13]. La VMAT se puede administrar utilizando una tasa de dosis constante (cdr-VMAT) o una tasa de dosis variable (vdr-VMAT) durante la rotación del pórtico. En comparación con IMRT, las ventajas potenciales de VMAT incluyen una gran reducción en el tiempo de tratamiento y una reducción paralela de la dosis corporal integral, es decir, reducción del riesgo de neoplasias malignas secundarias [15].

En el tratamiento del cáncer de próstata, el VMAT proporciona una eficiencia algo mejorada en comparación con la IMRT. Estudios recientes indicaron ventajas clínicas potenciales para IMRT y VMAT en comparación con 3D-CRT, donde la planificación de VMAT se mostró como la modalidad más efectiva para mantener o mejorar la cobertura de PTV con la mayor reducción en la dosis rectal y vesical [16].



Ilustración 4: Equipo acelerador lineal Trilogy del Centro de Radioterapia Dean Funes con el que se realizan los tratamientos con tecnología RapidArc™

7.3.1.5 Radioterapia estereotáctica

Esta técnica consiste en la administración de dosis individuales de radiación muy altas en unas pocas fracciones de tratamiento, lo que da como resultado una alta dosis biológica efectiva (BED). La estereotaxia hace referencia a la localización precisa de la lesión apoyándose en algún sistema de imágenes. Gracias a esa localización precisa es que se puede ajustar los haces y entregar mayor dosis. Se utiliza, por ejemplo, para extirpar tumores primarios y oligometastásicos pequeños y bien definidos en etapa temprana en ubicaciones en las cavidades abdominopélvica y torácica, y en sitios espinales y paraespinales. Debido a la alta dosis de radiación, es probable que se dañe cualquier tejido inmediatamente adyacente al tumor, sin embargo, como la cantidad de tejido normal en la región de dosis es pequeña, la toxicidad clínicamente significativa es baja [11].

7.3.1.5.1 Radiocirugía (RC)

Esta modalidad se aplica en el tratamiento de tumores intracraneales. El componente estereotáctico de la técnica se refiere a la inmovilización del paciente con un sistema de armazón rígido para la cabeza o una máscara termomoldeable, que establece un sistema de coordenadas específico del paciente para todo el proceso de tratamiento. Después de la colocación del marco o máscara se realiza un estudio de imagen TAC para localizar el volumen objetivo en relación con las coordenadas del marco de la cabeza. Se puede administrar mediante un acelerador lineal o equipos diseñados exclusivamente para este tratamiento, como puede ser el GammaKnife o el CyberKnife [13].

7.3.1.5.2 Radioterapia corporal estereotáctica (SBRT)

En el resto del cuerpo, se la denomina SBRT a esta técnica caracterizada por un hipofraccionamiento de la dosis. La SBRT intenta proporcionar una ventaja clínica en relación con la RT convencional mediante la reducción de la dosis a los tejidos normales y las estructuras críticas, y la maximización de la cobertura del tumor mediante el uso de técnicas precisas de localización del tumor, inmovilización del paciente, planificación especializada y guía por imágenes. En su mayoría se limita la elegibilidad a tumores bien delimitados con un diámetro de sección transversal máximo de hasta 5 cm, aunque algunos centros han informado resultados para tumores de hasta 7 cm.

Existen varias características que distinguen a la SBRT de la RT convencional, incluyendo un aumento general en la cantidad de haces, uso de disposiciones de haces no coplanares, márgenes de haz pequeños o nulos para la penumbra, el uso de distribuciones de dosis no homogéneas y técnicas de dose-painting, incluida la IMRT [17]. El número limitado de fracciones de tratamiento hace que la SBRT sea más conveniente para el paciente que la RT tradicional, esta modalidad de tratamiento permite que el paciente esté menos tiempo fuera de su casa (ya que generalmente los centros de radioterapia se encuentran solamente en grandes ciudades) lo que implica menos desarraigo y permite poder reintegrarse a la vida laboral y social más rápidamente, con menos gastos de hotelería y movilidad. También, a los centros de radioterapia le permite tratar más pacientes reduciendo o evitando las listas de espera ya que los turnos de tratamiento se desocupan más rápidamente, y además, como duran menos se pueden facturar más rápido.

7.3.2 Volúmenes: ICRU Report 62

Como se detalla en el reporte 62 de la Comisión Internacional de Unidades y Medidas de Radiación (ICRU)[17], la determinación del tejido a irradiar se consigue definiendo volúmenes que encierran dichos tejidos (Ilustración 5) y puntos de referencia para localizarlos.

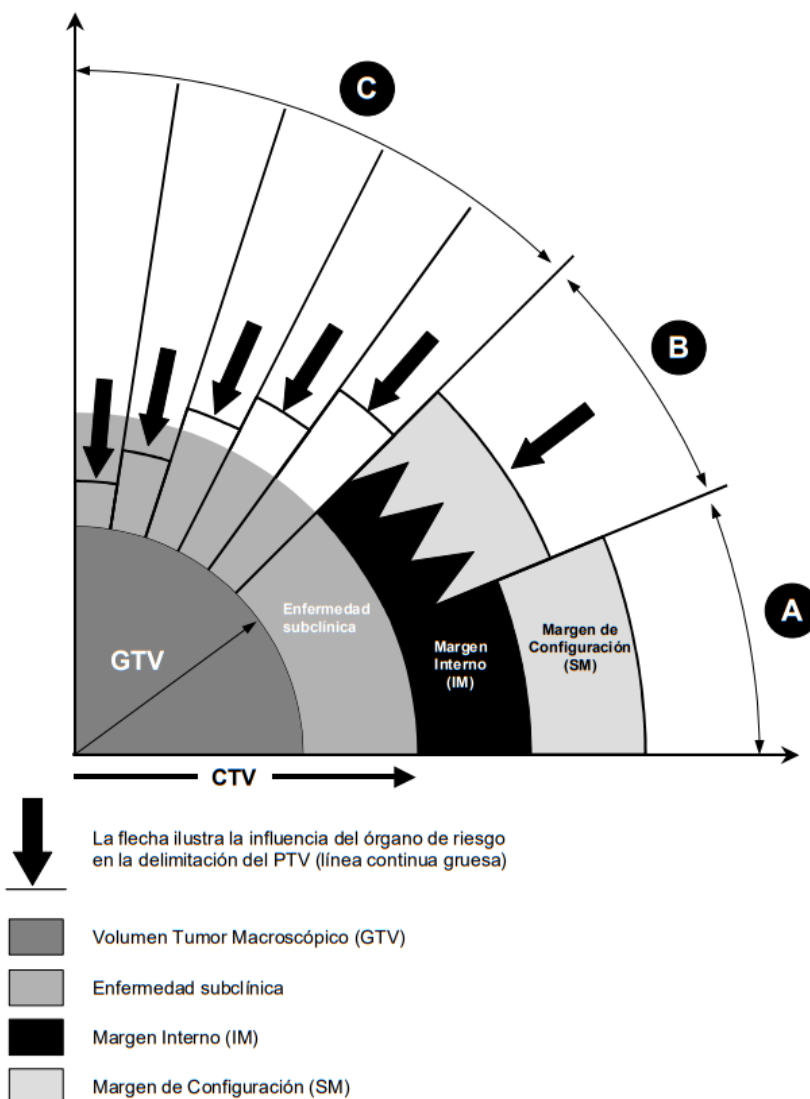


Ilustración 5: Esquema de la relación entre los distintos volúmenes (GTV, CTV y PTV) en tres situaciones clínicas (A, B y C). En A se añade un margen en torno al GTV para tener en cuenta una potencial invasión subclínica, definiendo el CTV. Además, se considera un margen adicional de seguridad por variaciones geométricas e incertidumbres, se añade un IM por las variaciones en posición, forma o tamaño del CTV y un SM para tener en cuenta todas las variaciones e incertidumbres en el posicionado haz-paciente. $CTV + IM + SM$ definen el PTV. En B, como la adición de todas las fuentes de incertidumbre, lleva generalmente a un PTV excesivamente grande, en lugar de añadir los márgenes, se acepta un compromiso y un PTV menor. Y en C, la presencia de OAR (médula espinal, nervio óptico, etc.) reduce dramáticamente el tamaño del margen de seguridad aceptable [18]

7.3.2.1 Volumen Tumor Macroscópico (GTV)

El GTV es un concepto clínico-anatómico, es la localización y extensión demostrable del crecimiento de tejido maligno. Consiste en el tumor primario ("GTV primario") y posiblemente linfadenopatías metastásicas ("GTV nodular") y otras metástasis ("GTV M"). Casi siempre corresponde a aquellas partes donde la densidad de células tumorales es mayor.

La forma, tamaño y posición del GTV pueden determinarse mediante examen clínico (inspección, palpación, endoscopia) o técnicas de imagen (TAC, radiografía, ultrasonografía, RMN y estudios de medicina nuclear como cámaras gamma o PET).

7.3.2.2 Volumen Blanco Clínico (CTV)

El CTV es un volumen de tejido que contiene el GTV. Es un concepto puramente clínico-anatómico y se describe como aquel que incluye, junto al tumor conocido, estructuras con sospecha de diseminación no demostrada clínicamente. Cuando se realiza el examen microscópico de un cáncer, a menudo se encuentran extensiones subclínicas en torno al GTV.

La prescripción se basa entonces en la asunción de que, en algunos órganos o tejidos definibles anatómicamente, podría haber, con algún nivel de probabilidad, células cancerosas; aun cuando éstas sean subclínicas y no puedan detectarse con las técnicas disponibles. La estimación de esta probabilidad se basa en la experiencia clínica proveniente de tratamientos documentados y de su seguimiento. En un paciente particular, pueden existir más de un CTV en los cuales se pueden prescribir dosis diferentes.

7.3.2.3 Volumen Blanco de Planificación (PTV)

El PTV es un concepto geométrico utilizado para seleccionar los tamaños y configuraciones apropiados de los haces, de modo que se asegure que la dosis prescrita es realmente administrada al CTV (la dosis al PTV es representativa de la dosis al CTV). Durante el tratamiento existen variaciones en la posición, tamaño y forma del CTV. Estas son, en su mayoría, de origen fisiológico, por ejemplo, por la respiración, el llenado de la vejiga o el recto, el latido cardíaco o por movimientos intestinales además de las relacionadas al reposicionamiento diario del paciente. Para evitar que todas estas variaciones generen desviaciones significativas respecto a la prescripción de dosis en el CTV deben añadirse márgenes, generando así el PTV.

Si no se añaden estos márgenes, durante parte del tratamiento algunos tejidos pueden desplazarse hacia el interior o el exterior del campo terapéutico, produciendo una sobredosificación o una subdosificación. Como no existe la solución ideal que cumpla todos los requerimientos, a la hora de la planificación se debe buscar un compromiso aceptable, por lo que la delimitación del PTV implica un juicio y la responsabilidad del oncólogo radioterapeuta y del físico médico.

7.3.2.4 Órganos de Riesgo (OAR)

Los Órganos de Riesgo (“estructuras normales críticas”) son tejidos normales o sanos, cercanos al volumen blanco, cuya sensibilidad a la radiación puede influir significativamente en la planificación del tratamiento o en la dosis prescrita.

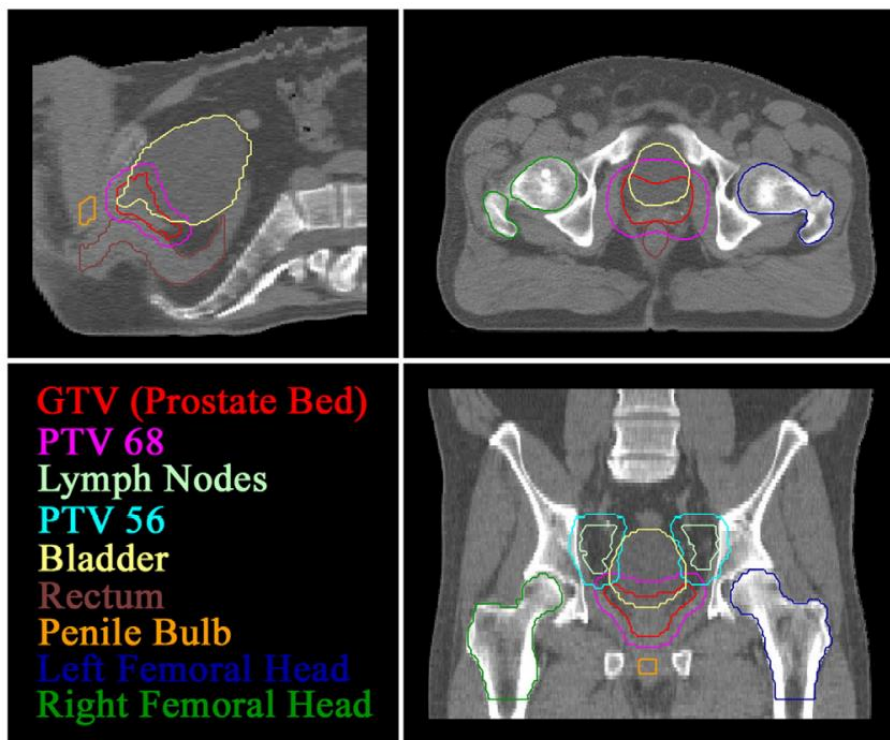


Ilustración 6: Vista de corte sagital, axial y coronal de TAC de abdomen con los contornos de GTV, PTV y los órganos de riesgo (vejiga, recto, cabezas femorales y bulbo peniano) para tratamiento de cáncer de próstata [18]

7.3.3 Simulación

En los últimos 30 años, el desarrollo de tecnologías como la tomografía axial computarizada (TAC) y la resonancia magnética nuclear (RMI) han hecho posible la visualización tridimensional del GTV, así como de los OAR, lo que permitió el desarrollo de la planificación del tratamiento tridimensional y las técnicas de terapia conformada [13]. Además, en los últimos tiempos, se ha extendido el uso de otras tecnologías como el ultrasonido y la tomografía por emisión de positrones (PET).

El objetivo de la obtención de imágenes durante la simulación es proporcionar una visualización de la anatomía del paciente tal como aparecerá durante el tratamiento. La modalidad de imagen más adecuada para una situación clínica determinada depende de las características de los tejidos que se están estudiando. En general, la TAC es la modalidad de imagen primaria para la simulación de RT y forma la base para muchos cálculos de planificación del tratamiento. Por otro lado, la RM es el estándar de oro para la visualización de neoplasias cerebrales y se usa cada vez más en simulación de RT [19].

Las imágenes producidas en la TAC con rayos X proporcionan indirectamente una medida de la densidad electrónica del tejido (Unidad Hounsfield), que se representa mediante una escala de grises. Los datos de densidad son importantes para el proceso de planificación porque se utilizan para predecir y modelar la interacción física de los haces de fotones o electrones en el paciente. Las otras modalidades de imagen aportan a la determinación de la extensión y localización del tumor, porque proporcionan un mejor contraste de tejidos blandos o información metabólica o fisiológica adicional [13].

7.3.3.1 Inmovilización

Como en la mayoría de los casos el objetivo es tratar al paciente con la dosis fraccionada en múltiples sesiones, es necesario inmovilizarlo y posicionarlo de forma reproducible en la simulación. Es por esto que se utilizan ampliamente dispositivos de inmovilización, que a su vez permiten el uso de márgenes más pequeños, reduciendo así la dosis a los tejidos normales circundantes.

Existen una amplia gama de dispositivos auxiliares de inmovilización, los dispositivos de inmovilización de uso común están contruidos con un material de molde de plástico fundido, moldes de espuma solidificada (cuñas) o un dispositivo de molde inflable reutilizable (colchonetas de vacío). Una vez que se ha aprobado y documentado el posicionamiento del paciente, el proceso de planificación del tratamiento continúa con una definición clara de todos los volúmenes de tratamiento y órganos de riesgo, la colocación de los campos de tratamiento y el cálculo de la dosis de radiación [13].

Cabe mencionar que, aunque se busca la mayor rigurosidad posible, hay fuentes de incertidumbres que son difíciles de controlar, como el movimiento de órganos/tumores por la respiración, la función cardíaca, la actividad peristáltica y el llenado y vaciado exacto de órganos [19].

7.3.4 Planificación de tratamiento

La RT se puede considerar como un sistema de subcomponentes: diagnóstico, imagenología, delimitación anatómica, diseño del plan de tratamiento, control de calidad y tratamiento fraccionado, que a su vez incluye otros subcomponentes, como configuración del paciente, guía de imágenes, y entrega de dosis. La delineación o contorno y el diseño del plan de tratamiento, conforman lo que comúnmente se denomina planificación del tratamiento de radiación [18].

Las tareas de planificación del tratamiento de radiación son en gran parte impulsadas por computadora. El primer paso es definir el PTV, es decir, el CTV con consideraciones del movimiento del órgano y el margen de configuración. Para esto primero se contornean las estructuras anatómicas del paciente y con estos contornos se forma base para el diseño del plan de tratamiento. Usando prescripciones de dosis para los PTV y restricciones de dosis para los OAR definidas por el médico, se diseña una disposición de fuentes de radiación que cumpla los objetivos.

En la EBRT, se pueden manipular parámetros como el número, las modalidades (es decir, fotones, electrones, etc.) y las geometrías de los haces, además de los parámetros por haz, como la energía, la modulación y la intensidad del haz o unidades monitor. Los sistemas de planificación de tratamiento (TPS) modernos automatizan muchos de estos parámetros. El TPS estima la distribución de dosis y los resultados que genera se revisan utilizando información de dosis tridimensional y análisis clínicos como el histograma dosis-volumen (DVH). Por lo tanto, la planificación del tratamiento es realizada por un sistema compuesto por el físico médico o planificador, el TPS y el médico que finalmente aprobará el plan.

Los planes para IMRT o VMAT se logran con softwares avanzados capaces de resolver problemas de optimización inversa, que tienen como objetivo determinar los parámetros óptimos del equipo a partir de un conjunto de objetivos de dosis especificados a priori para el PTV y los OAR, para proporcionar así las distribuciones de dosis planificadas. Un buen software es capaz de proporcionar una solución en minutos. Sin embargo, un enfoque de planificación inversa normalmente exige una gran cantidad de esfuerzo manual y habilidades considerables para generar un plan de tratamiento de alta calidad. El físico médico todavía debe pasar por muchas iteraciones para ajustar y afinar manualmente los parámetros de planificación del tratamiento hasta que se logra una distribución de dosis aceptable, además de recibir comentarios del médico muchas veces antes de que se apruebe el plan. Esto da como resultado un proceso lento (de varias horas, incluso días), que implica una variabilidad en la calidad del plan que depende de factores como el tiempo disponible para generar el plan, las directrices de la institución o las habilidades del planificador. Además, el tiempo prolongado de planificación dificulta la implementación de estrategias adaptativas y puede retrasar la administración del tratamiento, lo que tiene un impacto negativo en el control del tumor y la calidad de vida de los pacientes [20].

Para reducir más la complejidad de la planificación y mejorar la eficiencia de la planificación del tratamiento, se han investigado enfoques de planificación del tratamiento basados en datos que utilizan el conocimiento de casos anteriores para la predicción de las distribuciones de dosis y las restricciones de un nuevo caso. Este concepto fue utilizado por investigadores hace más de una década en forma de planificación basada en el conocimiento (KBP) y también se han estado implementando clínicamente [3].

7.4 Knowledge-Based Planning (KBP)

La KBP se basa en desarrollar motores inteligentes que, después de extraer los datos históricos de planificación de una biblioteca de planes de pacientes anteriores, puedan predecir los histogramas dosis-volumen (DVH) para cualquier OAR de un paciente nuevo [21]. El objetivo es crear planes basados en un compilado cada vez mayor de conocimientos de planificación, así como aumentar la optimización de plan para ahorrar tiempo y mantener planes de tratamiento de alta calidad entre planificadores de diferentes experiencias y niveles de habilidad [4].

Los métodos tradicionales de KBP incluyen métodos basados en atlas, modelado estadístico y Machine Learning (ML). El software comercial RapidPlan™ es un ejemplo del módulo KBP basado en un método tradicional, que fue lanzado en 2014 por Varian Medical Systems. Este software estima los histogramas dosis-volumen (DVH, ilustración 7) y genera objetivos de optimización para un nuevo plan [3].

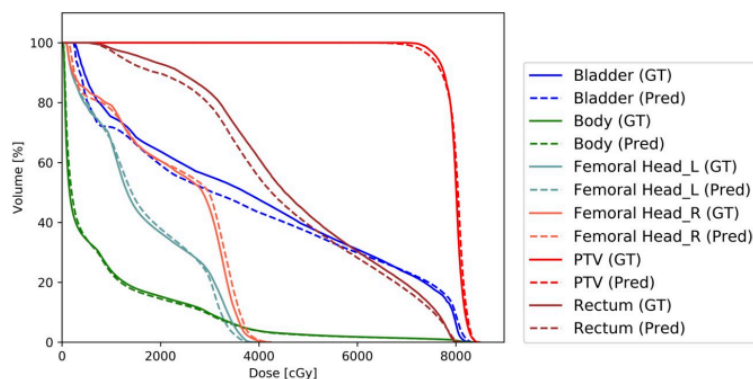


Ilustración 7: Histograma Dosis-Volumen relativo acumulado de una planificación para tratamiento de cáncer de próstata y una predicción con una CNN [22]

En general, los métodos tradicionales de KBP utilizan características geométricas, es decir, la distancia de los OAR al PTV y el histograma de volumen de superposición OAR (OVH), ya sea para encontrar los casos anteriores que mejor coinciden de un repositorio o para construir modelos de predicción de dosis, es decir, ML, modelo estadístico (ML sigue un marco de trabajo similar a los métodos KBP basados en atlas, en términos de entradas: características geométricas y anatómicas hechas a mano, y salidas: métricas DVH).

En los enfoques basados en atlas, las características físicas (OVH, proyecciones de la vista del ojo del haz, ubicación del tumor) se identifican primero para determinar la similitud con los planes clínicos anteriores. A esto le sigue la transferencia de conocimientos (es decir, restricciones de dosis, valores de DVH, parámetros geométricos del haz, etc.) para predecir los DVH alcanzables de un nuevo caso o para proporcionar un mejor punto de partida a la planificación por prueba y error. Dentro de los métodos basados en atlas, un enfoque indirecto selecciona los casos coincidentes en función de los parámetros dosimétricos predichos a través de los modelos. Por el contrario, un enfoque directo compara la similitud entre el paciente antiguo y el nuevo en función del DVH, las imágenes de TC o las proyecciones de estructuras de la vista desde el ojo del haz (BEV) y adopta los parámetros de planificación de los mejores casos de coincidencia [3].

7.4.1 Características

Un tema común en los métodos KBP tradicionales es que la relación geométrica del objetivo con respecto a las estructuras críticas cercanas presenta una fuerte correlación con las métricas de calidad del plan alcanzables. Varias características geométricas, como el histograma de volumen superpuesto (overlap volume histogram, OVH), histogramas de distancia al objetivo (distance to target histograms, DTH) y distancia OAR a PTV se han correlacionado con la dosis.

7.4.1.1 Métodos basados en OVH

El histograma de volumen superpuesto es una característica común a los enfoques basados en modelos y en atlas para la predicción de dosis. La expansión del objetivo ocurre hasta que el OAR se superponga por completo con el objetivo, y la contracción se repite hasta que no haya superposición entre el objetivo y el OAR. El OVH resultante describe el porcentaje del volumen OAR que se superpone con un objetivo uniformemente expandido o contraído. En general, los modelos impulsados por OVH asumen que la dosis a un OAR es inversamente proporcional a la distancia desde el objetivo.

7.4.1.2 Métodos basados en proyecciones

Los algoritmos basados en proyección suelen utilizar la perspectiva de la Beam's eye View (BEV) y análisis estadístico para evaluar la similitud entre un caso nuevo y casos anteriores. Se puede identificar el caso anterior que mejor se ajuste en función de la suma de los valores de información calculados desde cada perspectiva de BEV en todos los haces de un plan.

7.4.1.3 Métodos basados en DTH

El DTH traza el volumen fraccional de un OAR dentro de una cierta distancia de la superficie del PTV. Además de los volúmenes de PTV y OAR, la métrica DTH también se usa como una característica de entrada en los enfoques de ML para KBP. Las técnicas de ML incluyen

regresión no lineal multivariable (MVNLR) y regresión de vector de soporte (SVR). DTH es equivalente a OVH cuando la distancia es euclidiana. El DTH también se ha utilizado con modelos basados en regresión multivariable, disponible comercialmente como RapidPlan™ en el software de planificación de tratamientos Eclipse® (Varian Medical Systems, Palo Alto, CA) [3].

7.4.2 Problemática

Todas las alternativas de planificación automática han sido probadas durante estos años en diferentes poblaciones de pacientes y zonas anatómicas, y han logrado agilizar considerablemente el proceso de planificación, reduciendo en tiempo en un 70-90%, tanto para IMRT como para VMAT. También se ha demostrado que estos métodos KBP mejoran la calidad, la consistencia y la eficiencia del plan. Pero, incluso con estos avances, los métodos KBP todavía tienen dos inconvenientes principales. En primer lugar, utilizan objetivos de volumen de dosis, ya sea de dimensión cero (puntos de volumen de dosis) o unidimensionales (histograma dosis-volumen, DVH), para las estructuras delineadas. Estos objetivos son insensibles a las variaciones espaciales de la dosis dentro de las estructuras demarcadas y ciegos a aquellas estructuras que no están demarcadas. Como resultado, los planes finales con objetivos de DVH aceptables aún pueden contener distribuciones de dosis inaceptables, y requiere pasos de procesamiento posterior en los que el físico médico agrega manualmente contornos auxiliares de ayuda a la planificación y vuelve a optimizar para controlar estas características espaciales. En segundo lugar, estos métodos generalmente se basan en la extracción de características artesanales, como el OVH y la distancia al histograma objetivo. Estas características artesanales en el plan del paciente no cubren todas las características inherentes de la estructura, causando que la calidad de las distribuciones de dosis, la predicción no se puede mejorar fácilmente [20].

Por estas razones, se está buscando desarrollar un algoritmo que pueda extraer automáticamente las características de las estructuras contorneadas del paciente con el fin de lograr una predicción más precisa y efectiva de las distribuciones de dosis en 3D. A esta necesidad se le agrega el advenimiento del hardware informático avanzado y las herramientas de aprendizaje profundo (DL) que han traído avances en la informática médica en los últimos años.

7.5 Deep Learning (DL)

En general, los algoritmos clásicos de ML funcionan bien en una amplia variedad de problemas, sin embargo, no han logrado resolver los problemas centrales de la IA, como el reconocimiento del habla o de objetos. El desarrollo del aprendizaje profundo o Deep Learning está motivado por la búsqueda de la solución a estas problemáticas.

Muchos problemas de ML se vuelven extremadamente difíciles y costosos computacionalmente cuando la cantidad de dimensiones en los datos es alta. Esto se debe a que, para generalizar bien, los algoritmos de ML deben guiarse por creencias previas sobre qué tipo de función deben aprender y hacer suposiciones sólidas y específicas de la tarea. Las tareas de AI (Inteligencia artificial) tienen una estructura que es demasiado compleja para limitarse a propiedades simples especificadas manualmente, por lo que se buscan algoritmos de aprendizaje que incorporen suposiciones de propósito más general. Es por esto que, en las redes neuronales, no se incluyen suposiciones tan fuertes y específicas de la tarea, para que puedan generalizarse a una variedad mucho más amplia de estructuras. La idea central en el DL es que los datos fueron generados por la composición de factores o características en múltiples niveles en una jerarquía.

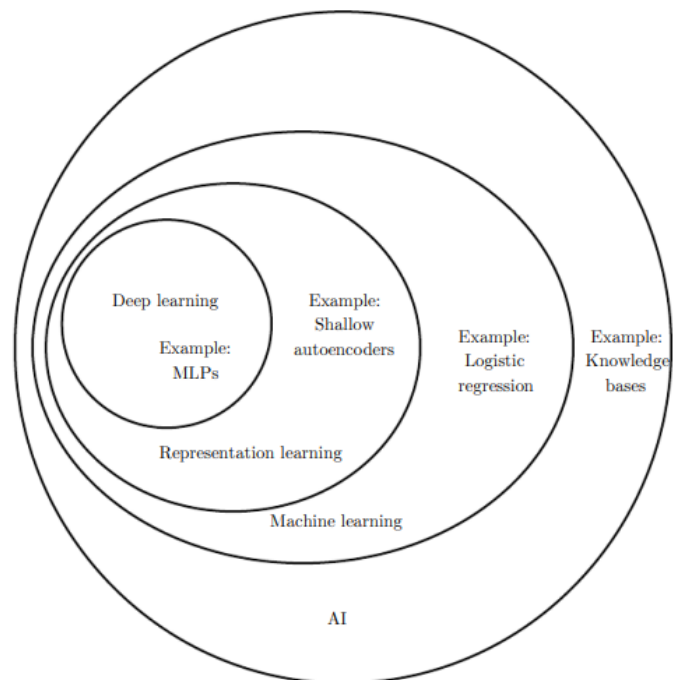


Ilustración 8: Diagrama de Venn que muestra cómo el Deep Learning es un tipo de Machine Learning, que se utiliza para muchos de los enfoques de la IA [23]

Las Deep feedforward networks o perceptrones multicapa (MLP), son los modelos de DL por excelencia, cuyo objetivo es aproximar el problema a alguna función F . Estos modelos se denominan feedforward porque la información fluye a través de la función que se evalúa desde x , a través de los cálculos o capas intermedias utilizados para definir F , y finalmente a la salida y . El algoritmo de aprendizaje debe decidir cómo usar las capas internas para producir el resultado deseado, pero los datos de entrenamiento no dicen qué debe hacer cada capa individual. Debido a que los datos de entrenamiento no muestran el resultado deseado para cada una de estas capas, estas capas se denominan capas ocultas.

Finalmente, estas redes se denominan neuronales porque están vagamente inspiradas en la neurociencia. Cada capa oculta de la red suele tener un valor vectorial. Puede interpretarse que cada elemento del vector desempeña un papel análogo al de una neurona, en el sentido de que recibe información de muchas otras unidades y calcula su propio valor de activación. Una unidad/nodo/neurona es una función que toma como entrada un vector x y produce un escalar (Ilustración 9), además está parametrizada por un vector de peso w y un término de sesgo (bias) denotado por b . Además, se le aplica la función f que es la función de activación. La salida de la unidad se puede describir como $f(\sum_{i=1}^n x_i \cdot w_i + b)$.

La idea de usar muchas capas de representación con valores vectoriales se extrae de la neurociencia. Sin embargo, la investigación moderna de redes neuronales está guiada por muchas disciplinas matemáticas y de ingeniería, y el objetivo de las redes neuronales no es modelar perfectamente el cerebro [23].

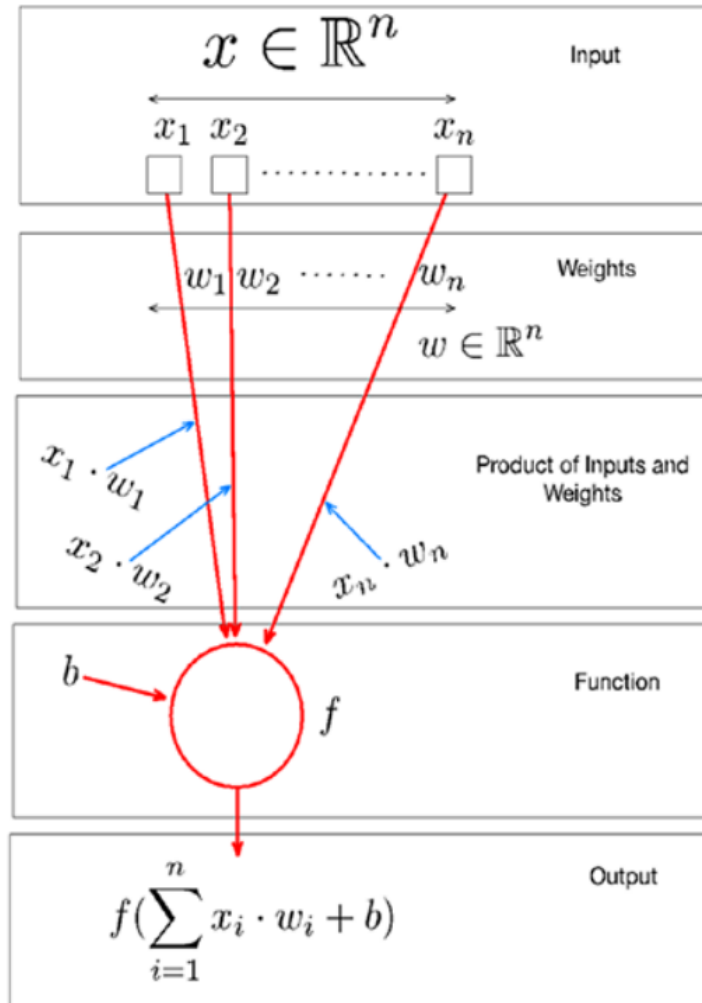


Ilustración 9: Neurona/nodo/unidad unitaria [24]

La forma más simple de una red neuronal consta de tres capas: una capa de entrada, la capa oculta y la capa de salida. Entonces, las neuronas dentro de cada capa son nodos que están conectados a nodos subsiguientes a través de enlaces que corresponden a conexiones biológicas axón-sinapsis-dendrita. Las redes profundas constan de múltiples capas ocultas además de la capa entrada y la capa de salida (Ilustración 10). En general, el número de capas determina la profundidad de la red y el número de neuronas determina su ancho. Estos modelos pueden aprender características más profundas al filtrar la información a través de las múltiples capas ocultas.

Las neuronas de entrada se activan a través de sensores que perciben el entorno, las siguientes neuronas se activan a través de conexiones ponderadas de neuronas previamente activas. El aprendizaje se trata de encontrar el peso para cada parámetro de cada neurona que haga que el modelo exhibe el comportamiento deseado. Dependiendo del problema y de cómo estén conectadas las neuronas, tal comportamiento puede requerir largas cadenas causales de etapas computacionales, donde cada etapa transforma de manera no lineal la activación agregada de la red [25].

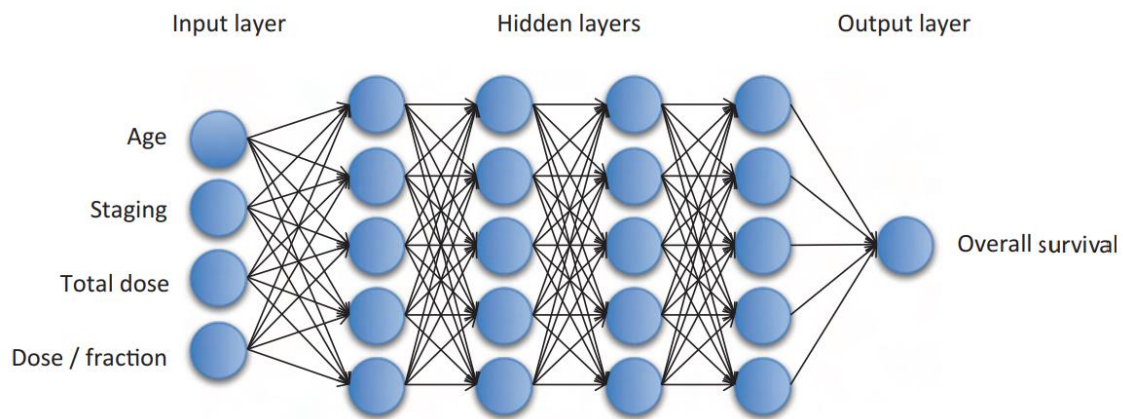


Ilustración 10: Ejemplo de una red neuronal. Fuente: *Big Data in Radiation Oncology* [5]

El principal desafío es tener un buen desempeño del modelo con entradas nuevas y desconocidas, no sólo con aquellas con las que se lo entrena. La capacidad del modelo de desempeñarse bien con entradas no observadas previamente se denomina generalización. Para entrenar un modelo, se genera un conjunto de entrenamiento, del cual se puede calcular alguna medida de error y tomar estrategias para reducirlo, es decir, como en un simple problema de optimización. Lo que separa el aprendizaje automático de la optimización es que se quiere que el error de generalización también sea bajo. Se estima el error de generalización de un modelo de aprendizaje automático midiendo su rendimiento en un conjunto de pruebas de ejemplos.

Entonces cuando se utiliza un algoritmo de DL, no se fijan los parámetros con anticipación, si no que, se toman muestras del conjunto de entrenamiento, que luego se usan para elegir los parámetros de forma automática y así reducir el error del conjunto de entrenamiento y recién evaluar con el conjunto de prueba. Bajo este proceso, el error del conjunto de prueba siempre será mayor o como mucho igual al valor del error de entrenamiento [24].

7.5.1 Conceptos básicos

7.5.1.1 Descenso por el gradiente

Los algoritmos de DL implican algún tipo de optimización, que generalmente se refiere a la tarea de minimizar alguna función $L(x)$ alterando x . A esta función L que se minimiza se la llama función de costo, función de pérdida o función de error.

Suponiendo una función $y = f(x)$, donde x e y son números reales. A la derivada de esta función se la denomina como $f'(x)$. La $f'(x)$ da la pendiente de la función en el punto x . Entonces, la derivada es útil para minimizar una función porque indica para qué dirección cambiar x para disminuir el valor de y . Es decir, se puede reducir $f(x)$ moviendo x en pequeños pasos con el signo opuesto de la derivada. Esta técnica se denomina descenso de gradiente. Cuando $f'(x) = 0$, la derivada no proporciona información sobre en qué dirección moverse, es decir que se encuentra en un mínimo o máximo local o en un punto silla (Ilustración 11). Este algoritmo busca llegar a un mínimo, es decir, un punto donde $f(x)$ es menor que en todos los puntos vecinos, por lo que ya no es posible disminuir $f(x)$.

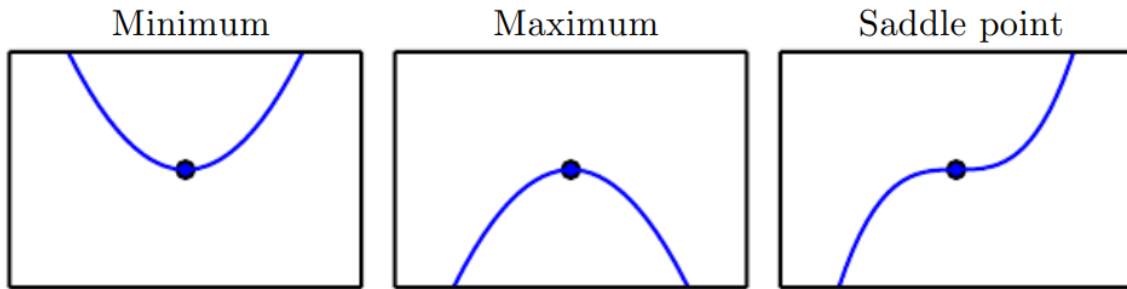


Ilustración 11: Ejemplos de valores de $f(x)$ donde $f'(x) = 0$ [23]

Un punto que obtiene el valor absoluto más bajo de $f(x)$ es un mínimo global. Es posible que haya solo un mínimo global o múltiples mínimos globales de la función. Lo más habitual es que existan mínimos locales que no sean globalmente óptimos. En DL, se optimizan funciones que pueden tener muchos mínimos locales que no son óptimos. Esto hace que la optimización sea muy difícil, especialmente porque la entrada a la función es multidimensional. Por lo tanto, generalmente se acepta encontrar un valor de y que sea muy bajo, pero no necesariamente mínimo. Es posible que no se garantice que el algoritmo de optimización llegue incluso a un mínimo local en un período de tiempo razonable, pero a menudo encuentra un valor muy bajo de la función de costo lo suficientemente rápido como para ser útil [23].

7.5.1.2 Underfitting y Overfitting

El underfitting o ajuste insuficiente ocurre cuando el modelo no es capaz de obtener un valor de error suficientemente bajo para el conjunto de entrenamiento. Por otro lado, el overfitting o sobreajuste ocurre cuando la brecha entre el error de entrenamiento y el error de prueba es demasiado grande, es decir, el modelo se especializó demasiado para el conjunto de entrenamiento y es incapaz de generalizar para cualquier otro conjunto.

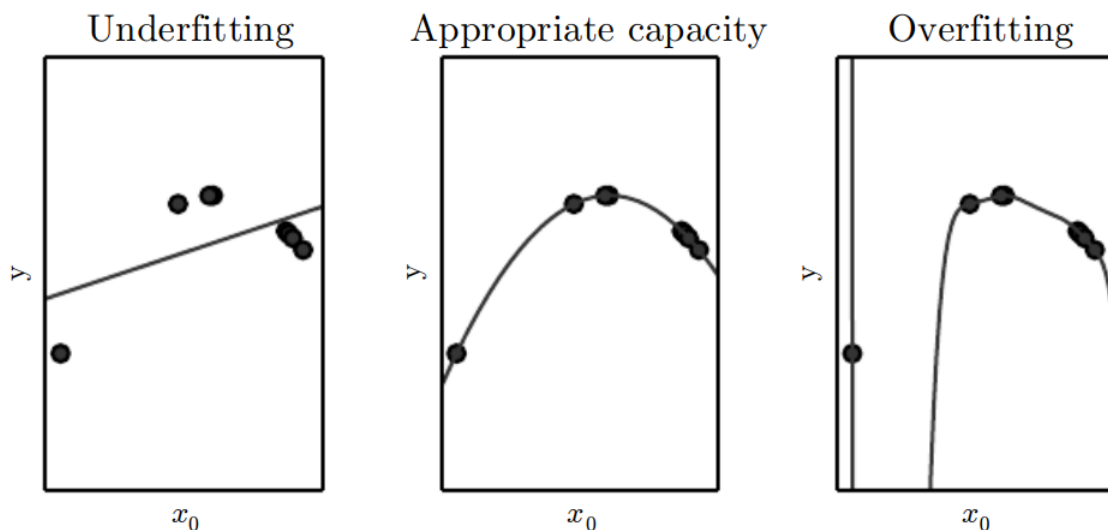


Ilustración 12: Ejemplos de situaciones de ajuste insuficiente, ajuste apropiado y sobreajuste [23]

Entonces el sobreajuste ocurre cuando durante el entrenamiento el modelo se ajusta de manera cercana o incluso exacta al conjunto de datos de entrenamiento a expensas de la capacidad de generalizar, ya que no es capaz de abarcar los patrones más generales de los datos de prueba. Se han utilizado diferentes enfoques, como abandono o aumento de datos, para prevenir el sobreajuste del modelo.

7.5.1.3 Minilotes o minibatches

La mayoría de los algoritmos de optimización convergen mucho más rápido si se les permite calcular rápidamente estimaciones aproximadas del gradiente en lugar de calcular el gradiente exacto. Calcular exactamente la optimización de la función de pérdida es muy costoso computacionalmente, porque demanda evaluar el modelo con cada dato del dataset completo. Es por esto que en la práctica comúnmente se implementa el cálculo tomando muestras al azar de una pequeña cantidad de ejemplos del conjunto de datos y luego tomando el promedio solo de esos ejemplos, estas pequeñas muestras son llamadas minilotes o minibatches. Otra consideración que motiva la estimación estadística del gradiente es la redundancia en el conjunto de entrenamiento. En el peor de los casos, todas las muestras del conjunto m de entrenamiento son copias idénticas entre sí y una estimación podría calcular el gradiente correcto con una sola muestra, utilizando m veces menos cálculos. Esto en la práctica es poco probable, pero se puede llegar a encontrar datasets con una gran cantidad de ejemplos que hacen contribuciones muy similares al gradiente (redundantes). Los algoritmos de optimización que usan todo el conjunto de entrenamiento se denominan métodos de gradiente determinista o por lotes, porque procesan todos los ejemplos de entrenamiento simultáneamente de un gran lote. Los algoritmos de optimización que usan solo un grupo pequeño de ejemplos a la vez a veces se denominan métodos estocásticos [23].

7.5.1.4 Parámetros e hiperparámetros

En los algoritmos de DL se pueden variar algunas configuraciones para controlar su comportamiento en el aprendizaje. Esto se logra ajustando valores llamados hiperparámetros, que por ejemplo son el tamaño del minibatch, la tasa de aprendizaje o el momentum. Los valores de los hiperparámetros no son adaptados por el propio algoritmo de aprendizaje dado que no es apropiado aprender un hiperparámetro en el conjunto de entrenamiento porque siempre elegirán la máxima capacidad posible del modelo, lo que provocará un sobreajuste. Por lo general, cuando se quiere generar un algoritmo para setear automáticamente los hiperparámetros se genera el conjunto de validación a partir de los datos de entrenamiento. Entonces se generan dos subconjuntos: uno se utiliza para entrenar y aprender los parámetros y el otro para estimar el error de generalización durante o después del entrenamiento, lo que permite actualizar los hiperparámetros. Por lo general, se usa alrededor del 80% de los datos de entrenamiento para el entrenamiento y el 20% para la validación.

7.5.1.5 Descenso por el gradiente estocástico (SGD)

Prácticamente todo el DL está impulsado por el algoritmo de descenso de gradiente estocástico o SGD, que es una extensión del algoritmo de descenso por el gradiente. Un problema del aprendizaje automático es que se necesitan grandes conjuntos de entrenamiento para una buena generalización, pero como se mencionaba, los grandes conjuntos de entrenamiento también son más costosos, ya que exigen un mayor número de cálculos en cada paso del descenso. Entonces, la idea del descenso de gradiente estocástico es que el gradiente se puede estimar aproximadamente utilizando un pequeño conjunto de muestras extraído uniformemente del conjunto de entrenamiento. El tamaño de minibatch generalmente se elige para ser un número relativamente pequeño de ejemplos, que van desde 1 hasta unos pocos cientos [23].

7.5.1.6 Retropropagación

Cuando se usa una red neuronal para aceptar una entrada y producir una salida, la información fluye hacia adelante a través de la red. Las entradas proporcionan la información inicial que luego se propaga hasta las neuronas ocultas en cada capa y finalmente produce la salida. Esto es llamado propagación directa. El algoritmo de retropropagación permite que la información del costo fluya hacia atrás a través de la red para calcular el gradiente y así, permite evaluar numéricamente una expresión analítica para el gradiente de derivadas con un procedimiento simple y económico computacionalmente. En realidad, la propagación hacia atrás se refiere sólo al método para calcular las derivadas. Muchas tareas de ML implican calcular otras derivadas, ya sea como parte del proceso de aprendizaje o para analizar el modelo aprendido. El algoritmo de retropropagación también se puede aplicar a estas tareas y no se limita a calcular el gradiente de la función de costo con respecto a los parámetros [23].

7.5.1.7 Momentum

Aunque el descenso de gradiente estocástico sigue siendo una estrategia de optimización muy usada, puede llegar a ser bastante lenta. El método del impulso o momentum está diseñado para acelerar el aprendizaje. El algoritmo acumula un promedio móvil exponencialmente decreciente de gradientes pasados y continúa moviéndose en su dirección.

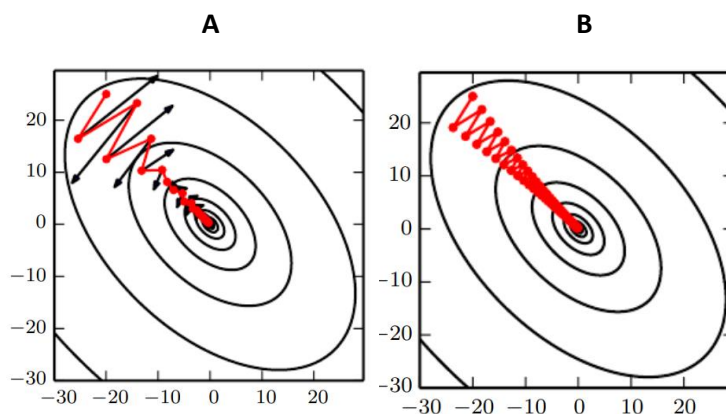


Ilustración 13: Representación de la acción del momentum. Las líneas de contorno representan una función de pérdida cuadrática condicionada. En la figura A, el camino rojo que atraviesa los contornos indica el camino seguido por la regla de aprendizaje con momentum. En cada paso del camino, una flecha indica el paso que daría el descenso de gradiente en ese punto, perdiendo tiempo moviéndose hacia adelante y hacia atrás a través del eje angosto de cada contorno como se ve en la figura B [23]

Formalmente, el algoritmo utiliza una variable v que simula la velocidad y el momentum es la dirección y la velocidad a la que se mueven los parámetros a través del espacio multidimensional de parámetros. La velocidad se establece en un promedio exponencialmente decreciente del gradiente negativo. Entonces antes, el tamaño del paso era simplemente la norma del gradiente multiplicado por la tasa de aprendizaje. Ahora, el tamaño del paso depende del tamaño y la alineación de una secuencia de gradientes anteriores. Así se consigue que el tamaño del paso sea mayor cuando muchos gradientes sucesivos apuntan exactamente en la misma dirección. El hiperparámetro α que toma valores $[0, 1)$ determina qué tan rápido decaen exponencialmente las contribuciones de los gradientes anteriores. Los valores de α usados normalmente en la práctica incluyen .5, .9 y .99. Al igual que la tasa de aprendizaje, α también puede adaptarse con el tiempo. Por lo general, se comienza con un valor pequeño y luego se eleva [23].

7.5.1.8 Tasa de aprendizaje o Learning Rate (LR)

El Learning Rate es un valor escalar positivo que determina el tamaño del paso y generalmente se establece el LR como una constante pequeña. Los algoritmos más avanzados adaptan sus tasas de aprendizaje durante el entrenamiento o aprovechan la información contenida en las segundas derivadas de la función de costo.

7.5.1.9 Funciones de optimización con LR adaptativo

Como en la práctica la función de pérdida es muy sensible en algunas direcciones en el espacio de parámetros e insensible a otras, es necesario disminuir gradualmente la tasa de aprendizaje con el tiempo.

7.5.1.9.1 AdaGrad

El algoritmo AdaGrad se basa en un mecanismo de tasa de aprendizaje adaptativo que asigna una tasa de aprendizaje más alta a los parámetros que se han actualizado de forma más suave y una tasa de aprendizaje más baja a los parámetros que se han actualizado drásticamente. Adapta individualmente las tasas de aprendizaje de todos los parámetros del modelo, mediante el escalamiento inversamente proporcional a la raíz cuadrada de la suma de todos sus valores cuadrados históricos. El efecto neto es un mayor progreso en las direcciones de pendiente más suave del espacio de parámetros. Sin embargo, se ha encontrado empíricamente que, para entrenar modelos de redes neuronales profundas, la acumulación de gradientes cuadrados desde el comienzo del entrenamiento puede resultar en una disminución prematura y excesiva en la tasa de aprendizaje efectiva. Por lo que esta función es útil en algunos modelos de aprendizaje profundo, pero no en todos [26].

7.5.1.9.2 ADAM

Adam significa Adaptive Moment Estimation. Es como una combinación de método de impulso y método de AdaGrad, pero cada componente se vuelve a ponderar en el paso de tiempo t .

7.5.1.10 Función de activación

Las funciones de activación se aplican al resultado de cada capa. En teoría, cuando una función de activación no es lineal, una red neuronal puede aproximarse a cualquier función (dado un número suficiente de unidades en la capa oculta). Por lo tanto, siempre se usan funciones de activación no lineales para resolver problemas dentro del ámbito del aprendizaje profundo. Son funciones continuamente diferenciables, que permiten calcular gradientes y usar funciones de optimización para encontrar los parámetros que minimizan la función de pérdida. Si una función no es continuamente diferenciable, los métodos basados en gradientes no avanzarían en el entrenamiento de una red. Además, por evidencia empírica se prefieren las funciones suaves y que se comporten como funciones de identidad cerca del origen.

Las diferentes funciones de activación reportadas en la literatura son Sigmoid, Tanh, SWISH, ReLU y sus variantes, incluyendo Leaky-ReLU, ReLU paramétrica (PReLU). ReLU y sus variantes generalmente se prefieren como funciones de activación debido a su capacidad para superar el problema del gradiente de fuga [24].

7.5.1.10.1 ReLU

La unidad lineal rectificada transforma la entrada como $f(x) = \max(0, x)$. Aunque parece una unidad lineal, tiene una función derivada, por lo tanto, permite calcular el gradiente de las pérdidas. Se usa más comúnmente como una unidad oculta y los resultados muestran que conduce a gradientes grandes y consistentes, lo que ayuda al aprendizaje.

En la mayoría de los casos, una ReLU puede ser una opción predeterminada que conduciría a resultados deseables de manera oportuna. Sin embargo, cuando las entradas se acercan a cero, el gradiente de la función se vuelve cero y, por lo tanto, se atasca dentro de los pasos de entrenamiento sin progreso en el entrenamiento. Esto se conoce comúnmente como el problema de “dying ReLU” [24].

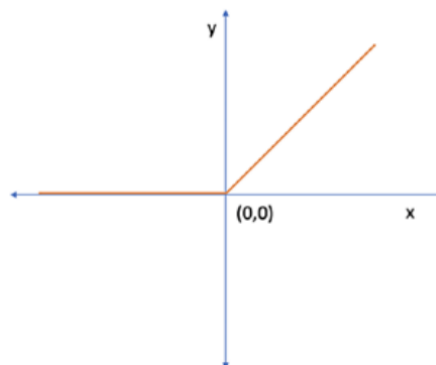


Ilustración 14: Función ReLU [24]

7.5.1.10.2 Sigmoide

La activación sigmoidea transforma la entrada de la siguiente manera: $y = \frac{1}{1+e^{-x}}$. Las unidades sigmoideas se pueden usar en la capa de salida junto con la función entropía cruzada binaria (una función de pérdida ampliamente usada en problemas de segmentación semántica binaria). La salida de esta unidad puede modelar una distribución de Bernoulli sobre la salida y condicionada sobre x.

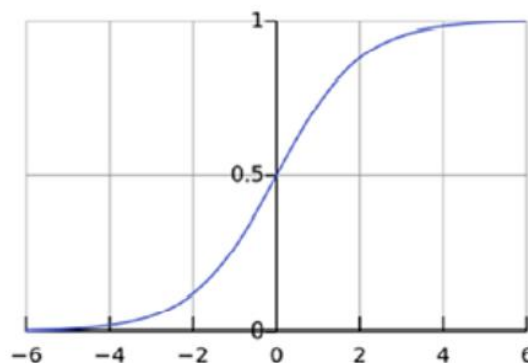


Ilustración 15: Función sigmoidea [24]

7.5.1.10.3 Softmax

La función Softmax es muy similar a la sigmoide, generalmente se usa solo dentro de la capa de salida para tareas de clasificación múltiple junto con la función de pérdida de entropía cruzada. La capa Softmax normaliza las salidas de la capa anterior para que todas las posibilidades de clasificación sumen uno. Por lo general, las unidades de la capa anterior

modelan una puntuación no normalizada de la probabilidad de que la entrada pertenezca a una clase en particular. La capa softmax normaliza esto para que la salida represente la probabilidad de cada clase.

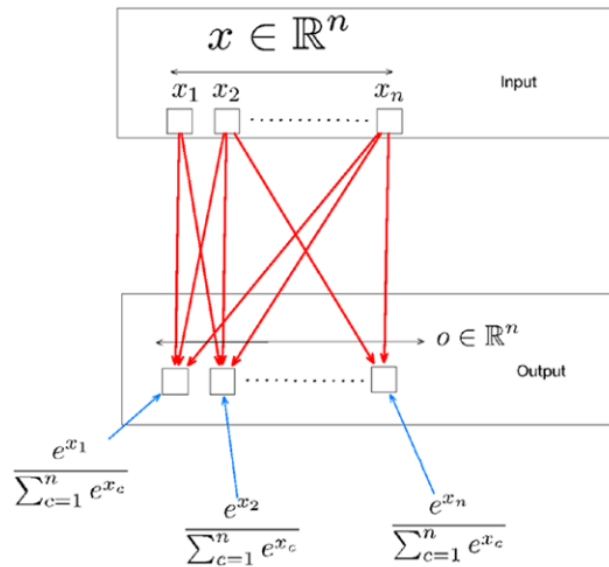


Ilustración 16: Capa de salida con Softmax [24]

7.5.1.11 Función de pérdida

La función de pérdida, en su forma más simple, es la diferencia entre la salida prevista y la salida objetivo. Se presenta una función de pérdida en forma de una función objetivo que se debe minimizar durante el paso de retropropagación para mejorar el rendimiento de la red.

7.5.1.11.1 MSE Loss

Crea un criterio que mide el error cuadrático medio (norma L2 cuadrada) entre cada elemento de la entrada X y objetivo y : $MSE = \frac{1}{n} \sum_{i=1}^n (X_i - y_i)^2$.

7.5.1.11.2 Sharp Loss

Generalmente en imágenes médicas, especialmente en radioterapia, la región de interés clínico representa solo una pequeña parte de la imagen total. Este desequilibrio de datos conduce a una disminución en la precisión de un modelo de predicción de dosis. Buscando solucionar dicho inconveniente se desarrolló la función de pérdida llamada Sharp Loss.

La función Sharp Loss es una variante escalada dinámicamente que deriva de la pérdida clásica del error cuadrático medio: $MSE = \frac{1}{n} \sum_{i=1}^n (D_{pi} - D_i)^2$ donde n es el número total de vóxeles, D_{pi} es la dosis predicha para el vóxel i y D_i es la dosis esperada para el vóxel i .

Para redirigir el proceso de optimización de la red y así, centrarse en los vóxeles más relevantes de dosis más altas, se emplea la función sigmoidea para reducir el peso de los vóxeles de dosis baja en la pérdida de la red. Esta función se modificó en dos aspectos. En primer lugar, se permitió que el factor γ excediera el valor unitario para aumentar el gradiente de pérdida, especialmente para valores de dosis bajos. En segundo lugar, la curva se movió hacia la dirección positiva del eje x , restando $0,03$ de D_i para mantener positivos los valores reales. La función

sigmoidea modificada se propuso como factor modulador de la pérdida de MSE y así crear la función Sharp Loss que se define como: $SharpLoss = \frac{1}{n} \sum_{i=1}^n \frac{1}{1+e^{-(Di-0.03)*\gamma}} (Dpi - Di)^2$.

En el estudio realizado, esta función logró resultados de predicción de dosis superiores en comparación con los de la pérdida MSE. Con diferentes valores del parámetro no negativo γ , el valor $\gamma=100$ presentó el mejor resultado de predicción [27].

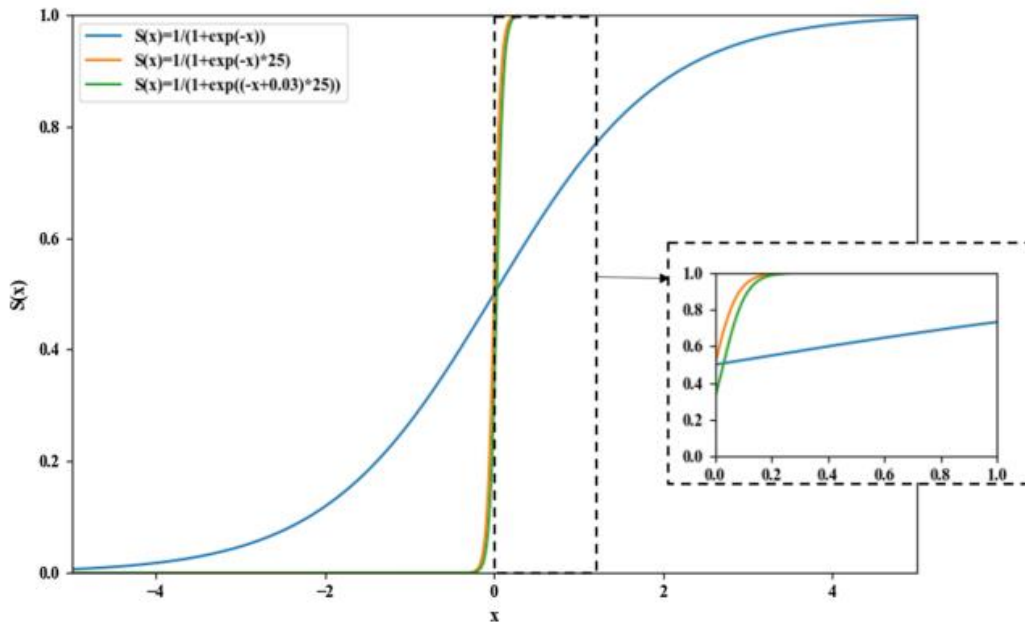


Ilustración 17: Gráfico de las funciones, en azul la función sigmoidea original, en naranja la función con $\gamma = 25$ y en verde la función con $\gamma = 25$ y corrida 0.03 a la derecha [27]

7.5.1.12 Dropout

Mientras que una red neuronal típica tiene todos los nodos o neuronas activados durante todo el entrenamiento, una capa de dropout o abandono omite aleatoriamente una combinación de ciertos nodos junto con sus conexiones cada vez que se actualiza el gradiente, lo que evita que la red se sobreajuste o adapte en exceso a los datos de entrenamiento.

Una capa de dropout proporciona un método computacionalmente económico pero poderoso para regularizar los modelos. Específicamente, con el abandono se entrena al conjunto que consta de todas las subredes que se pueden formar eliminando neuronas que no son de salida de una red base, como se muestra en la ilustración 18.

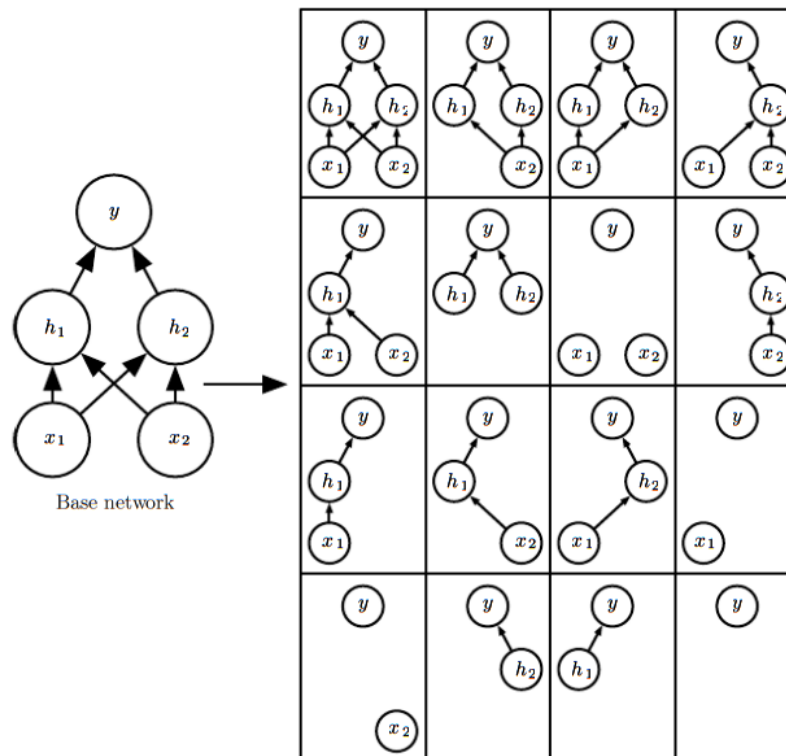


Ilustración 18: Ejemplo de las variantes de una red que se entrenan cuando se le aplica una capa de dropout. [23]

En la mayoría de las redes neuronales se puede eliminar de forma eficaz un nodo multiplicando su valor de salida por cero. Específicamente, para entrenar con abandono, se tiene un algoritmo de aprendizaje basado en minilotes que realiza pequeños pasos en el descenso por el gradiente estocástico. La probabilidad de eliminar un nodo es un hiperparámetro que se fija antes de que comience el entrenamiento. Por lo general, se setea las neuronas de entrada con una probabilidad de 0,8 y las ocultas con una probabilidad de 0,5 [23].

7.5.1.13 Data augmentation

Los modelos DL llegan a menores errores y mejores generalizaciones mientras más datos de calidad se les entreguen para entrenar. Como en la práctica la cantidad de datos es limitada (especialmente los datos médicos), una forma en que se soluciona este problema es creando datos "falsos" y agregarlos al conjunto de entrenamiento. Para algunas tareas es razonablemente sencillo crear nuevos datos falsos. Se pueden generar nuevos pares (x, y) fácilmente simplemente transformando levemente las entradas y los objetivos del conjunto de entrenamiento.

Las imágenes tienen muchas dimensiones e incluyen una enorme variedad de factores de variación, muchos de los cuales se pueden simular fácilmente. Operaciones como trasladar las imágenes unos pocos píxeles en cada dirección a menudo pueden mejorar en gran medida la generalización. Muchas otras operaciones, como rotar o escalar la imagen también han demostrado ser bastante efectivas. A menudo, los esquemas de aumento de conjuntos de datos diseñados a mano pueden reducir drásticamente el error de generalización de una técnica de DL [23].

7.5.1.14 Batch Normalization (BN)

La normalización por lotes se aplica para abordar el problema de los cambios de covarianza interna, un cambio en la distribución de valores unitarios ocultos dentro de los mapas de características que pueden reducir la velocidad de convergencia. La BN esencialmente unifica la distribución de los valores del mapa de características al establecerlos en la media cero y la varianza unitaria, lo que, a su vez, mejora la generalización de la red al suavizar el flujo del gradiente. Intuitivamente, el problema se puede entender como dos pasos: 1) una función aprendida no es muy útil si su entrada cambia y 2) cada capa es una función y los cambios de parámetros de las capas inferiores cambian la entrada de la capa superior. Este cambio podría ser muy drástico, ya que puede cambiar la distribución de las entradas [26].

7.5.1.15 Métricas

Generalmente en los volúmenes médicos, parte de la anatomía de interés ocupa solo una región muy pequeña de la imagen. Esto a menudo hace que las funciones de pérdida y métricas convencionales tiendan a comenzar en valores muy altos haciendo que el proceso de aprendizaje quede atrapado en los mínimos locales de la función de pérdida, lo que produce una red cuyas predicciones están fuertemente sesgadas hacia el fondo y que la región de interés de primer plano se detecte parcialmente o directamente no se detecte. Es por esto que surgieron métricas específicas para estas tareas [28].

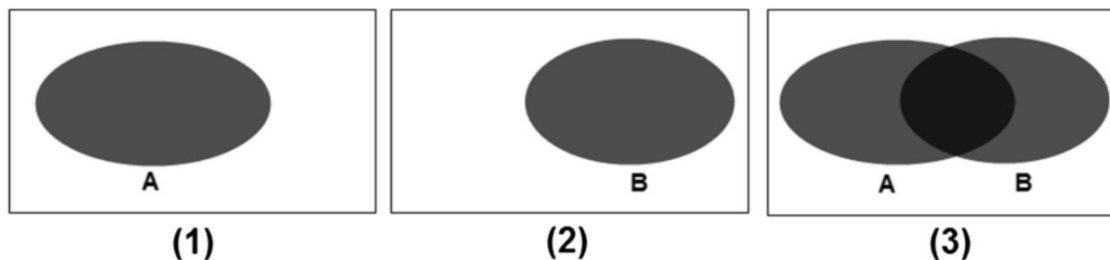


Ilustración 19: Representación de una imagen objetivo (1) y una imagen generada (2). En (3) se indica con el tono más oscuro la intersección de las dos figuras, $A \cap B$, y el total de la superficie gris indica la unión, $A \cup B$ [28]

7.5.1.15.1 IoU – índice de Jaccard

La medida de rendimiento estándar que se usa comúnmente para el problema de segmentación de objetos es la intersección sobre unión (IoU) también conocido como índice de Jaccard (JAC). Esta métrica da la similitud entre la región predicha (A) y la región real de un objeto presente en la imagen (B), y se define como el tamaño de la intersección ($A \cap B$) dividida por la unión de las dos regiones ($A \cup B$). Entonces la métrica se calcula como $IoU = TP / (TP + FP + FN) = 2 * (\sum_i^N a_i * b_i) / ((\sum_i^N a_i + b_i) - (\sum_i^N a_i * b_i))$ donde TP, FP y FN son el recuento de verdaderos positivos, falsos positivos y falsos negativos, respectivamente [29].

7.5.1.15.2 F1 score

F1-score es una combinación de *precision* y *recall*. *Precision*, también llamada valor predictivo positivo, es la fracción de valores generados, que muestra la correcta capacidad de detección del modelo. Se calcula como: $Precision = TP / (TP + FP)$, es decir los valores positivos correctamente predicho (True positive, TP, intersección) sobre la suma de todos los valores positivos predichos (valores positivos correctamente predichos y valores positivos incorrectamente predichos: False positive, FP). Y *Recall* se refiere a la relación entre los valores

positivos bien correctamente predichos y el número total de valores a predecir y se calcula como $Recall = TP / (TP + FN)$.

Entonces la métrica F1 se calcula como $F1 = 2 * Precision * Recall / (Precision + Recall) = 2 * TP / (2 * TP + FP + FN)$ [30]. Este valor es equivalente al coeficiente de Dice.

7.5.1.15.3 Coeficiente de Dice

El coeficiente de puntuación de Dice (DSC), derivado de Lee Raymond Dice, es una cantidad con rango de valores de [0, 1] que evalúa el rendimiento de la segmentación cuando se dispone de una realidad básica. Es la métrica más utilizada para validar segmentaciones de volumen médico. Este coeficiente acerca un conjunto de datos que consiste en todos los ejemplos positivos predichos por un modelo ($A \cap B$) a un conjunto de datos que consiste en la verdad básica. La pérdida de Dice se puede optimizar directamente con el DSC [31].

El coeficiente de Dice entre dos volúmenes (A y B) binarios se puede escribir de la siguiente manera: $DSC = 2 * |A \cap B| / (|A| + |B|)$ es decir: $DSC = 2 * (\sum_i^N ai * bi) / (\sum_i^N ai + bi)$ [32].

7.5.2 Deep Learning para KBP

En los últimos años, se han investigado varias arquitecturas de red de Deep Learning para KBP. A diferencia de los métodos de KBP tradicionales, que requieren ajustes y funciones hechas a mano, los métodos DL pueden aprender automáticamente las características de la imagen que se adaptan a la tarea de predicción específica a partir de los datos sin procesar (la TAC, los contornos y los mapas de dosis). Más recientemente, se han aplicado métodos de DL a tareas de predicción de dosis, debido a que son capaces de descubrir estructuras complejas y pueden aprender características directamente del conjunto de datos sin procesar. Por lo tanto, una diferencia clave entre KBP tradicional y basado en DL es la forma en que se utilizan los conocimientos previos [3].

7.5.3 Redes Neuronales Convolucionales (CNN)

Generalmente, las imágenes naturales tienen muchas propiedades estadísticas que son invariantes a la translación. Las redes neuronales convolucionales (CNN) aprovechan estas propiedades al compartir los parámetros “aprendidos” en múltiples ubicaciones de imágenes. Este uso de los parámetros ha permitido que las CNN reduzcan drásticamente la cantidad de parámetros de un modelo DL y aumenten significativamente los tamaños de red sin requerir un aumento correspondiente en la cantidad de datos de entrenamiento. El nombre “redes neuronales convolucionales” indica que emplean la operación matemática llamada convolución, que es únicamente un tipo especializado de operación lineal. Entonces, se puede decir que, las redes convolucionales son simplemente redes neuronales que utilizan la convolución en lugar de la multiplicación general de matrices, en al menos una de sus capas [23]. Más específicamente, en una CNN los datos pasan a través de múltiples capas de codificación en serie (la salida de cada capa alimenta a la siguiente) para crear un conjunto de funciones de nivel superior [5].

A medida que las entradas se procesan a través de las capas de la red, aumenta el nivel de abstracción de las características resultantes. Entonces, las capas inferiores captarán la información local, mientras que las capas más profundas utilizarán filtros con campos receptivos

mucho más amplios, capaces de captar más información global [32]. Finalmente, se introducen capas completamente conectadas en la última parte de una CNN cuando los pesos ya no se comparten. Las activaciones en la última capa se envían a través de una función SoftMax para producir una distribución entre clases, y la red se entrena utilizando la máxima probabilidad [33].

La evolución de los métodos de DL ha motivado el uso de CNN para predecir las distribuciones de dosis de vóxeles específicos del paciente a partir de información anatómica (contornos y/o TC), ya sea en forma de corte por corte (2D) o directamente como matriz 3D. La distribución de dosis predicha se puede utilizar posteriormente como objetivo para generar automáticamente un plan de tratamiento [20].

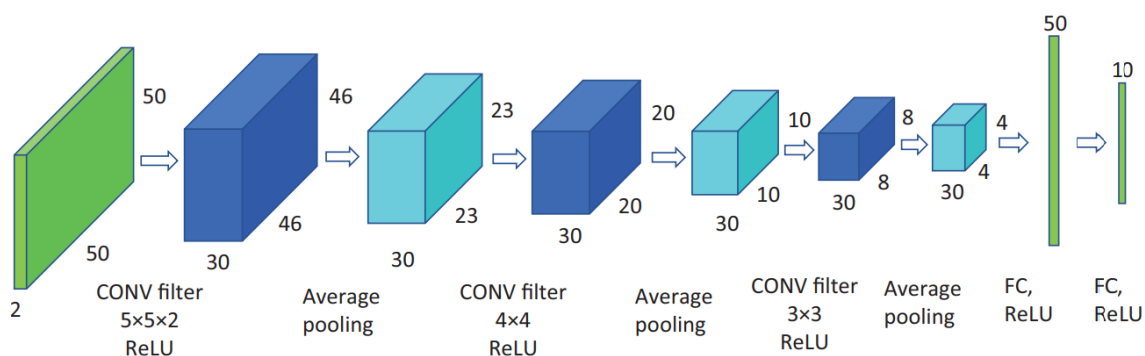


Ilustración 20: Ejemplo de una red neuronal convolucional. Fuente: Big Data in Radiation Oncology [5]

7.5.3.1 Convolución, kernels y Max Pooling

Las capas convolucionales son una parte sustancial de las soluciones con CNN en visión artificial. La convolución es una operación matemática lineal entre dos funciones (por ejemplo, f y g) que produce una tercera función h , que es una integral que expresa la cantidad de superposición de una función (f) a medida que se desplaza sobre la otra función (g) y se describe formalmente como: $h(t) = \int_{-\infty}^{\infty} f(\tau) \cdot g(t - \tau) d\tau$ y se denota como $h = f * g$ [26].

En DL, una capa de convolución consta de un conjunto núcleos o kernels compuestos por valores o pesos específicos aprendidos para convolucionar con los píxeles de la imagen, es decir, actúan como filtros que permiten extraer características de las imágenes. Los resultados finales de este procedimiento se denominan mapas de características de salida. Como se puede ver en la Ilustración 21, el proceso de convolución se compone de un producto de elementos seguido de una suma, que se repite sucesivamente con todos los píxeles de la imagen para generar una nueva matriz [26].

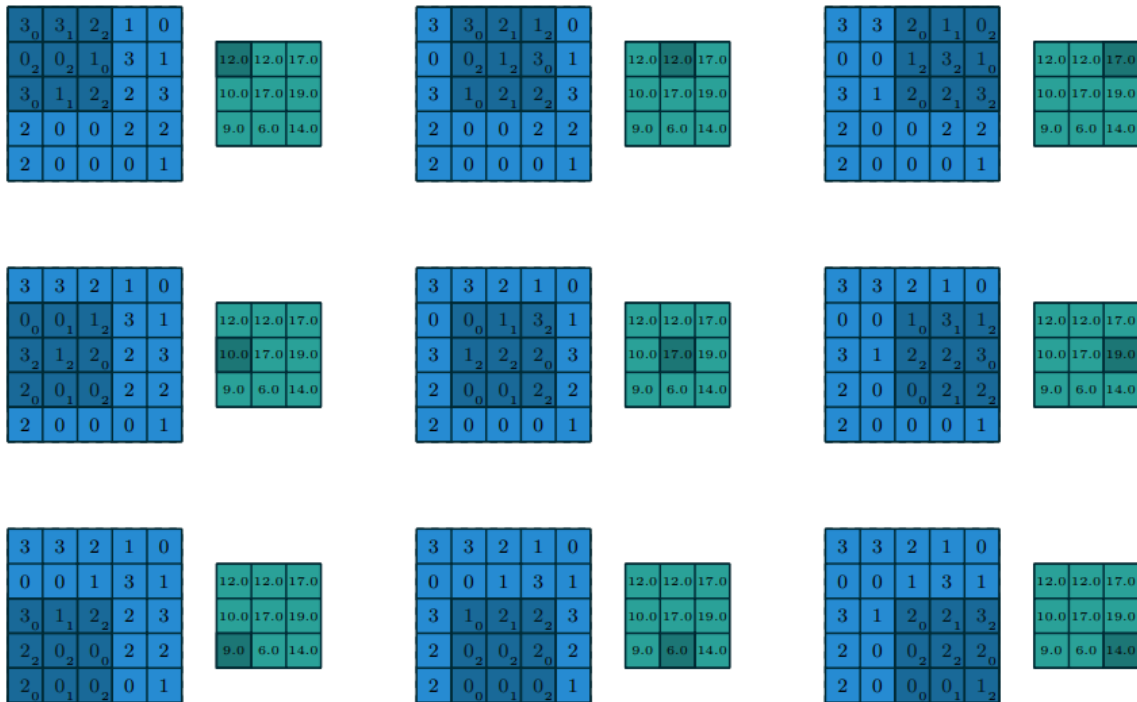


Ilustración 21: Ejemplo de una convolución en una imagen de 5x5 con un kernel 3x3, con stride de 1 y sin relleno de ceros. Un núcleo (área sombreada) de valor se desliza por el mapa de características de entrada. En cada ubicación, se calcula el producto entre cada elemento del kernel y el elemento de entrada al que se superpone y los resultados se suman para obtener la salida en la ubicación actual [34]

La salida de una capa convolucional se ve afectada tanto por la entrada, así como por la forma del núcleo, el relleno cero y los pasos, es por esto que la relación entre las propiedades de la convolución no es trivial. Como se mencionaba, en una convolución típica se recibe un arreglo como entrada y se multiplica con una serie de kernels para producir las salidas, a las que generalmente se les agrega un vector de sesgo antes de pasar el resultado a través de una no linealidad. El kernel es compartido por todas las ubicaciones espaciales de los píxeles de la entrada para crear cada mapa de características, cuyo objetivo principal es preservar su posición aproximada en relación con los demás en lugar de la ubicación exacta. Mediante diferentes elecciones de núcleos, se pueden lograr diferentes operaciones de las imágenes, suelen incluir detección de bordes, desenfoque, nitidez, etc. [26].

Cabe mencionar que, aunque los ejemplos se representan con arreglos 2D, se puede generalizar a convoluciones N-D: en una convolución tridimensional, el kernel es un paralelepípedo que se desliza a lo largo de la altura, el ancho y la profundidad del mapa de características de entrada [34].

Las CNN suelen presentar posteriormente a una convolución, una etapa de agrupación, lo que agrega otro nivel de complejidad con respecto a las redes totalmente conectadas. Las operaciones de agrupación reducen el tamaño de los mapas de características mediante el uso de alguna función para resumir subregiones, como tomar el promedio o el valor máximo (Max pooling) y así aligerar el cálculo de las capas posteriores. La agrupación funciona deslizando una ventana a través de la entrada y alimentando el contenido de la ventana a una función de agrupación, como se representa en la ilustración 22 con la selección de valores máximos.

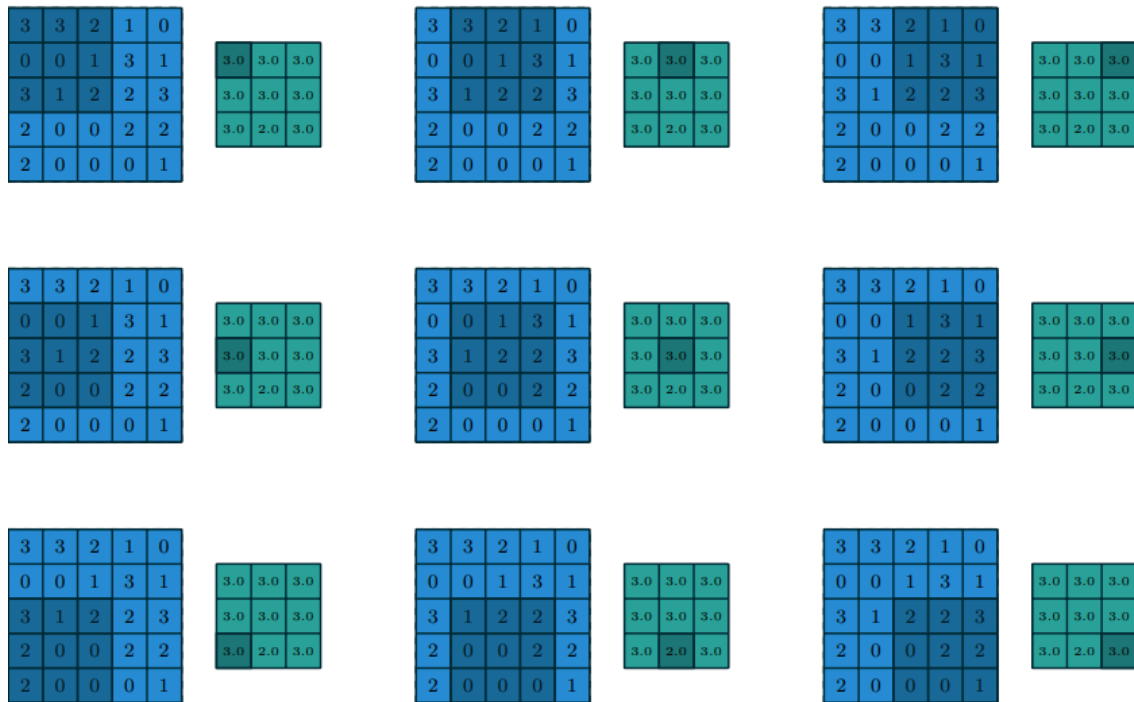


Ilustración 22: Ejemplo de Max pooling 3x3 en una imagen de 5x5 con stride de 1 [34].

7.5.3.2 Convolución transpuesta y Up-Sample

Muchas aplicaciones de DL de visión artificial requieren predicciones para cada píxel de la imagen de entrada. Los modelos CNN para tales aplicaciones generalmente se componen de un codificador de características, que disminuye la resolución espacial mientras aprende la representación de alta dimensión, y de un decodificador, que recupera la resolución espacial de entrada original [35]. Conceptualmente, se pueden ver como operaciones inversas. Un módulo decodificador consta de al menos una capa que aumenta la resolución espacial, denominada capa de muestreo ascendente. Esta es el sustento del decodificador y la mayoría de las arquitecturas eligen entre tres opciones de operación para generarla: interpolación de vecinos más cercanos, interpolación bilineal o convolución transpuesta.

Los operadores de muestreo ascendente por interpolación están basados en la distancia. Estos no tienen parámetros entrenables, pero son livianos, computacionalmente económicos y usan información estrictamente local. La mayoría ejecutan el muestreo por canal, por lo que se pueden extender fácilmente a cualquier arreglo multicanal [36]. El inconveniente de la interpolación (ya sea bilineal, vecinos más cercanos o bicúbica, porque tienen un rendimiento similar) es que se aplica una operación convolucional posterior adicional, para que existan parámetros entrenables. Esta estrategia requiere mucha memoria, resultando cuatro veces más costoso que una convolución transpuesta. Además, dado que se ejecutan solo en función de las distancias espaciales, pueden tener un desempeño insatisfactorio en tareas orientadas a los detalles, como lo es la creación de imágenes [35].

Más recientemente emergieron operadores de muestreo ascendente basados en el aprendizaje. La convolución transpuesta (también conocida erróneamente como deconvolución o convolución de pasos fraccionados) es la capa de muestreo ascendente con parámetros entrenables más utilizada en DL. Tiene kernels formados de parámetros que aprenden muestreo superior a través de la retropropagación (de igual forma que en la convolución normal) y son independientes de los datos durante la predicción (se mantienen con los mismos valores

aprendidos en toda la imagen), porque se corrige solo en el entrenamiento. De alguna forma, puede verse a la convolución transpuesta como una convolución invertida en el sentido de cómo la entrada y la salida están relacionadas entre sí: los píxeles individuales en la imagen de entrada de baja resolución se "convolucionan" iterativamente con los kernels y la salida se suma sobre las ubicaciones de destino en el espacio de alta resolución [35]. Sin embargo, no son operaciones rigurosamente inversas, ya que calcular la inversa exacta de la convolución es un problema insuficientemente restringido. En resumen, la convolución transpuesta busca devolver sus dimensiones originales a la información que pasó por un proceso sucesivo de capas convolucionales-pooling y es equivalente a intercalar las características de entrada con 0 y aplicar una operación convolucional estándar [37]. Es necesario rellenar con ceros la entrada a convolucionar, para mantener el mismo patrón de conectividad en la convolución equivalente, de tal manera que la primera aplicación (superior izquierda) del kernel solo toque el píxel superior izquierdo, como se puede ver en la ilustración 23 [34].

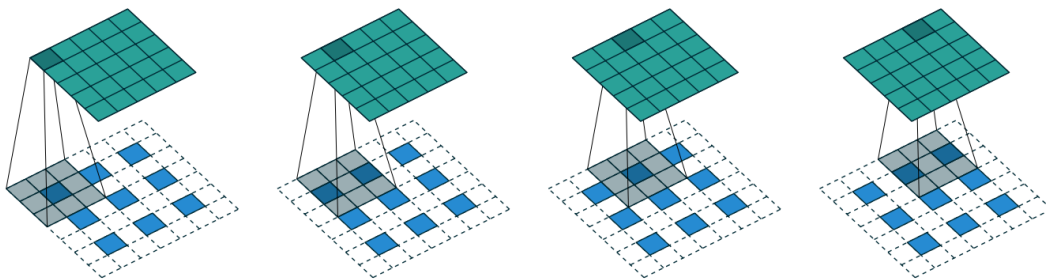


Ilustración 23: Ejemplo de una convolución transpuesta donde a una imagen 3x3 (píxeles azules) se la convierte en una imagen de 5x5 (verde) con un kernel 3x3 (sombreado) [34]

7.5.4 U-Net

La arquitectura U-Net fue presentada por primera vez por Ronneberger et al. para procesar la segmentación de imágenes biomédicas para el ISBI (International Symposium on biomedical imaging) cell tracking challenge de 2015. Es un tipo de CNN que pertenece a la clase de redes totalmente convolucionales, que es capaz de incluir características tanto locales como globales de las imágenes de entrada para generar una predicción pixel a pixel. Esta arquitectura de red está compuesta por una ruta de contracción (codificador) y una ruta de expansión (decodificador), como se puede apreciar en la ilustración 24 [20].

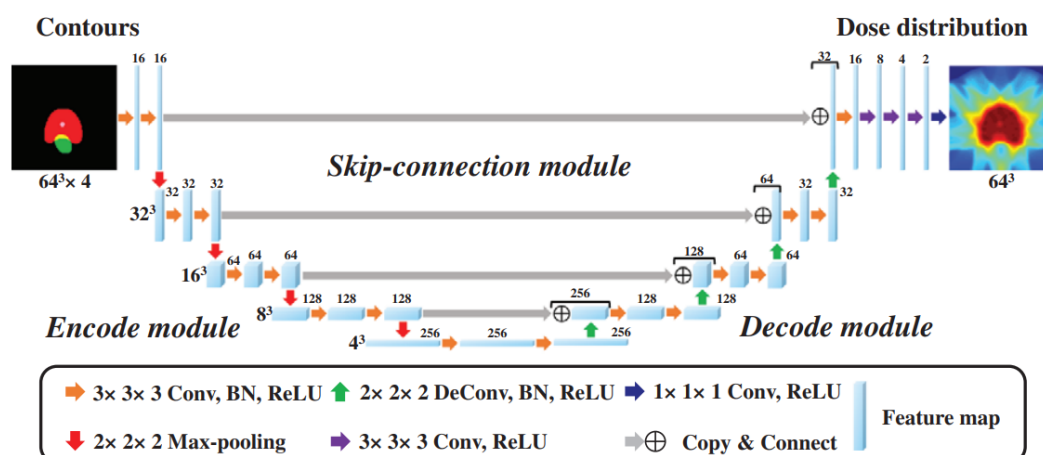


Ilustración 24: Representación de una U-Net 3D (V-Net) para predicción de dosis [38]

En el codificador se crean mapas de características de múltiples escalas con muestreo descendente y luego utiliza la operación de muestreo ascendente para fusionar dichas características de las múltiples escalas del codificador en el decodificador. La ruta de contracción sigue la arquitectura típica de una red convolucional, se compone de bloques con dos convoluciones con kernels de 3×3 , que se aplican repetidamente, cada una seguida de una unidad lineal rectificadora (ReLU) y una operación de Max Pooling de 2×2 con paso 2 para reducir a la mitad el tamaño de muestreo. En cada paso de reducción, se duplica el número de canales debido a la cantidad de kernels que se utilizan. La ruta expansiva incluye bloques con una convolución transpuesta con kernels 2×2 que reduce a la mitad el número de canales de características y duplica el tamaño de la imagen, una concatenación con el mapa de características recortado de la ruta de contracción y dos convoluciones de 3×3 , cada una seguida de una función ReLU. Entonces el resultado es que el camino expansivo es casi simétrico al camino de contracción y produce una arquitectura en forma de U. Finalmente se emplea una convolución de 1×1 en la capa final para asignar cada vector de características de 64 componentes al número apropiado de clases de objetos que se quieren segmentar [33].

7.5.5 Transfer Learning (TL) y ResNet

Se ha comprobado empíricamente en numerosos estudios que el rendimiento del DL se ve significativamente afectado por el volumen de datos de entrenamiento. Es por esto que, los modelos entrenados previamente con datasets masivos como ImageNet, se convirtieron en una herramienta poderosa para acelerar la convergencia del entrenamiento y mejorar la precisión de nuevos modelos [39]

El transfer learning (TL) o aprendizaje de transferencia, con las redes preentrenadas populares actuales (como ResNet), permite dicha optimización del rendimiento de los modelos de DL y lograr resultados más precisos cuando hay un número limitado de imágenes para el entrenamiento. El objetivo del TL es ofrecer una red troncal o “backbone” conformada de alguna de estas redes con los pesos ya establecidos a partir de entrenamientos con datasets masivos. Su efectividad se debe a que estas redes forman un codificador universal, que cubre características de bajo y alto nivel (detecciones de bordes, formas y texturas) útiles en tareas de clasificación, detección y segmentación [40]

Por otro lado, al crear las redes para TL surge la problemática de que las redes neuronales más profundas son más difíciles de entrenar. El mayor obstáculo se da por el problema de vanishing gradients o desvanecimiento de gradiente. Resumidamente, con el aumento de la profundidad de la red, el valor de gradiente llega a ser muy cercano a 0 (ya que se multiplica varias veces un valor muy pequeño) y las redes se exponen a la degradación de los datos, porque los pesos de cada entrenamiento son prácticamente iguales a los del anterior.

Una solución es abordar el problema de la degradación mediante la introducción de aprendizaje residual. Empíricamente se demostró que es más fácil optimizar el mapeo residual que optimizar el mapeo original. Las conexiones residuales o de acceso directo son aquellas que saltan una o más capas y no agregan parámetros adicionales ni complejidad computacional, como se muestra en la ilustración 25. Al agregar estas conexiones, la red aún puede ser entrenada por SGD con backpropagation al mismo tiempo que se puede implementar fácilmente usando librerías comunes sin modificar los solucionadores.

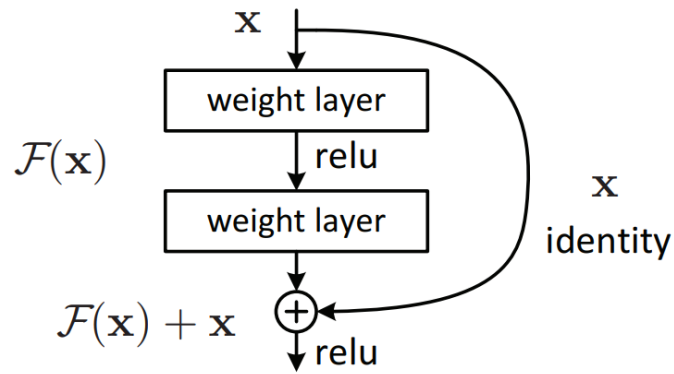


Ilustración 25: Representación de una capa residual [41]

En general, se adopta el aprendizaje residual cada pocas capas apiladas. Además, en [41], se comprobó que las redes residuales de 18 capas son precisas y convergen rápido, sin necesidad de ser muy extensas. Es decir, aunque la red "no es demasiado profunda" (18 capas), el solucionador SGD actual aún puede encontrar buenas soluciones. En la ilustración 26 se puede ver gráficamente la extensión de la ResNet18.



Ilustración 26: Representación de la ResNet18 [33]

8 Materiales y Métodos

8.1 Dataset

En la primera etapa de este proyecto, se generó el dataset que se utiliza para entrenar y evaluar las redes. Este dataset es un conjunto de 135 pares de imágenes tridimensionales de pacientes que fueron tratados en el Centro de Radioterapia Dean Funes entre las fechas 23/06/2016 y 07/11/2022. Se seleccionaron los pacientes que cumplieron los siguientes criterios de inclusión: cáncer de próstata, tratamiento con SBRT en el equipo acelerador lineal Trilogy, con dosis total prescrita de 36,25 Gy, fraccionada en 5 sesiones.

Los pares de arreglos de datos 3D consisten en un conjunto de cortes de tomografía computarizada de pelvis, con las demarcaciones de los órganos relevantes y la distribución de dosis generada en la planificación del tratamiento. De los 135 pares se seleccionaron aleatoriamente 120 para entrenamiento y validación, con una proporción de 4:1. Y los 15 casos restantes se utilizaron para testear posteriormente los modelos entrenados.

8.1.1 Simulación del tratamiento y demarcación

Para la etapa de simulación, a los pacientes se les dieron instrucciones de vaciar la vejiga y luego tomar 500 ml de agua 1 hora antes de que se realizará la tomografía. Para realizar la tomografía axial, fueron inmovilizados con una bolsa de vacío en posición supina.

Las imágenes de TAC de planificación se tomaron en un tomógrafo GE Light Speed ULTRA (GE Healthcare, Waukesha, EE. UU.). La zona escaneada fue desde el borde inferior de la vértebra L-2 hasta 5 cm por debajo del tubérculo isquiático con un espesor de corte de 2,5 mm. Posteriormente, el conjunto de datos resultantes se transmitió al sistema de planificación de tratamiento Eclipse V11.0 (Varian Medical Systems, Palo Alto, CA).

Finalmente, los oncólogos delinearon y verificaron el CTV y los órganos en riesgo, se contornearon principalmente la próstata (CTV), la vejiga y el recto, y también se tenían en las cabezas femorales, las vesículas seminales y los intestinos, entre otras estructuras OAR que no fueron tenidos en cuenta en este proyecto.

El PTV se obtuvo expandiendo el CTV en tres dimensiones con un margen de 5 mm en todas las dimensiones menos en posterior (contacto con el recto) que se usó 3mm teniendo en cuenta las incertidumbres de posicionamiento y los movimientos de los órganos.

8.1.2 Planificación

En la etapa de planificación, se produjeron planes para el tratamiento en un acelerador Trilogy (Varian Medical Systems, Palo Alto, CA). La dosis administrada clínicamente y los contornos para cada paciente se extrajeron del sistema de planificación de tratamiento Eclipse. La prescripción de dosis objetivo para todos los pacientes fue de 36,25 Gy, administrada en cinco fracciones de 7,25Gy.

Los planes se diseñaron para entregar la dosis de prescripción con técnica RapidArc™, en dos arcos completos. A lo largo de cada arco, se conformó la irradiación a la forma del tumor gracias al movimiento de un colimador multihojas (MLC, en inglés multileaf collimator).

La conformidad de la dosis se evaluó a través de la visualización de la distribución de dosis en cortes axiales, sagitales y coronales y a partir del análisis del histograma de dosis-volumen (DVH) tanto para el volumen objetivo como para los OAR.

8.1.3 Preprocesamiento

De cada paciente, se exportaron del sistema ARIA v11 (VARIAN Medical Systems) del Centro dos archivos DICOM: el conjunto de cortes tomográficos con la demarcación de los OAR y PTV y el arreglo tridimensional con el plan de dosis. Posteriormente, se preprocesaron con el software 3DSlicer (Ilustración 27) en donde se descartaron las estructuras sobrantes, quedando en todos los casos únicamente el recto la vejiga y el PTV (próstata) y se acotó el mapa de dosis al espacio ocupado estrictamente por estas estructuras. Todos los datos de los pacientes se anonimizaron completamente y se guardaron estos archivos con formato NifTI.

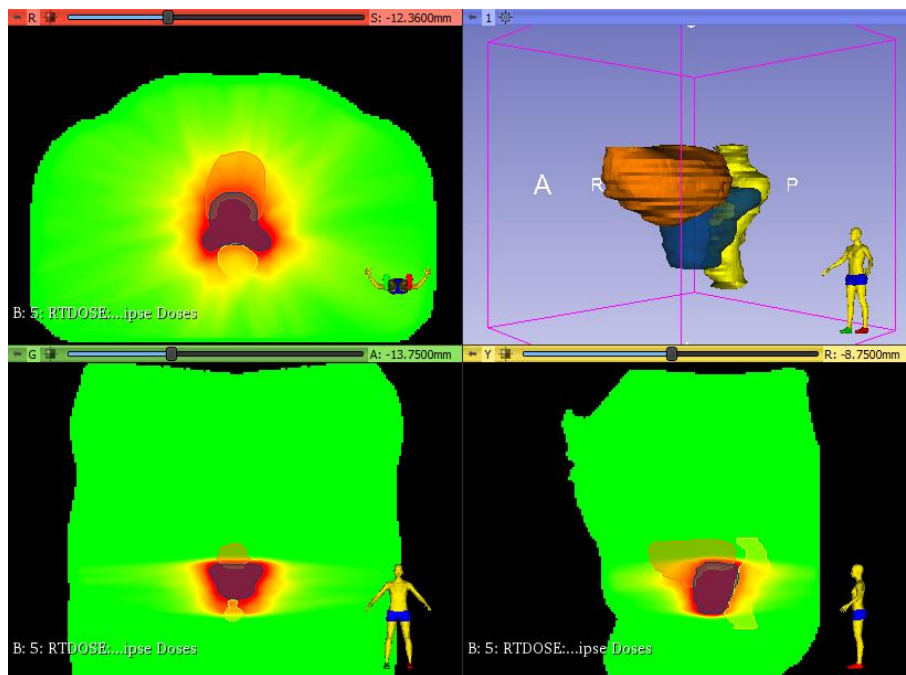


Ilustración 27: Ejemplo de cortes en los tres ejes y gráfico 3D de los datos de un paciente en el software 3DSlicer, en verde el cuerpo del paciente, en rojo la concentración de dosis, en azul el PTV, naranja la vejiga y amarillo el recto

El dataset NifTI se cargó a un Jupyter Dashboard, de un “virtual environment” creado para este proyecto, por medio del terminal Ubuntu con el siguiente código:

`~$ explorer.exe .` (para abrir la carpeta home, pegar el archivo necesario en esta carpeta)

`~$ ls` (muestra las carpetas disponibles en home)

`~$ scp -r /home/ines/pps/DatasetPI isadir@nabucodonosor.ccad.unc.edu.ar:~/` (para cargar los archivos que contiene la carpeta DatasetPI)

`~$ cd~` (para salir de las carpetas)

Posteriormente, en una Jupyter Notebook, se transformaron a tensores y se interpolaron las dimensiones de los volúmenes a $64 \times 64 \times 64$ píxeles, con la función “`torch.nn.functional.interpolate`” (que utiliza por default el algoritmo de “nearest” o “vecinos

cercanos” para el sobre-muestreo), con el fin de evitar el desbordamiento de la GPU. Luego se guardaron los pares como archivos NumPy, en la ilustración 28 se puede ver un ejemplo de estos pares ya listos para ingresar a los códigos de las CNN. Todos los códigos se generaron con la biblioteca de código abierto PyTorch versión 1.12.0+cu116.

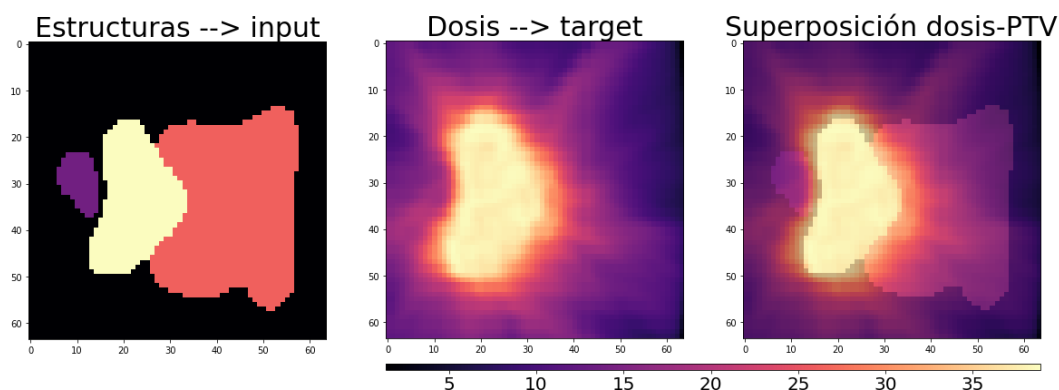


Ilustración 28: Ejemplo de un corte transversal de un volumen de estructuras (en morado el recto, en amarillo la próstata y en rosado la vejiga) (izquierda), distribución de dosis correspondiente (centro) y superposición de ambas imágenes (derecha)

Finalmente, para incrementar el dataset y tener a disposición más datos para trabajar, se realizó una etapa de “data augmentation”, es decir, incrementar el dataset con la creación de datos artificiales. Luego de un análisis de las duplas de imágenes (los datos de entrada) y del problema, se llegó a la conclusión que un “espejado” en los cortes coronales de los arreglos de datos generaría nuevos volúmenes realistas, porque las estructuras (órganos) involucradas son impares y aproximadamente ubicadas en el eje del cuerpo. El concepto de “espejar” se puede explicar cómo: siendo IM la imagen 3D original la imagen espejada resultante es ES, con $ES[x,y,z] = IM[64-x,y,z]$, siempre y cuando 64 sea la cantidad total de x's (píxeles), como ocurre en todos los ejemplos del dataset, debido al paso de interpolación realizado anteriormente.

El data augmentation se puso en marcha con la función “flipud” de la librería NumPy, que realiza un “espejado” en el primer eje de la estructura que se ingrese, y así, se consigue exactamente lo que se requería, espejando todos los cortes en el eje x, como se puede ver en la ilustración 29 y 30. De esta forma se obtuvo la duplicación del dataset, con 240 ejemplos para entrenamiento y validación y 30 para testeo final.

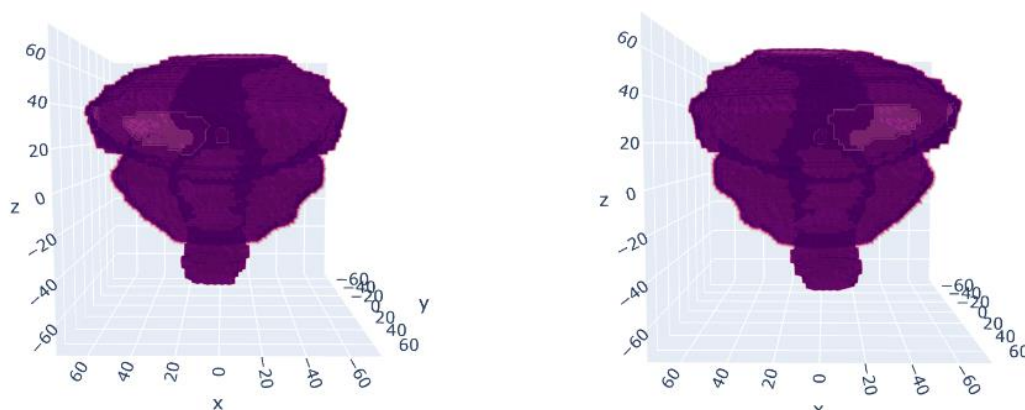


Ilustración 29: Representación 3D vista desde el frente de un volumen del conjunto original y su versión “espejada”

Los ejemplos “artificiales” creados se guardan en la misma carpeta donde se guarda su versión original, esto es importante de aclarar porque no debe existir ninguna correlación entre los datos que se utilizan para entrenar y los que se utilizan para testear, para que el testeo no

esté sesgado. Entonces en la carpeta “DatosPI/EntrenamientoPI” se tienen los 120 ejemplos originales y sus “reflejados” de forma intercalada (para no sesgar la validación al elegir los elementos finales de la carpeta) y en “DatosPI/TesteoPI” los 30 ejemplos restantes.

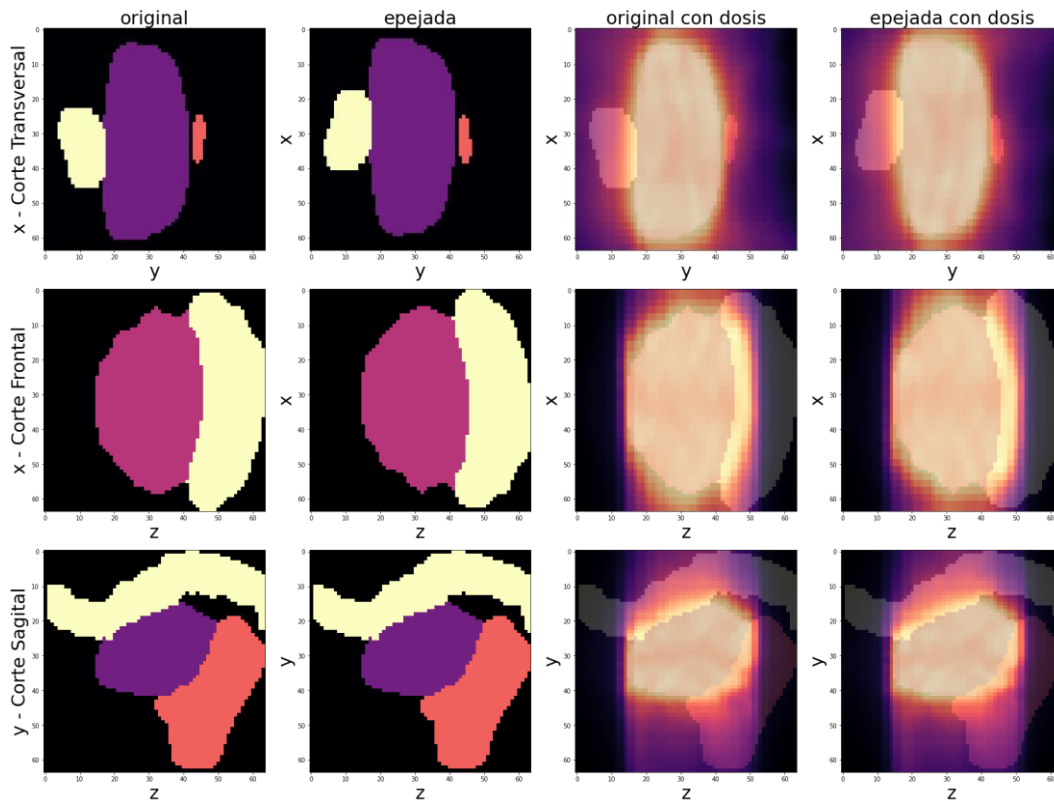


Ilustración 30: Representación de los cortes centrales en cada uno de los tres ejes, de un ejemplo aleatorio del dataset inicial, su versión del grupo “espejado” y ambos superpuestos con la distribución de dosis correspondientes

Cabe comentar que todas las gráficas de cortes generadas con la librería “matplotlib.pyplot” (como la ilustración 28 y la 30) están rotadas 90° en sentido de las agujas del reloj, por la forma predeterminada en que leen las imágenes las funciones de dicha librería. Es como si la persona estuviera en posición decúbito lateral en vez de la posición decúbito supina en la que se tomaron las imágenes, como se ejemplifica en la ilustración 31.

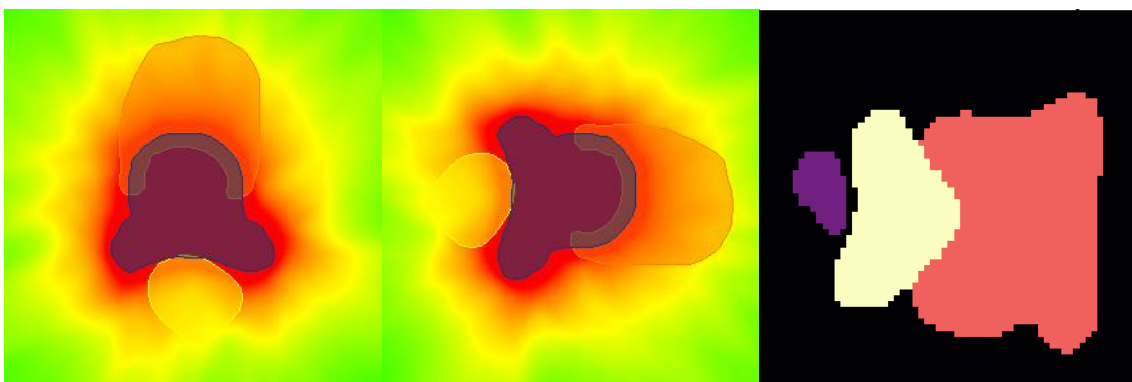


Ilustración 31: Ejemplificación de la rotación en la orientación de los gráficos generados con la librería “matplotlib.pyplot”. La primera imagen de la izquierda es la orientación original de los archivos NIfTI y la del centro muestra el equivalente NIfTI en la rotación NumPy final (derecha), dejando siempre de izquierda a derecha recto, PTV y vejiga. Las imágenes NIfTI y NumPy no corresponden al mismo paciente

8.2 Software y hardware

Para la primera etapa del proyecto se utilizó la computadora Nabucodonosor, que es un nodo único de prueba que sirve para casos particulares:

Hardware:

- Nodo Supermicro 1027GR-TSF con placa madre X9DRG-HF.
- 2 Xeon E5-2680v2 de 10 núcleos cada uno.
- 64 GiB RAM en 8 módulos de 8 GiB DDR3 1600 MT/s.
- 3 GPU NVIDIA GTX 1080Ti (GP102, 11 GiB GDDR5) conectados por PCIe 3.0 16x.
- 1 SSD 240GiB para Sistema Operativo conectados a SATA-2.
- 3 SSD 1TiB para datos en RAID0 por ZFS conectados a SATA-3.

Software:

- Debian Buster (amd64).
- NVIDIA driver 390.
- CUDA {8.0, 9.0, 9.1}.
- CUDNN {5.1, 7.0, 7.1}.
- gcc-{5.5, 6.4, 7.3, 8.2}.
- clang-{3.9, 4.0, 5.0, 6.0}.
- docker 18.06.
- nvidia-docker 2.0.3.

Fuente: web CCAD [42]

Por complicaciones con la capacidad de la memoria y los tiempos de entrenamiento se requirió utilizar una de las otras computadoras disponibles en el CCAD. Entonces para los entrenamientos finales de 500 épocas con las combinaciones con mejor desempeño se realizaron en el clúster Mendieta. Esta tiene 12 nodos de cómputo con el sistema de colas SLURM, es decir, en Mendieta se ingresa a un nodo cabecera que no cuenta con GPU y se tiene que entrar a una cola de espera para utilizar los recursos una vez que se quiere entrenar los modelos.

Hardware:

- 13 chasis Supermicro 1027GR-TSF con
- MoBo: X9DRG-HF.
- CPU: dual Intel Xeon E5-2680v2, 10 cores Ivy Bridge EP con AVX256.
- RAM: 64 GiB DDR3@1600 MT/s.
- NIC: MT25408A0-FCC-QI, Infiniband 40 Gbps.
- GPU: dual NVIDIA A30, GA100GL (14/27 de una GA100 completa), 28 SM Ampere, 24 GiB HBM2.

Software:

- Rocky Linux 8.5 (Green Obsidian)
- NVIDIA Driver 510.47.03
- gcc@11.2.0
- Open MPI 4.1.3
- NVHPC@22.3, con NVCC@11.6.112
- SLURM@20.11.9
- GROMACS@2021.5

Fuente: web CCAD [43]

9 Implementación de las redes U-Net

9.1 Datos

En cada Jupyter Notebook de cada modelo, se cargó la carpeta donde se encuentran los 240 ejemplos para entrenamiento y validación. Los volúmenes de estructuras (datos categóricos con valores 0, 1, 2 o 3 para el fondo, recto, próstata y vejiga respectivamente) se convirtieron a One-hot encoding. Esto genera que los tensores 3D (64, 64, 64) se transformen en tensores binarios 4D (64, 64, 64, 4, representado en la ilustración 33) donde en la dimensión agregada representa con 1 o 0 cada píxel según sea parte o no del órgano al que corresponde como se muestra en la ilustración 32.

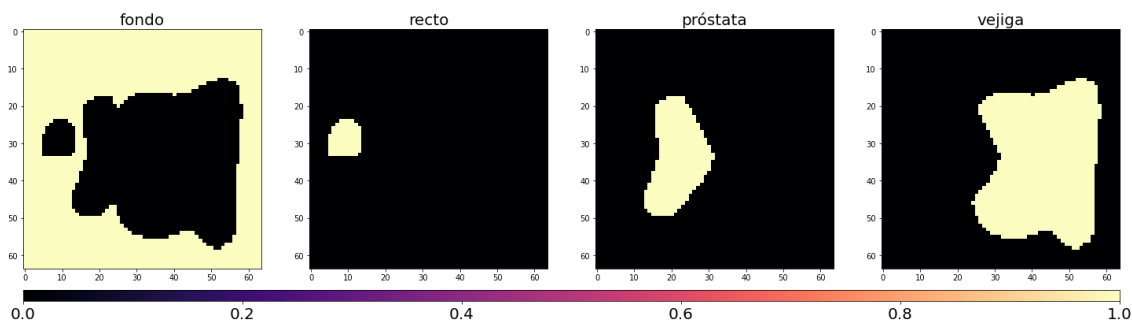


Ilustración 32: Representación de la 4ª dimensión un corte de un volumen con One-hot encoding. En vez de un solo arreglo 3D relleno de valores entre 0 y 3, ahora se tienen cuatro volúmenes 3D (en otras palabras, un tensor 4D) con valores 0 o 1, que forman parte de la información del mismo paciente

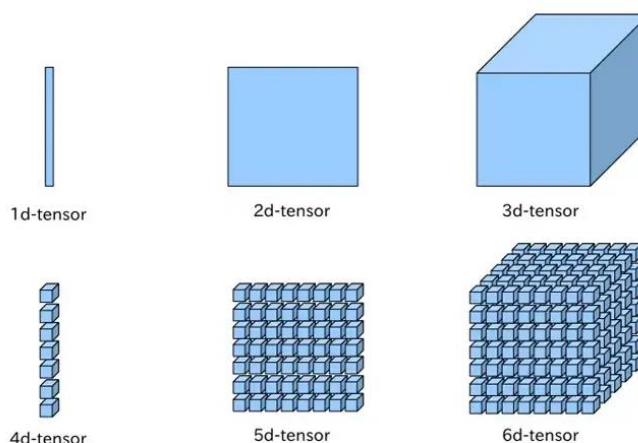


Ilustración 33: Representación del aumento de dimensión de los tensores de datos

Por otro lado, a los vóxeles que componen las imágenes 3D del plan de dosis únicamente se los escala dividiendo por 36,25 para que los valores de los píxeles que los conforman estén entre 0 y 1, ya que técnicamente la prescripción de dosis total es de 36,25 Gray. Esto es a nivel teórico ya que en realidad los valores máximos de dosis de algunos píxeles pueden llegar a valores alrededor de 38 Gy (más del 100% de la dosis prescrita).

Finalmente, con las funciones *permute* y *unsqueeze* se acomodaron las dimensiones y se agregaron las faltantes respectivamente para que los datos tengan formato [batch, imagen, alto, ancho, profundidad], es decir, los tensores de estructuras quedaron con formato 5D del tipo `torch.Size([1, 4, 64, 64, 64])` y las dosis `torch.Size([1, 1, 64, 64, 64])`, listos para ingresar en un modelo.

9.2 Diseño de las redes

9.2.1 U-Net Básica

La “primera red” que se implementó fue la U-Net Básica adaptada para datos 3D. Toma los arreglos de datos tridimensionales de los órganos como entrada y los procesa con las operaciones correspondientes, puntualmente convoluciones 3D, Max Pooling 3D y convoluciones transpuestas 3D. Al final de las operaciones la red entrega una salida como se puede ver en la ilustración 34.

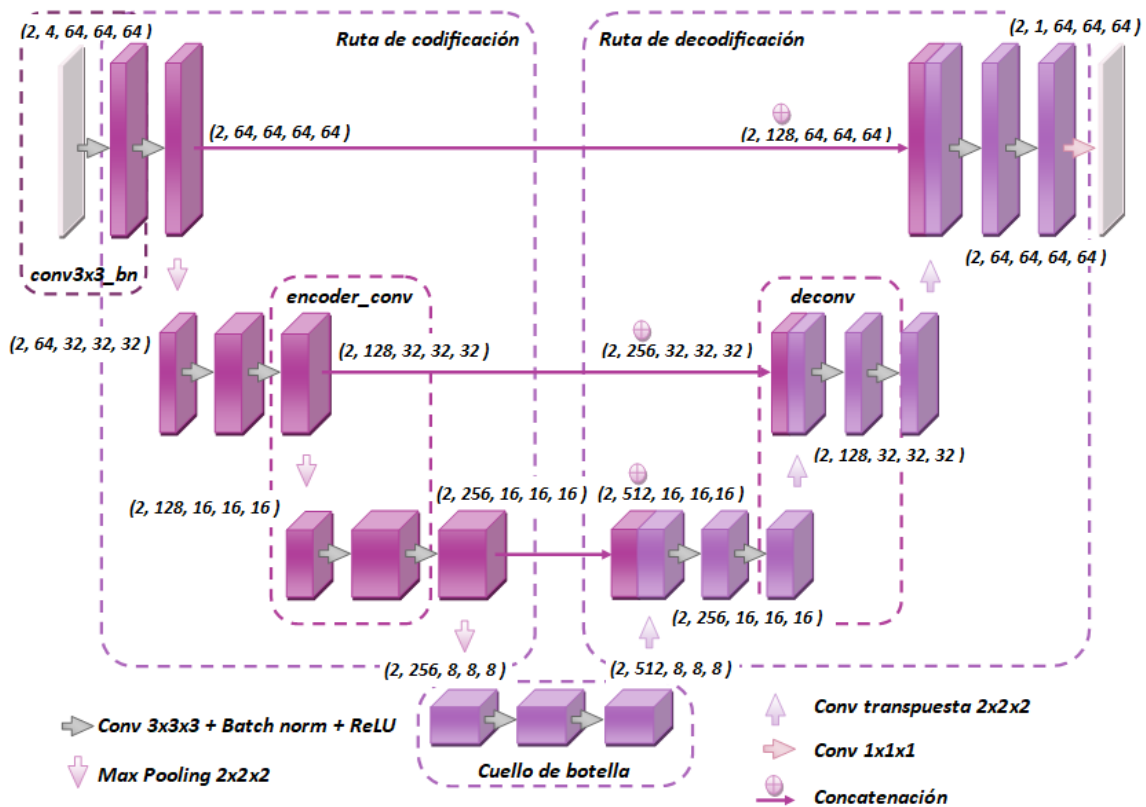


Ilustración 34: Representación conceptual de la arquitectura de la primera red implementada

La red se construyó a partir de un bloque básico llamado “conv3x3_bn”, que está compuesto por una convolución 3D con kernels $3 \times 3 \times 3$, zero padding (relleno de 0 para no perder el borde de la imagen) y stride de 1. Luego a esta operación se le agregó una capa de Batch Normalization 3D y como función de activación se utilizó una unidad lineal rectificadora, ReLU.

La primera parte de la arquitectura, la ruta de codificación, comienza con dos bloques básicos sucesivos que se utilizan como entrada de los datos y luego se repite 3 veces un bloque llamado “encoder_conv” que está compuesto de una capa de Max Pooling 3D con kernels de $2 \times 2 \times 2$ y luego dos repeticiones del bloque “conv3x3_bn”. En cada una de las 3 etapas el tamaño de los volúmenes se reduce a la mitad mientras que la cantidad de mapas de características se duplica, como muestran la ilustración 34 y el resumen que se encuentra en el [Anexo I](#).

La segunda parte, la ruta de decodificación, se construyó con 3 repeticiones de un bloque llamado “deconv” que se compone de una capa de convolución transpuesta 3D con kernels de $2 \times 2 \times 2$ y salto de 2, seguida de dos bloques básicos, donde el primero recibe como entrada la concatenación de la salida de la convolución transpuesta y la salida de la convolución del nivel correspondiente de la ruta de codificación, generando así los skip-connection. En estas

etapas los mapas de características disminuyen a la mitad mientras que se va duplicando el tamaño hasta recuperar las dimensiones de entrada.

Finalmente, la salida se obtiene con una última capa de convolución 3D con kernels 1x1x1 que cumple la función de colapsar (con una combinación lineal) todos los mapas de características que llegaron a hasta ese punto en únicamente uno y así generar el mapa de dosis que se está prediciendo.

De esta forma se consigue una estructura con 52 operaciones y más de 22 millones de parámetros entrenables como se detalla al final del resumen en el [Anexo I](#).

Para comprobar la compilación de la red y tener una idea previa del funcionamiento y sus capacidades se realizó un ensayo de entrenamiento de 5000 con un solo ejemplo (sobreajuste) y se graficó un corte de la predicción lograda, ilustración 35.

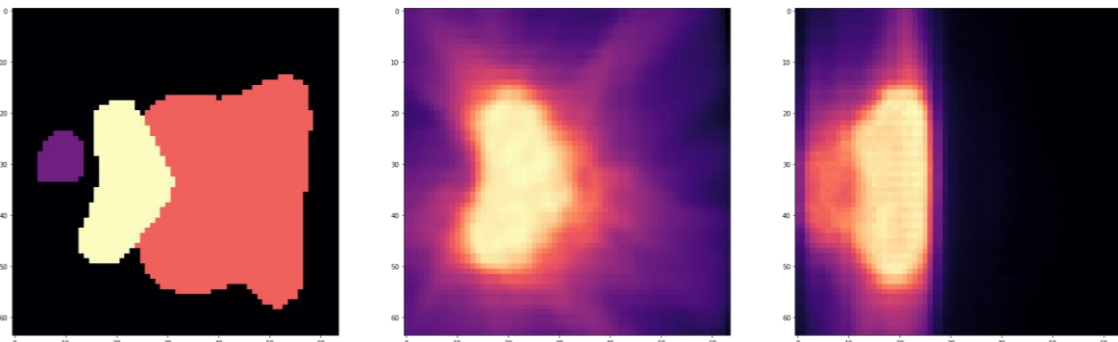


Ilustración 35: Representación de un corte transversal de las estructuras (izquierda), el objetivo (centro) y la predicción (derecha), obtenida con el entrenamiento de 5000 épocas para un solo ejemplo con la red U-Net Básica

9.2.2 U-Net mejorada

En la siguiente etapa se generó una arquitectura similar, pero agregando una capa más de profundidad y ciertas funciones que se esperaba mejoraran el desempeño.

Para esta “segunda red” se comenzó generando una clase llamada “*encoding_block*”, en la que se realizaron operaciones equivalentes a dos bloques “*conv3x3_bn*” seguidos. La primera novedad que se implementó fue la función de la librería de PyTorch “*ReflectionPad3d*” que cumple la función del zero padding de la convolución, solo que en vez de generar un relleno de 0s realiza un relleno con los valores vecinos al borde, como se muestra en la ilustración 36 y 37. Este cambio de operación introduce menos ruido a las convoluciones, ya que se están agregando datos similares a los que se tenían originalmente. En segundo lugar, se sustituyó la función de activación ReLU por una función PReLU (Parametric Rectified Linear Unit) que agrega un parámetro que se va actualizando a medida que entrena la red. Y, por último, se agregó al final de esta clase una capa dropout para evitar cualquier sobreajuste que se pudiese llegar a generar. Con lo anterior armado, para la ruta de codificación se realizó una secuencia de “*encoding_block*” y capas Max Pooling 3D para generar la extracción de características. Luego, para la ruta de decodificación, se creó una clase llamada “*decoding_block*” donde se permite elegir entre una operación de convolución transpuesta o una operación de Upsampling combinada con una convolución normal. Y como capa de salida se utilizó una convolución 1x1x1 para generar el volumen de distribución de dosis de igual forma que en la primera versión del modelo.

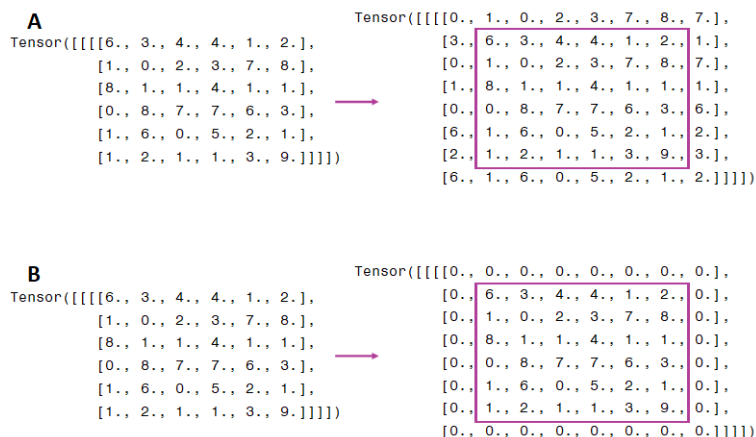


Ilustración 36: A: Representación de una operación ReflectionPad donde se genera un padding de 1 para convolucionar con kernels 3x3. B: Representación de zero padding para la misma situación

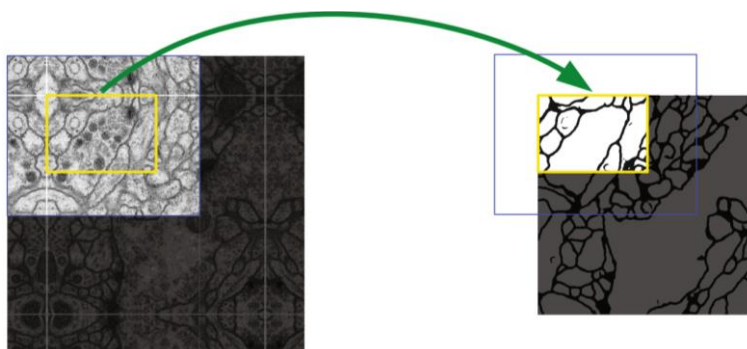


Ilustración 37: Representación del trabajo de Ronneberger et al. en donde se puede ver la utilización de la operación "ReflectionPad" a gran escala (se puede apreciar como la imagen por fuera de las líneas blancas se ve como una copia espejada de lo que se encuentra por dentro) [7]

De esta forma, al ejecutar la red con todos los cambios y seleccionando la función convolución transpuesta para la ruta de decodificación, resultó que la cantidad total de parámetros se incrementó aproximadamente en un factor de 4, siendo específicamente 85.435.420 parámetros entrenables en un total de 112 capas, detalladas en el [Anexo I](#).

Con estos bloques se genera una estructura similar de U que en la primera red solo que esta vez se agrega una capa más, de forma tal que en el cuello de botella las dimensiones de los tensores llegan a ser de 1024 mapas de características y tamaño 4x4x4 como se puede ver en el final del resumen de la red, que se encuentra en el [Anexo I](#).

La mejora que se introdujo en la etapa de entrenamiento fue la función de PyTorch "lr_scheduler.ReduceLROnPlateau", con la que la tasa de aprendizaje (que comienza en 1e-4) se divide por 10 cuando la pérdida de validación no se reduce 15 épocas sucesivas hasta llegar a 10-8.

Al igual que con la primera arquitectura, para comprobar el funcionamiento se realizó un ensayo de entrenamiento de 5000 con una sola imagen y se graficó la predicción lograda, como muestra la ilustración 38. En esta oportunidad, se puede ver la diferencia que hace haber cuadruplicado la cantidad de parámetros entrenables, permitiendo ajustar de mejor forma la imagen de dosis, que son datos complejos.

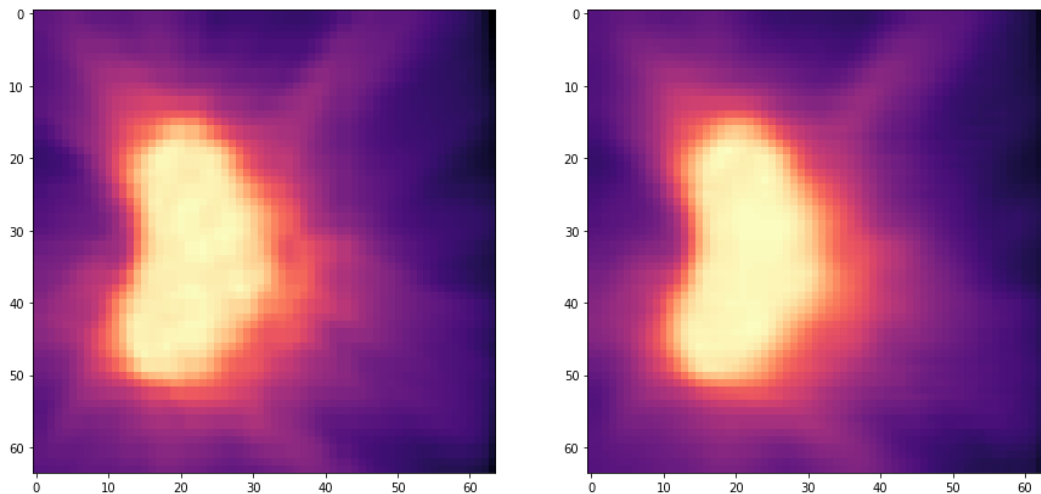


Ilustración 38: Representación de un corte transversal del objetivo (izquierda) y de la predicción (derecha) obtenida con el entrenamiento de 5000 épocas para un solo ejemplo con la red U-Net Mejorada

9.2.3 ResU-Net

La “tercera red” creada es una combinación de una red 3D U-Net y la ResNet18 3D mediante la técnica de transfer learning (TL), y es denominada ResU-Net. Al igual que en las redes anteriormente implementadas, esta consta de una ruta de codificación que extrae las características de la imagen y una ruta de decodificación que le devuelve las dimensiones para lograr la predicción de la dosis. Para su constitución se recuperaron los bloques “conv3x3_bn” y “deconv” de la primera red al mismo tiempo que se les agregaron las capas ReflectionPad3d, PReLU y dropout de la segunda red.

La novedad es que ahora la ruta de codificación se creó con TL, utilizando capas entrenadas previamente de la red “r3d_18” (ResNet18 3D), disponible en la librería de PyTorch “torchvision.models”. Para que su uso sea viable, primero se debe adaptar la capa de entrada, ya que, en un principio, a esta red la crearon para recibir imágenes RGB, por lo que la cantidad de canales de entrada son 3 y no 4, como los que se necesitan para ingresar los volúmenes de estructuras con formato One-hot de este trabajo. Una vez que se redefinió la convolución de entrada, las siguientes capas se pueden utilizar directamente ya que están hechas con las mismas cantidades de mapas de características que la que se venían trabajando (64, 128, 256 y 528), lo que significa que son totalmente compatibles con los skip-connections o concatenaciones de la U-Net.

Detallando más el procedimiento del TL, cabe aclarar que, para “descargar” la red con los pesos de los kernels ya entrenados, se utiliza la función “torchvision.models.video.r3d_18”. Al tener la red cargada en la notebook se pueden llamar individualmente sus capas para usarlas. Primero se llama la capa de entrada llamada stem y se cambia el primer valor (tamaño de entrada, de 3 a 4) de la convolución 3D. Luego simplemente se deben llamar las siguientes capas a medida que se van utilizando, estas se denominan layer1, layer2, layer3 y layer4 como se puede ver en la descripción detallada de la red en el [Anexo II](#). Entonces, teniendo a disposición estas capas, se arma el “backbone” del TL que básicamente es la ruta de codificación formada por la sucesión de las “layers” de la ResNet 3D. Es importante tener cuidado de que a medida que se van implementando dichas capas, se vayan guardando los resultados en variables distintas, de forma tal que estén disponibles más adelante para la concatenación en la ruta de

decodificación. Luego la ruta de decodificación se construye con bloques “*deconv*” que van achicando la cantidad de mapa de características de forma simétrica a la ruta de codificación. Para finalizar se utiliza un bloque llamado “*out_conv*” que contiene la última convolución transpuesta y la convolución de salida con kernel 1x1x1. Como se puede deducir del resumen del [Anexo I](#), esta red vuelve a tener únicamente tres skip-connection, como en la primera implementación, es decir la cantidad máxima de mapas de características es de 512, aunque el menor tamaño que alcanzan es 4x4x4 (como en el segundo modelo), ya que en la capa stem de la ResNet18 3D se realiza una convolución con kernels 7x7x7 con salto o stride de 2 (reduciendo a la mitad el tamaño de entrada).

De esta forma, en esta oportunidad, al ejecutar la red resultó que la cantidad total de parámetros se disminuyó aproximadamente en un factor de 2 en relación al modelo anterior (duplicando el primer modelo), siendo específicamente 41.716.232 parámetros entrenables en un total de 109 capas, detalladas en el [Anexo I](#).

Para continuar con comprobando el correcto compilado de la red y tener una idea precoz de sus capacidades se realizó, al igual que con los otros modelos, un ensayo de entrenamiento de 5000 con el mismo ejemplo (sobreajuste) y se graficó la predicción lograda, como se ve en la ilustración 39.

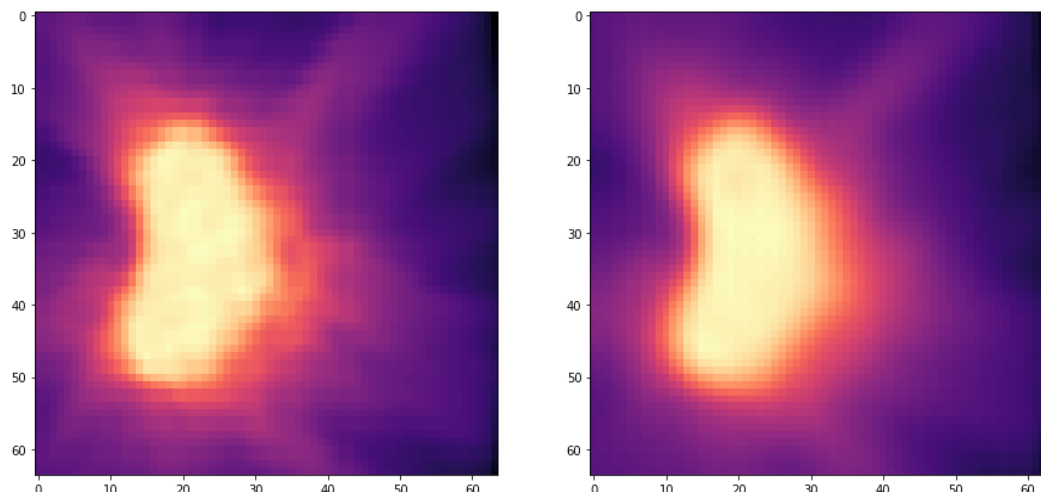


Ilustración 39: Representación de un corte transversal del objetivo (izquierda) y de la predicción(derecha) obtenida con el entrenamiento de 5000 épocas para un solo ejemplo con la red ResU-Net

9.3 Entrenamientos / validaciones

En la primera etapa de puesta en marcha de las redes, se realizaron una serie de entrenamientos cortos para ir tanteando qué combinación de características obtenía el mejor desempeño con cada red. Entonces, para probar preliminarmente los modelos creados se realizó una sucesión de entrenamientos variando las funciones de loss y la cantidad de épocas. Para esto se generó una función llamada “*fit*” en la que se definen todas las configuraciones para el entrenamiento (*model.train()*) y verificación (*model.eval()*):

- Optimizador: función Adam
- Mini Batch: se entrenó con conjuntos de 2 volúmenes (por limitación de hardware).
- Pérdida: MSE, L1, BCEWithLogitsLoss y SharpLoss
- Device: CUDA
- Métricas: IoU y Dice
- Epocas: se probaron diferentes números de épocas, 10, 100 y 500.

- Learnig Rate: en la primera red se utiliza $1e-4$, en la segunda y tercera se comienza con ese valor y se va adaptando con la función `"lr_scheduler.ReduceLROnPlateau"`

En cada época del entrenamiento se van guardando los avances de los cálculos del loss, el loss de verificación y las métricas de IoU y Dice, para que al término del proceso se pueda realizar un gráfico en el que se vea la evolución de estos valores en el desarrollo del entrenamiento. Además, se agregó a la función fit la capacidad de ir guardando los avances `"checkpoint"` del entrenamiento únicamente si el valor de la pérdida disminuye. Esto genera un backup o copia de seguridad del avance del entrenamiento del modelo que permite recuperar los valores de los parámetros entrenados si es que se detiene el proceso antes de terminar todas las épocas programadas.

Las métricas que se utilizan para tener un seguimiento en todos los entrenamientos son IoU y Dice, que varían de 0 a 1, donde 1 se considera una coincidencia perfecta con el objetivo. Estas métricas fueron ideadas en un principio para problemas de segmentación, en donde los datos son binarios y la intersección se puede calcular como una simple y derivable multiplicación (solo hay intersección donde coinciden valores 1, de otra forma o no coinciden los datos o son parte del fondo). Entonces, cabe aclarar que para este proyecto se utilizaron estas métricas simplemente para poder hacer un seguimiento de la mejora de las predicciones mientras avanza el entrenamiento. En el cálculo de las métricas se utilizaron las imágenes 3D, pero con los valores escalados de dosis redondeados a un solo dígito después del punto decimal. De esta forma las distribuciones de dosis quedan expresadas como porcentajes de redondos (10%, 20%, 30%, etc., se puede apreciar mejor en la ilustración 40), haciendo más probable la coincidencia entre valores de los vóxeles de la predicción y del objetivo. Esto se puede pensar como que se partió del umbral de 0 a 36,25 Gy de dosis y se generaron intervalos de 3,625Gy. Se decidió proceder de esta forma porque si se agregaba más decimales el cambio del valor de cualquiera de las métricas era prácticamente imperceptible y no cumplían su función.

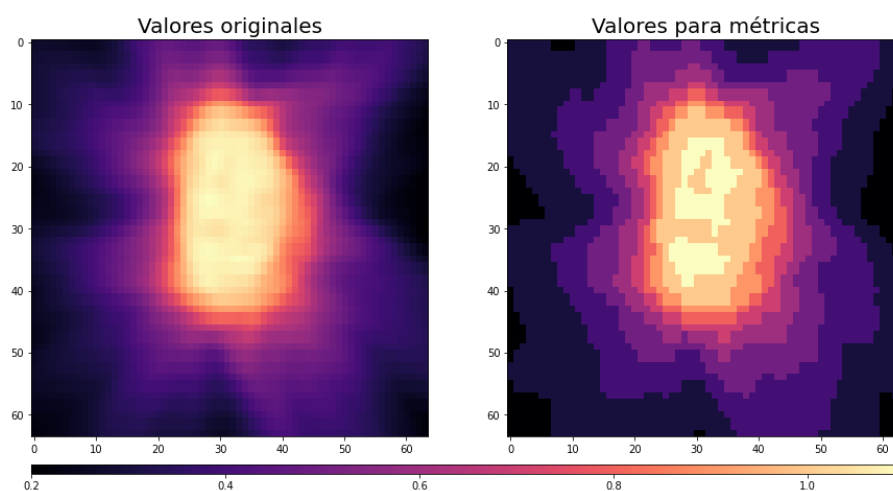


Ilustración 40: Representación de un corte transversal de un mapa de dosis de un ejemplo con los valores de vóxel originales y el mismo corte con los valores redondeados para calcular las métricas Dice e IoU

Debido a esta forma particular de aplicar las métricas, los valores finales no son directamente comparables con los valores que tiene para estas métricas otros trabajos del estilo, como el de Nguyen et al. [4].

Una vez obtenidos los resultados de esta serie de entrenamiento, se acomodaron en la [tabla 1](#) y se seleccionaron las combinaciones con mejor desempeño para realizar un entrenamiento final con 500 épocas en el segundo hardware. Este cambio no estaba planificado en un principio, pero se tomó la decisión de cambiar de hardware al probar empíricamente que, el que se estaba utilizando hasta ese momento, no tenía la capacidad de cómputo necesaria para hacer dicho entrenamiento con las dos últimas redes que son sustancialmente más grandes que la primera.

Como se venía diciendo, la U-Net Básica tiene menos parámetros entrenables, entonces es más rápida de entrenar. Por esta razón es que con esta red se probaron todas las funciones de pérdida y cantidad de épocas que se tenían planeadas (y que se fueron agregando), y según los resultados se decidían si hacer o no el entrenamiento con las otras redes. Es por esto que la parte de la tabla dedicada a la primera red es mucho más extensa que la de los otros dos modelos juntos. Todos estos entrenamientos se realizaron con la computadora del Centro de Computación de Alto desempeño de la UNC, Nabuconodosor. Esta computadora, aunque es muy simple de utilizar (similar a Google Colab) tienen una capacidad de cálculo y memoria limitada y es por esto que se utiliza en todos los entrenamientos un tamaño de batch demasiado pequeño (batchsize = 2), lo que genera que los entrenamientos tarden más tiempo y se sobreajuste la red.

En la segunda etapa, teniendo en cuenta los resultados obtenidos en la anterior, se eligió una combinación de parámetros final para cada red y se procedió a correr un entrenamiento de 500 épocas para cada una y se obtuvieron los resultados presentados en la [tabla 2](#). Los entrenamientos de 500 épocas se realizaron con la función de pérdida MSE loss para las tres redes, que es la que había mostrado mejor desempeño en el paso anterior. Estos tres entrenamientos se realizaron de forma idéntica a los anteriores solo que esta vez, los cálculos se realizaron con el clúster Mendieta que tiene mayor capacidad y permitió utilizar batches de 16 ejemplos y terminar cada tarea en un tiempo menor. En esta etapa se buscó realizar un entrenamiento excesivo para poder comparar las tres redes en igualdad de condiciones, ya que seguramente para la primera red no serían necesarias tantas épocas de entrenamiento para alcanzar su máximo desempeño.

9.4 Testeos

Una vez terminado cada entrenamiento largo, con la función “*predecir*” se generó y guardó una serie de predicciones de distribución de dosis para los ejemplos del set de prueba (los 30 ejemplos que se habían reservado en un principio y no fueron utilizados antes). Por cada red se creó una carpeta en la que se guardan las 30 predicciones conseguidas con esa red con el nombre “*pred_*” y el nombre del objetivo original (target) de ese ejemplo, de esta forma no se pierde el orden de las imágenes, es decir, no se pierde la correlación del mismo ejemplo en la misma posición en todas las carpetas (como se puede ver en la ilustración 41). Estas predicciones son las que se utilizaron para generar los gráficos comparativos y el análisis de resultados.

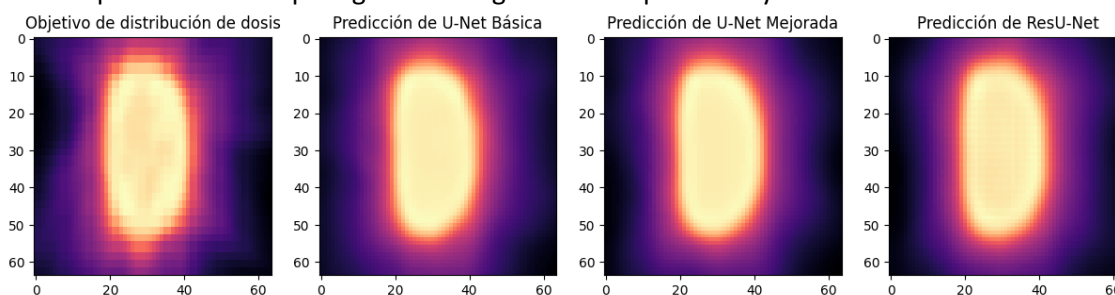


Ilustración 41: Representación de un corte transversal del mapa de dosis del ejemplo ubicado en la posición 6 de las cuatro carpetas (Dosis, UB, UM y RU)

El primer gráfico que se realiza para evaluar las predicciones es uno similar al de la ilustración 42, que se presenta en el paper de Sumida et al. En este gráfico se representa con un punto el valor del índice de Dice, para un porcentaje de dosis específico, de los mapas de dosis respecto al objetivo, con un color diferente para cada red. Entonces en el eje de las abscisas se tiene los valores relativos (porcentuales) de dosis y en el eje de las ordenadas se tiene el valor del índice de Dice (entre 0 y 1). Para este proyecto se adaptó este gráfico de forma tal que se tiene 13 puntos correspondientes a los 13 intervalos porcentuales de dosis que se generan al redondear los valores que van desde el 0 al 1.2 aproximadamente. Se decidió proceder de esta forma porque al tener tres modelos, a diferencia de los dos modelos que se comparan en el paper, la cantidad de datos a manipular aumenta considerablemente.

En este trabajo primero se presenta la comparación entre las tres redes y el objetivo para unos ejemplos puntuales y al final se muestra este gráfico con los valores promediados del grupo completo de los 30 ejemplos de testeo.

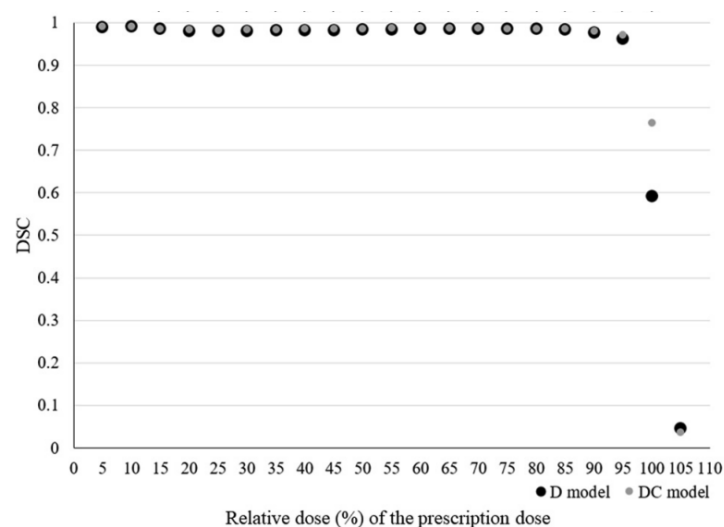


Ilustración 42: Gráfico del paper de Sumida et al., "Valores medios de índice de Dice de 16 pacientes evaluados entre la dosis-volumen predicha en los dos modelos con respecto al volumen de dosis de referencia" [44]

Posteriormente, se realizó un gráfico para poner los resultados en contexto, una comparación entre los histogramas de distribución de dosis en cada órgano. Para conseguir los histogramas diferenciados de cada órgano primero se requiere realizar el one-hot encoding de la imagen de órganos y graficar las cuatro capas binarias cómo se realizó anteriormente con los datos de entrenamiento. La gráfica del one-hot se realiza porque la capa donde se guarda cada órgano no es constante entre todos los ejemplos del dataset, debido a que la forma en la que se realizó la demarcación original no es constante. Entonces se debe identificar manualmente cada órgano para el ejemplo particular y cargar esa información a la función que realiza los histogramas. Teniendo identificada cada capa binaria se realiza una multiplicación (filtrado) con la imagen de dosis y de esta forma se guarda solamente el valor de los vóxeles de dosis para cada órgano.

Luego, con cada dosis filtrada para cada órgano se realiza el gráfico donde se muestra una cuadrilla de histogramas (tres filas y cuatro columnas) en la que cada fila tiene los histogramas correspondientes al mismo órgano y las columnas corresponden a la misma predicción/objetivo (teniendo en la primera columna la información de la distribución de dosis

objetivo). Los histogramas tienen en el eje de las abscisas la dosis medida en porcentajes relativos y en el de las ordenadas el conteo de vóxeles, también en valor relativo.

Finalmente, con la información de cada histograma, se realiza un gráfico DVH acumulado (histograma dosis-volumen) en el que, en un único gráfico, se representa las curvas de los histogramas acumulados para cada órgano en cada predicción y en el objetivo a comparar, similar al que se muestra en el paper de Murakami et al. (Ilustración 43).

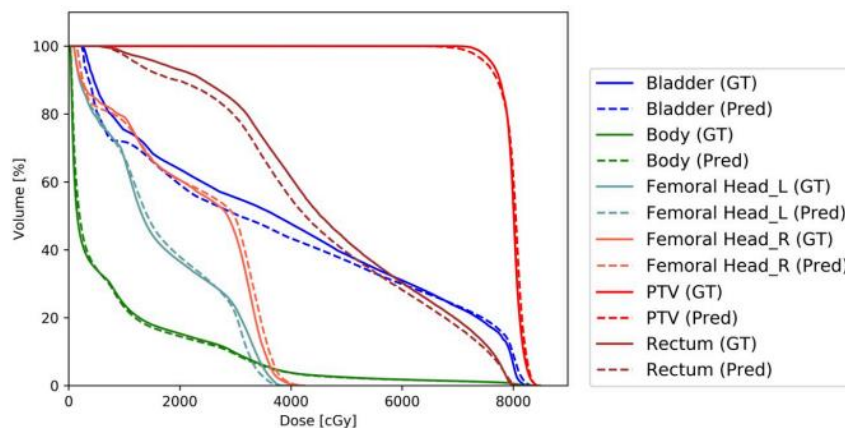


Ilustración 43: Gráfico DVH del volumen objetivo y predicción del paper de Muerakami et al [22]

Ejemplos de todos estos gráficos se presentan en la sección [Comparación con la planificación tradicional](#). En dicha sección, se verá como tendencia general para todos los ejemplos, es que la gráfica de los índices de Dice se ve desproporcionadamente menos satisfactoria que los histogramas y el DVH acumulado. Como línea general, se debe tener en cuenta que en los histogramas y el DVH se pierde la información posicional de los vóxeles, por lo que solo reflejan cantidades de vóxeles, mientras que los índices de Dice representan una comparación vóxel a vóxel entre la predicción y el objetivo, lo que genera que sea mucho más difícil conseguir similitudes (valores altos) en dicha gráfica.

10 Resultados

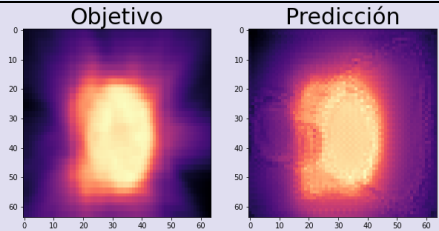
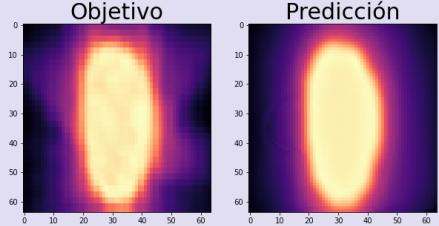
10.1 Desempeño de los modelos

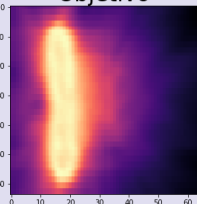
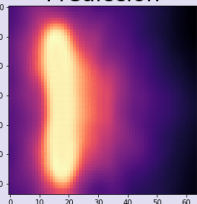
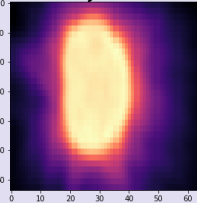
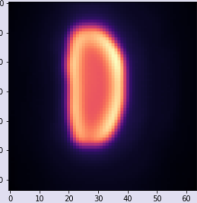
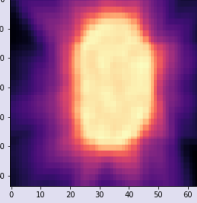
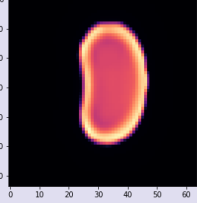
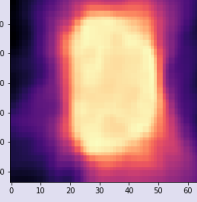
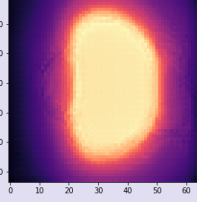
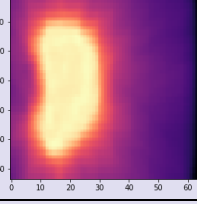
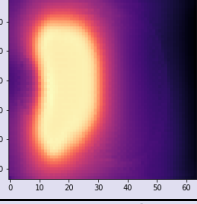
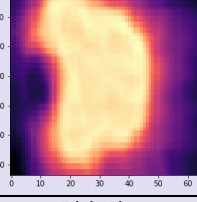
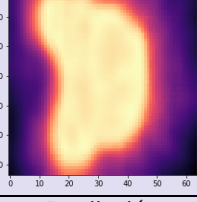
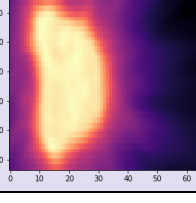
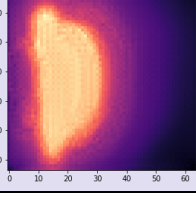
10.1.1 Primera etapa de entrenamientos

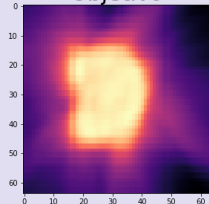
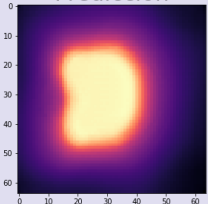
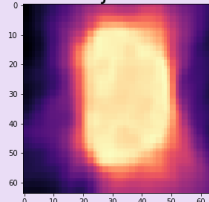
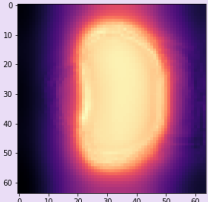
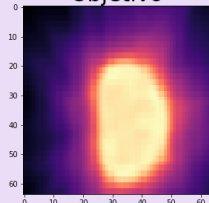
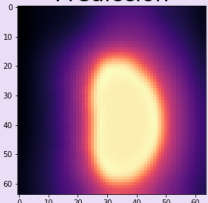
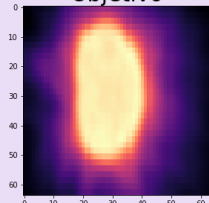
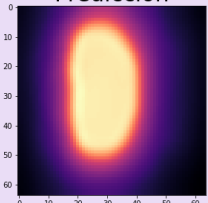
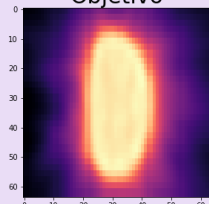
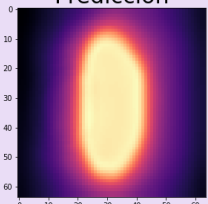
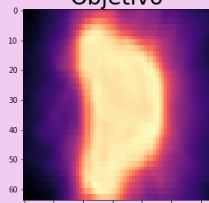
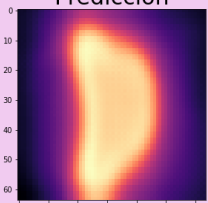
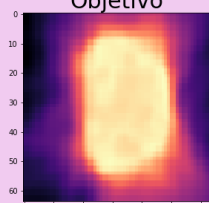
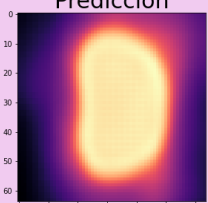
A medida que se fueron realizando entrenamientos en la primera etapa, se consiguieron variaciones de resultados y métricas que se pueden ver en la tabla 1. En las columnas de dicha tabla, primero se identifica cuál de los modelos anteriormente detallados se utiliza. Luego se aclaran cuáles fueron las condiciones particulares con las que se entrenó en cada instancia, siendo los datos más relevantes el tipo de función de pérdida utilizada y la cantidad de épocas realizadas. A partir de estos dos datos, se pueden comparar los desempeños de los entrenamientos con la misma función de pérdida con el valor final del loss de entrenamiento y validación obtenidos, que se indican en la siguiente columna. Por otro lado, independientemente de la función de pérdida, se pueden comparar todos los entrenamientos entre sí con el valor final de las métricas generadas, IoU y Dice. De todas formas, cabe aclarar, que estos valores de índices (Dice e IoU) son meramente orientativos, ya que son el cálculo resultante para el último ejemplo del set de validación que ve la red en cada época de entrenamiento, que no suele ser siempre el mismo porque está activada la función “*shuffle*” en el armado del dataloader. Es por esto que para cualquier otro ejemplo cualquiera de los índices no va a tener el mismo valor, aunque sea la misma red entrenada.

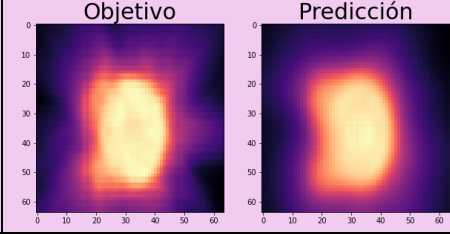
Finalmente, el objetivo de la última columna de la tabla es permitir hacer una inferencia rápida de la calidad de las predicciones mediante la representación aleatoria de dos cortes transversales, aproximadamente a la mitad, de un ejemplo del set de validación y su correspondiente predicción generada con la red entrenada con las condiciones detalladas.

Tabla 1: Características de cada entrenamiento y valores obtenidos

Modelo	Particularidades: 1- Épocas 2- Optimización 3- Learning Rate 4- Pérdida 5- Mini Batch	Resultados: - Pérdida de train y val - Índices Dice e IoU de val	Comparación de la predicción y el objetivo de ejemplo aleatorio del conjunto de validación
U-Net Básica	1- 10 2- Adam 3- 1e-4 4- MSELoss 5- 2	Loss: 0.00328 test_loss: 0.00408 Dice: 0.656 IoU: 0.488	
	1- 100 2- Adam 3- 1e-4 4- MSELoss 5- 2	Loss: 0.00076 test_loss: 0.00135 Dice: 0.734 IoU: 0.580	

1- 500 2- Adam 3- 1e-4 4- MSELoss 5- 2	Loss: 0.00011 test_loss: 0.0096 Dice: 0.787 IoU: 0.649	 
1- 10 2- Adam 3- 1e-4 4- BCEWithLogitsLoss 5- 2	Loss: 0.36290 test_loss: 0.35471 Dice: 0.011 IoU: 0.006	 
1- 100 2- Adam 3- 1e-4 4- BCEWithLogitsLoss 5- 2	Loss: - 0.04098 test_loss: - 0.09824 Dice: 0.011 IoU: 0.006	 
1- 10 2- Adam 3- 1e-4 4- L1Loss 5- 2	Loss: 0.03843 test_loss: 0.03952 Dice: 0.673 IoU: 0.507	 
1- 100 2- Adam 3- 1e-4 4- L1Loss 5- 2	Loss: 0.01711 test_loss: 0.02171 Dice: 0.764 IoU: 0.618	 
1- 500 2- Adam 3- 1e-4 4- L1Loss 5- 2	Loss: 0.00597 test_loss: 0.01959 Dice: 0.780 IoU: 0.639	 
1- 10 2- Adam 3- 1e-4 4- BCEDiceLoss 5- 2	Loss: 0.65547 test_loss: 0.65824 Dice: 0.136 IoU: 0.073	 

	1- 100 2- Adam 3- 1e-4 4- BCEDiceLoss 5- 2	Loss: 0.65065 test_loss: 0.65017 Dice: 0.137 IoU: 0.074	 
U-Net Mejorada	1- 10 2- Adam 3- ReduceLR 4- MSELoss 5- 2	Loss: 0.02326 test_loss: 0.00454 Dice: 0.619 IoU: 0.448	 
	1- 100 2- Adam 3- ReduceLR 4- MSELoss 5- 2	Loss: 0.00353 test_loss: 0.00125 Dice: 0.748 IoU: 0.598	 
	1- 10 2- Adam 3- ReduceLR 4- L1Loss 5- 2	Loss: 0.04093 test_loss: 0.02539 Dice: 0.753 IoU: 0.604	 
	1- 100 2- Adam 3- ReduceLR 4- L1Loss 5- 2	Loss: 0.04041 test_loss: 0.02391 Dice: 0.734 IoU: 0.580	 
ResU-Net	1- 10 2- Adam 3- ReduceLR 4- MSELoss 5- 2	Loss: 0.00739 test_loss: 0.00234 Dice: 0.710 IoU: 0.550	 
	1- 100 2- Adam 3- ReduceLR 4- MSELoss 5- 2	Loss: 0.00326 test_loss: 0.00158 Dice: 0.715 IoU: 0.557	 

	1- 10 2- Adam 3- ReduceLR 4- L1Loss 5- 2	Loss: 0.07012 test_loss: 0.03487 Dice: 0.703 IoU: 0.542	
	1- 100 2- Adam 3- ReduceLR 4- L1Loss 5- 2	Loss: 0.03967 test_loss: 0.02535 Dice: 0.739 IoU: 0.587	

En la primera parte de la tabla 1, correspondiente a la U-Net Básica, claramente se puede ver como la función de pérdida BCEWithLogitsLoss es incapaz de converger a una predicción aceptable, sin importar la cantidad de épocas que se realicen. Es por esta razón que esta opción fue la primera descartada y no se probó para los otros dos modelos. También se combinó la función anterior con una conformada con el índice de Dice (1- Dice), resultando en una función de loss que si converge a una predicción aceptable, pero como su desempeño fue menor al de MSE y L1, también se descartó su aplicación en las otras redes. Finalmente, se decidió realizar los entrenamientos de 10 y 100 épocas con pérdida MSE y L1 en las otras dos redes y así poder elegir una función para la segunda etapa. Se obtuvo que, en un análisis rápido para ambas funciones, los resultados finales fueron prácticamente iguales, por lo que la selección pasó a basarse en el tiempo que requiere el entrenamiento con cada función. Aunque el tiempo de entrenamiento en ambos casos es muy similar, para la función L1 se necesita de 2 a 5 segundos más por época que para la función MSE. Aunque este valor es muy bajo se puede hacer muy significativo en entrenamientos de muchas épocas, por lo que se terminó descartando la función L1 y se decidió realizar los entrenamientos finales con MSE.

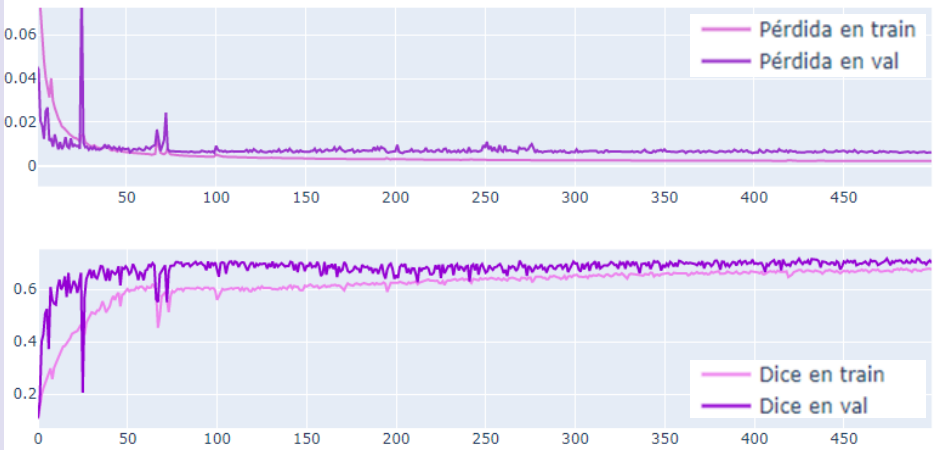
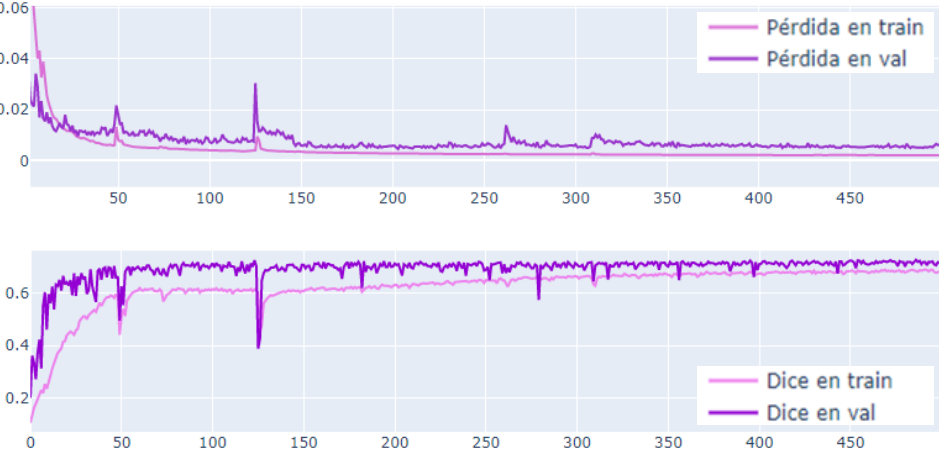
A modo anecdótico, queda mencionar que, se intentó realizar entrenamientos con la función de pérdida SharpLoss, detallada en el marco teórico, que fue creada específicamente para este tipo de aplicación, pero, a falta de la implementación original del paper y luego de varios intentos fallidos, se desistió de su aplicación en este trabajo. Con intentos fallidos refiere a que al codificar la ecuación descrita en el paper no se pudo lograr que ninguna de las tres redes realizara el descenso por el gradiente, el valor de pérdida siempre aparecía como NaN (Not a Number) y no realizaba ninguna mejora con el paso de las épocas.

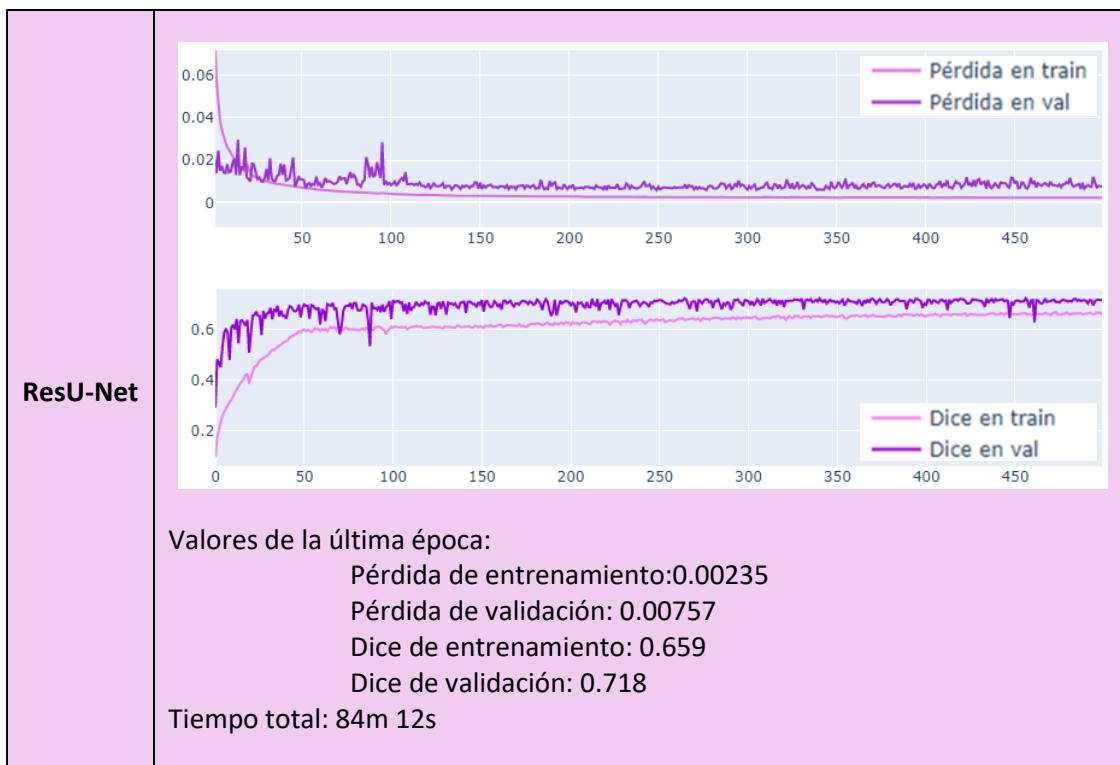
Con el panorama anterior y el acceso al clúster Mendieta se comenzó con la segunda etapa.

10.1.2 Segunda etapa de entrenamientos

En la siguiente tabla se pueden ver los gráficos que resultan de ir guardando los valores de métricas que se obtienen en cada época de entrenamiento. El primer gráfico tiene los valores de la función de pérdida y en el segundo los valores del índice de Dice con los datos de entrenamiento y validación (dataset de entrenamiento). Además, se puntualiza el valor final que se obtuvo para cada métrica en cada entrenamiento, para dar perspectiva de la magnitud de los valores con los que se está tratando y el tiempo que duró el entrenamiento.

Tabla 2: Métricas de la segunda etapa de entrenamiento. En el gráfico superior se encuentran los valores de pérdida de entrenamiento y validación, en el inferior los valores de índice de Dice de entrenamiento y validación

Red	Métricas del entrenamiento
<p>U-Net Básica</p>	 <p>Valores de la última época:</p> <ul style="list-style-type: none"> Pérdida de entrenamiento: 0.00221 Pérdida de validación: 0.00126 Dice de entrenamiento: 0.676 Dice de validación: 0.701 <p>Tiempo total: 125m 18s</p>
<p>U-Net Mejorada</p>	 <p>Valores de la última época:</p> <ul style="list-style-type: none"> Pérdida de entrenamiento: 0.00216 Pérdida de validación: 0.00626 Dice de entrenamiento: 0.684 Dice de validación: 0.717 <p>Tiempo total: 157m 26s</p>



En primer lugar, de la tabla anterior se puede concluir que un entrenamiento de 500 épocas fue más que suficiente, hasta se podría decir que excesivo, para entrenar cualquiera de los tres modelos, especialmente el primero que era notablemente más sencillo que los otros dos. Un detalle a tener en cuenta para las tres redes es que, aunque ahora la curva de índice de Dice tiene el valor promediado de todo el dataset para cada época, se ve muy inestable porque este número resulta de un cálculo muy sensible a las diferencias vóxel a vóxel entre las imágenes, por lo que la forma correcta de leer la segunda gráfica es considerar la tendencia global y no los valores puntuales, que son bastante erráticos.

Sobre el entrenamiento del primer modelo, viendo la evolución del índice de Dice, se puede decir que a las 150 épocas ya se estabiliza su “aprendizaje” alrededor del valor máximo que puede llegar a obtener, es decir, el mejor desempeño que puede alcanzar, aproximadamente un índice de Dice de 0.71 para los ejemplos de validación. El tiempo de entrenamiento que conlleva esta red con el segundo hardware es de alrededor de 14 segundos por época, durando poco más de dos horas para realizar el entrenamiento completo.

Continuando con la tercera red, ResU-Net, se puede ver que los resultados de su entrenamiento son similares a los de la primera. Esta vez, a partir de aproximadamente las 200 épocas se puede decir que el entrenamiento de la red llegó a su “techo”, con un valor de índice de Dice de alrededor de 0.71 para validación. El entrenamiento de esta red tarda cerca de 9 segundos por época, completándolo en aproximadamente una hora y media.

Finalmente, el mejor resultado que se obtuvo después de 500 épocas de entrenamiento fue con el segundo modelo, la U-Net Mejorada, lo que era previsible porque desde un principio presenta una cantidad de parámetros entrenables muy superior respecto a los otros dos modelos. En este entrenamiento también le toma 200 épocas estabilizar la tendencia del índice de Dice, ligeramente por arriba de la recta del 0.72. Como era de esperarse, a esta red le toma más tiempo entrenar, alrededor de 18 segundos por época, por lo que el entrenamiento total duró dos horas y media minutos.

10.2 Comparación con la planificación tradicional

10.2.1 Análisis de resultados para ejemplos particulares del set de testeo

Para la valoración de las predicciones del dataset de testeo obtenidas con las redes entrenadas anteriormente, se realizó una serie de gráficos que se detallaron en la sección [Testeos](#). Estos representan valores relevantes como porcentajes de dosis, la cantidad de vóxeles y el índice de Dice, y permiten comparar los resultados predichos con el objetivo de distribución de dosis realizado manualmente para cada paciente, con planificación tradicional en el Centro.

En la Jupyter Notebook, los gráficos realizados primero se muestran para cada uno de los 30 ejemplos y luego al final del código se puede encontrar un gráfico final que considera los valores promedios del Índice de Dice.

Primero se muestra un corte transversal del ejemplo particular y sus predicciones que se van a analizar. Por ejemplo, se muestra en la ilustración 44, el ejemplo en la posición 16 en las carpetas de set de testeo y las carpetas de las predicciones hechas con cada red.

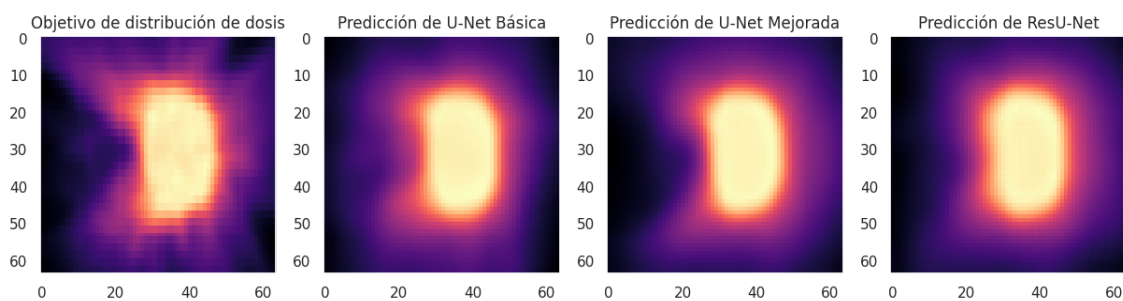


Ilustración 44: Corte transversal aproximadamente a la mitad de la distribución de dosis y las predicciones de cada red para el volumen en la posición número 16 dentro las carpetas

Luego se genera el gráfico con los índices de Dice para cada intervalo porcentual de dosis de este ejemplo particular, que se muestra en la ilustración 45 a continuación. La línea azul representa el índice de Dice objetivo, es decir, el que resulta de comparar el objetivo con sí mismo, y es por esto que su valor es 1 en todos los puntos de los intervalos porcentuales de dosis relevantes para este ejemplo. Para este caso en particular, se puede ver como la predicción generada con la segunda red (línea rosada) tiende a ser superior a la de la primera (línea morada) y a la tercera (línea lila), la cual es ligeramente peor en los valores más bajos y los valores más altos. Además, se ve una clara tendencia de las tres redes a generar más vóxeles con el valor de dosis total y el 0.1 del mismo, ya que hay dos picos notables en 10% y 100%. Otro dato no menor que aporta la gráfica es que las tres redes fracasan cuando se trata de predecir dosis mayor que el total, es decir de valor 110%. Con respecto a la primera red, se tiene un comportamiento bastante particular, porque para valores bajos es superada por la tercera red, se convierte en la de peor desempeño, pero para valores altos prácticamente iguala a la mejor.

Aunque en esta sección se habla particularmente del ejemplo 16, luego de correr el código de este gráfico para todos los ejemplos, se puede decir que los resultados de este caso particular son extrapolables a casi todos los ejemplos del dataset de testeo, particularmente en la característica de la superioridad de la segunda red, lo que se confirma en la última gráfica, que representa el promedio del dataset.

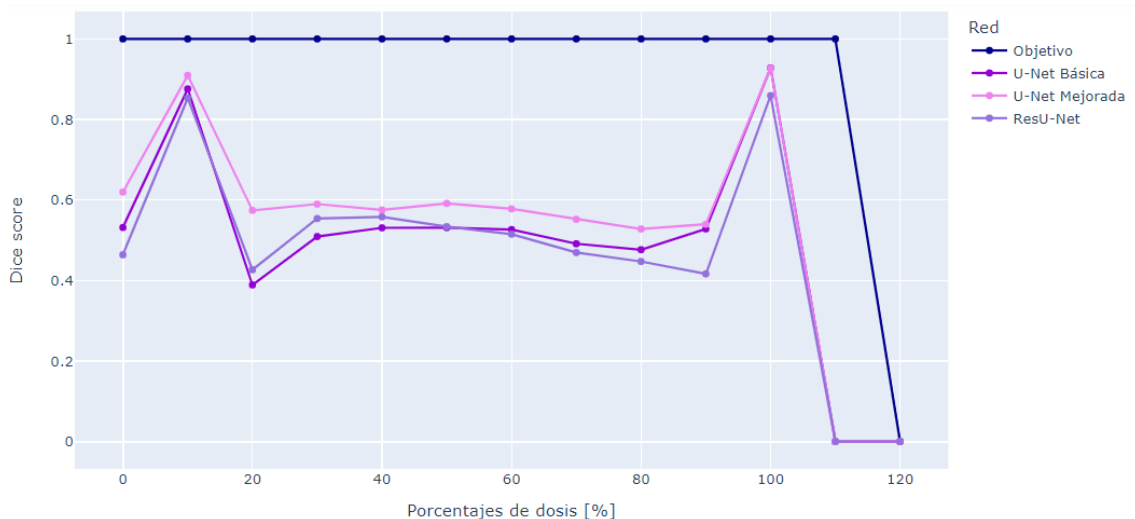


Ilustración 45: Gráfico de índice de Dice para los intervalos porcentuales de dosis del ejemplo 16

Luego se continua con la gráfica del one-hot encoding de los órganos, para saber en qué capa binaria se representa cada uno. En el ejemplo 16, que se muestra en la ilustración 46, se tiene al fondo en la capa 0, la vejiga en la capa 1 (a la altura de corte escogida generalmente no se ve), el recto en la capa 2 y la próstata en la capa 3. Esta información extrae de forma manual para realizar el gráfico de los histogramas.

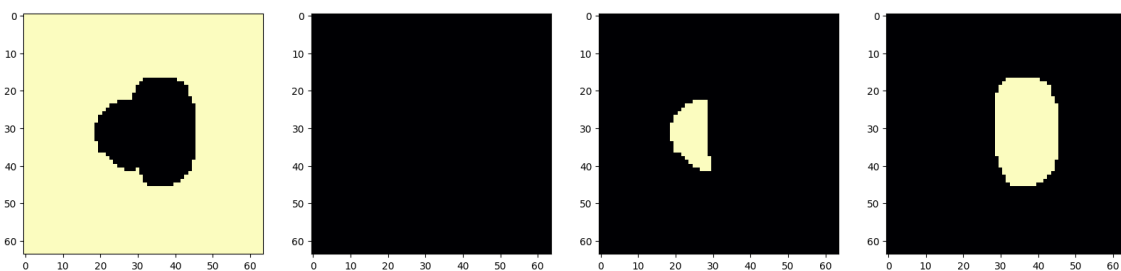


Ilustración 46: One-hot encoding de los órganos del ejemplo 16 del set de prueba. De izquierda a derecha se tiene la máscara correspondiente al fondo, luego a la vejiga (que a esta altura de corte no presenta ningún pixel), el recto y al último la próstata.

Continuando con la matriz de histogramas, en este gráfico se muestra de forma separada para cada órgano su distribución de dosis (cantidad de vóxeles que tienen cada intervalo porcentual de dosis), como se muestra en la ilustración 47 (para el ejemplo 16 sobre el cual se venía trabajando).

En la primera y tercera fila se puede ver como el pico máximo de todos los histogramas se encuentra a la izquierda del 0.2 de dosis, es decir, que todos los vóxeles de dicho pico reciben una dosis menor al 20% del total. A partir de esto se puede confirmar que las dosis se extrajeron de forma correcta, ya que la primera y tercera fila corresponden a la distribución de dosis en recto y la vejiga, que son los OAR que deben recibir la menor cantidad de dosis posible. Por otro lado, la fila del medio corresponde a la dosis en la próstata (PTV), por lo que es correcto que todos los histogramas sean un solo pico aproximadamente sobre el valor 1.00, ya que idealmente todo el tejido (todos los vóxeles) debe recibir el 100% de la dosis prescrita.

Siempre se graficaron los datos del objetivo en azul, para comparar con este las predicciones. Comenzando por la distribución del PTV, se puede ver que en los histogramas de la predicción de la red U-Net Mejorada (rosado) y el de la U-Net Básica (morado) se tiene el pico

máximo aproximadamente sobre el 1.00 de dosis, siendo las que más se acerca al objetivo, que presenta dicho pico en un valor intermedio entre 1.00 y 1.05. Por otro lado, la predicción de la ResU-Net (lila) tienen el pico del diagrama de PTV en un valor a la izquierda del 1.00, aproximadamente en 0.96. Lo anterior indica que prácticamente ningún vóxel recibe el 100% dosis en la predicción del tercer modelo. También hay que mencionar que a su vez la segunda predicción tiene más píxeles concentrados en su valor máximo (presenta un pico más alto y angosto), es decir, que le está dando a una mayor cantidad de vóxeles del PTV, una dosis alta y menos variada (un rango menor de valores).

Para el caso de los histogramas correspondientes a la vejiga se tiene más homogeneidad entre las predicciones, particularmente las tres mantienen cantidades muy bajas de vóxeles para dosis mayores al 20%.

Finalmente, en las distribuciones del recto se puede ver como las predicciones son más parecidas entre sí que al objetivo. Las tres atribuyen los valores más altos de dosis a una mayor cantidad de vóxeles, teniendo un último pico sobre el 0.9 de dosis, siendo esto más marcado en la primera y la tercera predicción. A pesar de esto, las tres predicciones muestran que el pico más alto está casi sobre el 0% de dosis, lo que es un resultado aceptable. Igualmente se tiene que resaltar que este es un defecto de las predicciones conceptualmente importante, ya que el recto es un OAR y mientras más dosis altas reciba peores serán los efectos secundarios y las consecuencias para el paciente.

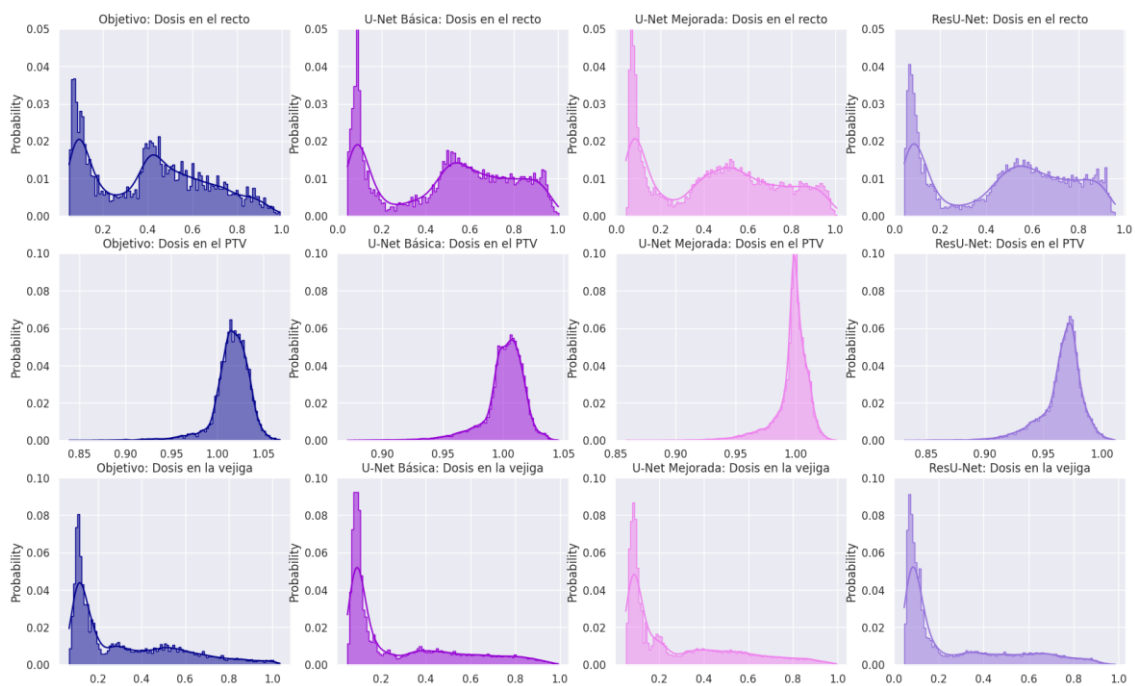


Ilustración 47: Histogramas de las distribuciones de las dosis relativas vs la cantidad relativa de píxeles del objetivo (primera columna) y las predicciones para cada órgano (desde arriba recto, PTV y vejiga) para el ejemplo 16

También se decidió graficar estos histogramas juntos para poder ver las diferencias más gruesas entre los resultados y no tanto los detalles como se analizó anteriormente. A simple vista lo que más resalta es la discrepancia de las distribuciones en el PTV, mostrando lo que se mencionaba en el punto anterior, que la primera red es la que más se acerca y la tercera es la

de peor desempeño, aunque las tres dejan mucho que desear. En el polo opuesto se tiene las distribuciones en la vejiga que son prácticamente iguales entre sí. Y finalmente en el recto, se puede ver más claramente la diferencia entre la línea azul y las predicciones, que le atribuyen las altas dosis cercanas al 100% a prácticamente el doble de vóxeles (0.005 contra el 0.01 del total de vóxeles para el 90% de dosis).

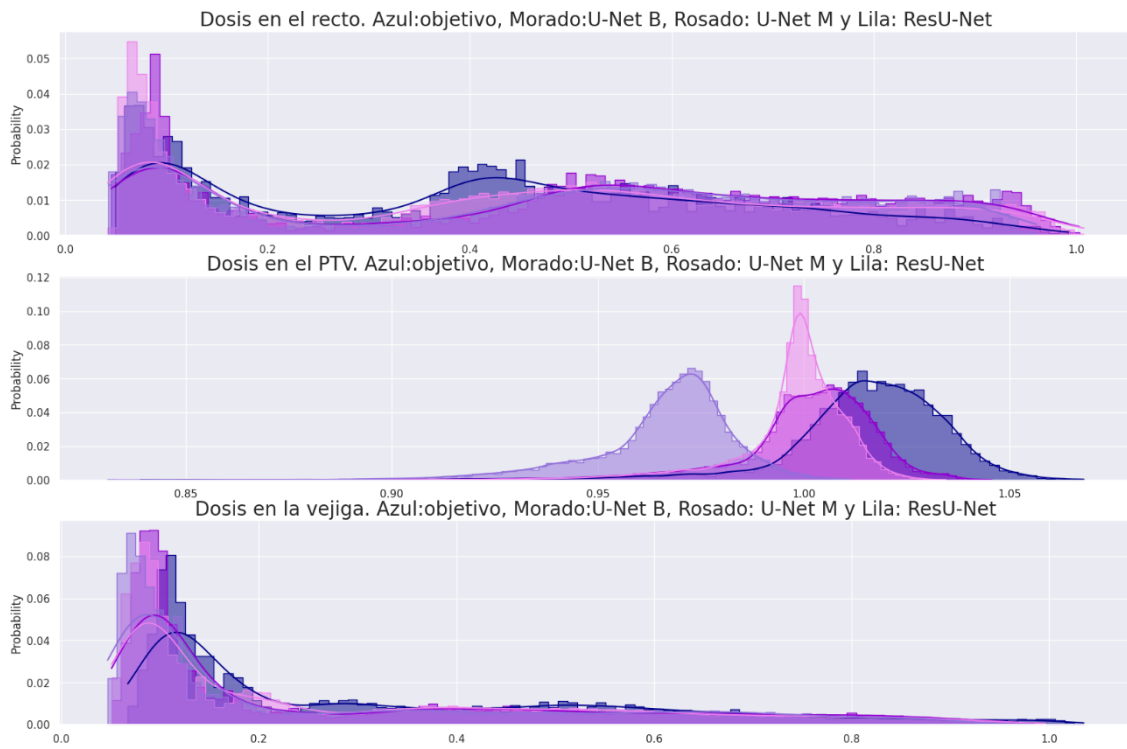


Ilustración 48: Histogramas superpuestos de las predicciones y el objetivo para cada órgano del ejemplo 16

Por último, se realizaron las curvas de DVH acumulado para cada uno de estos histogramas y se presentaron todos juntos en un gráfico para poder comparar que tan aproximadas son las curvas de las predicciones a las del objetivo, para cada órgano (Ilustración 49). En esta gráfica se tiene en el eje de las abscisas la dosis porcentual y en el eje de las ordenadas la cantidad relativa de vóxeles de cada órgano. Las curvas forman tres grupos de cuatro curvas: las que tienen línea punteada corresponden a las distribuciones en el PTV, las líneas continuas a las distribuciones en el recto y las líneas interrumpidas a las dosis en la vejiga. En cada grupo, la curva azul es la correspondiente al DVH acumulado objetivo, es decir, se busca que las otras tres curvas sean lo más parecidas posibles a la curva azul. Las curvas violetas representan los DVH acumulado resultante de la predicción conseguida con la red U-Net Básica, las curvas rosadas con la red U-Net Mejorada y las curvas lilas con la red ResU-Net.

La forma de leer esta gráfica es: por lo menos el X% de dosis se le aplica al Y del total del tejido del órgano, es por esto que para el PTV la curva se mantiene en el 1 del tejido hasta aproximadamente el 90% (corresponde que al 100% del tejido de la próstata se le aplique el 90% de la dosis o más). En base a lo anteriormente explicado, en una gráfica con resultados ideales, la curva correspondiente al PTV sería un escalón que se mantiene en 1 hasta el 100% de la dosis y las curvas para cualquier OAR tendría forma de "L" (lo más cercana al eje de las ordenadas que se pueda) así de esta forma al 100% del tejido se le aplica cerca del 0% de la dosis total. Esta idealización es geoméricamente imposible, ya que, los rayos de radiación necesariamente deben atravesar tejido en riesgo para depositar la dosis necesaria para tratar el objetivo o PTV.

Como se venía viendo en los gráficos anteriores, la tendencia es que la curva rosada, de la predicción realizada con el segundo modelo, sea la más aproximada al objetivo, aunque en varios puntos la curva de la primera red también es muy satisfactoria. Estas predicciones, en el PTV, discrepan con el objetivo únicamente en los valores de dosis mayores al 100%, mientras que la del tercer modelo llega a entregar el 100% de la dosis a menos del 0.05 de los vóxeles. Por otro lado, se ve más claramente como todas las redes le adjudican dosis más altas al tejido del recto, teniendo la separación máxima cerca del 45% de la dosis donde las predicciones de la primera y la tercera red le dan dicha dosis a un 20% más del tejido, siendo esto en general peor para la primera predicción (más del doble del tejido recibe el 80% de la dosis, estando el objetivo por debajo del 0.1 y estas predicciones cerca del 0.2). También se ve como la cantidad de dosis en la vejiga predicha por las tres redes es menor que la del objetivo, lo que también se puede correlacionar geoméricamente con la falta de dosis en el PTV.

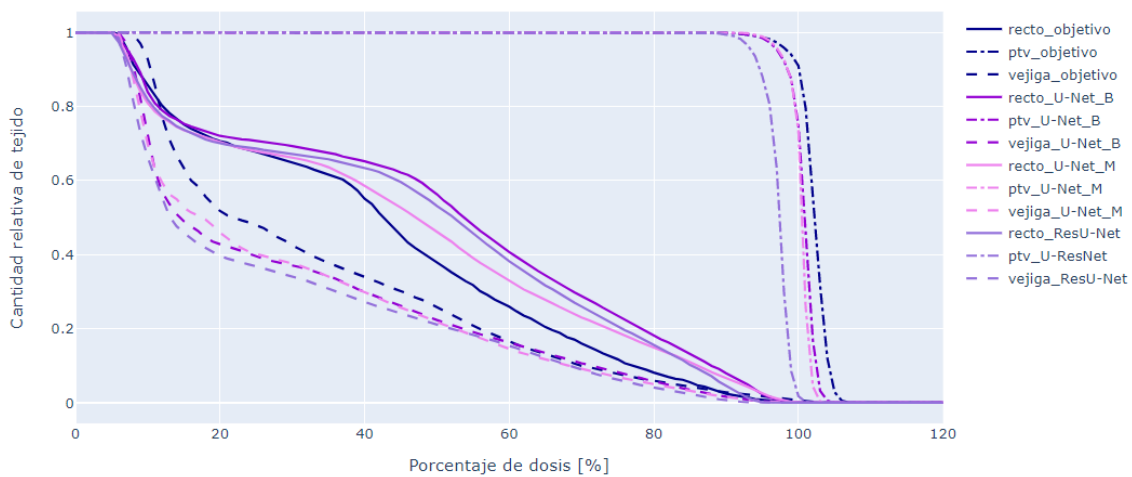


Ilustración 49: Gráfica DVH acumulado de todas las estructuras de todas las predicciones y el objetivo para el ejemplo 16

Vale la pena también analizar los resultados conseguidos para el ejemplo 7 (Ilustración 50), en el cual la forma de la próstata es atípica (más aplanada en relación a las otras) por lo que las redes vieron mucho más complicada la predicción de su distribución de dosis (Ilustración 51).

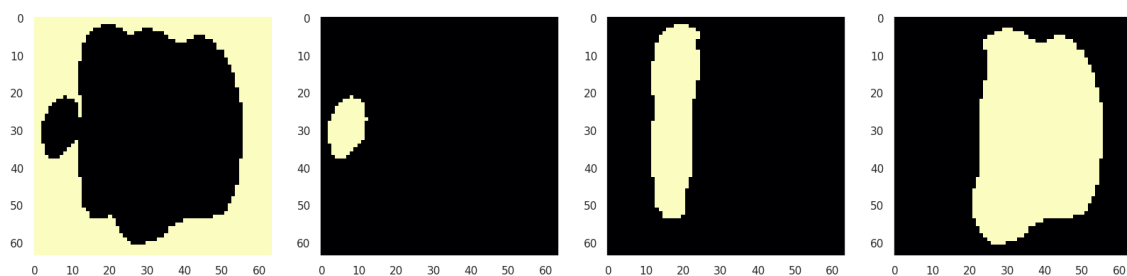


Ilustración 50: Corte transversal del one-hot encoding de los órganos del ejemplo 7 del set de testeo, de izquierda a derecha el fondo, el recto, la próstata y la vejiga

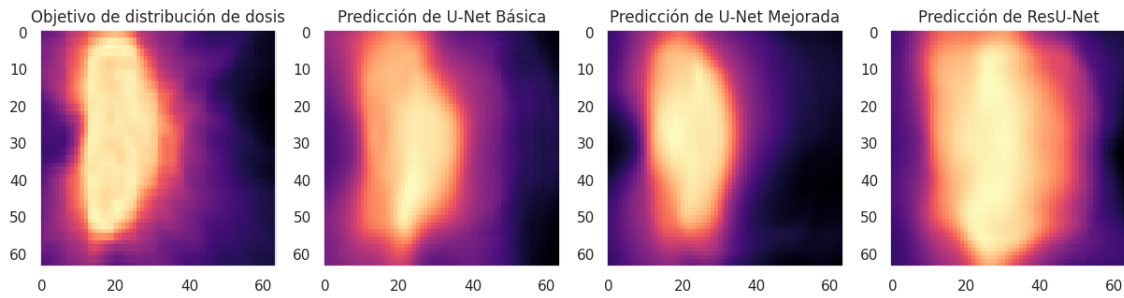


Ilustración 51: Corte transversal de la distribución de dosis objetivo y las predicciones de cada red para el ejemplo 7 del set de testeo

En la gráfica de los índices de Dice (Ilustración 52) se puede ver claramente como en general todas las mediciones empeoraron respecto al ejemplo 16, especialmente las del tercer modelo. Ahora los valores de Dice se encuentran prácticamente todos por debajo de 0.4. El único intervalo porcentual de dosis que pareciera mejorar es el del 0% de dosis, pero en realidad esto se puede concebir como una consecuencia de que ahora globalmente hay muchos más vóxeles con ese valor de dosis, que en realidad correspondían a otros intervalos, entonces ahora es mayor la probabilidad de que a un vóxel que realmente le corresponde el 0% de dosis se le adjudique ese valor. También corresponde destacar que, el segundo modelo presenta un índice de Dice distinto de cero para el intervalo porcentual del 110% de dosis, aunque sea un valor bastante bajo.

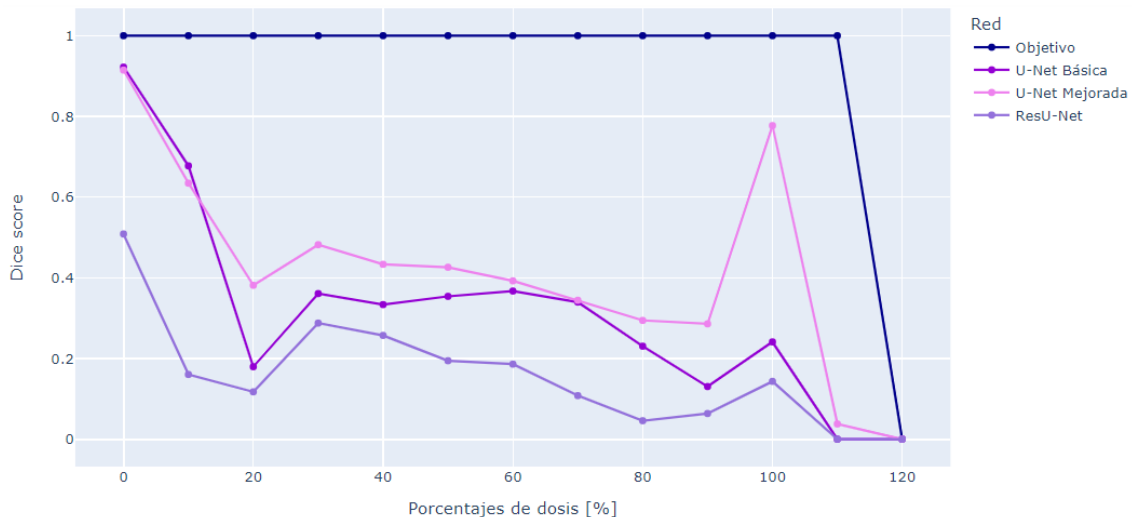


Ilustración 52: Gráfico de índice de Dice para los intervalos porcentuales de dosis del ejemplo 7

En los histogramas superpuestos (Ilustración 53), lo primero que resalta es la gran similitud que presentan todas las distribuciones correspondientes a la tercera red, a pesar de haber tenido tan malos resultados en la gráfica anterior. Aun así, se puede destacar problemas como que para el PTV el pico del histograma se encuentra sobre 0.9 en vez de 1, es decir que nuevamente no se le está entregando la dosis necesaria al tejido maligno. Además, en el histograma de la vejiga se puede ver como la tercera red sobresale en los intervalos de 80% y 90%, entregando grandes dosis a tejido en riesgo.

Por otra parte, la primera y la segunda red consiguen distribuciones bastante similares al objetivo para el recto y la vejiga, aunque la segunda red también presenta el mismo problema de entregar dosis altas a más vóxeles de un OAR. Con respecto al PTV, la primera red también “se queda corta” en lo que refiere a entregar el 100% de la radiación prescrita. Mientras tanto,

la segunda red muestra resultados sumamente satisfactorios, casi igualando el pico que presenta el objetivo a la derecha del 100%.

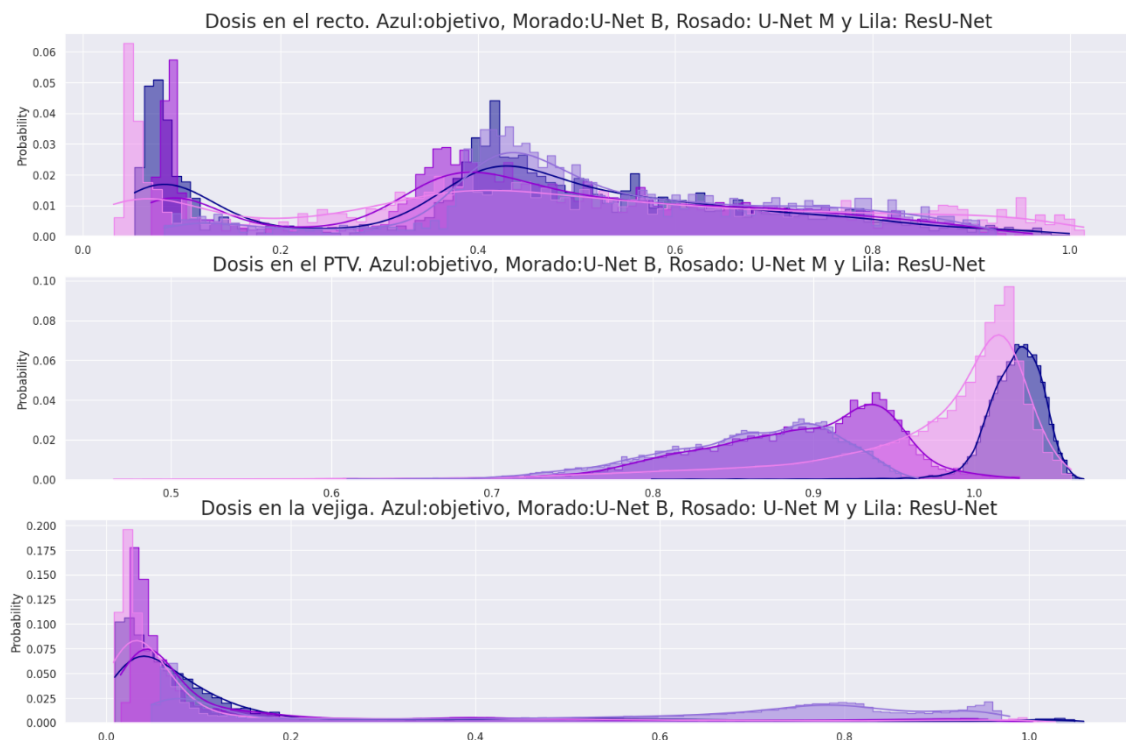


Ilustración 53: Histogramas superpuestos de las predicciones y el objetivo para cada órgano del ejemplo 7

Cerrando el análisis de este ejemplo, en el DVH acumulado (Ilustración 54), se muestra una gran discrepancia general de todas las curvas. Comenzando por el PTV, ninguna de las curvas logra aproximarse al 100% de dosis y comienzan a decaer mucho más suavemente que el objetivo, es decir que la distribución de dosis es mucho más variada (mientras más brusca la caída del “escalón”, menor la diferencia de dosis entre los vóxeles). Por ejemplo, la curva correspondiente a la segunda predicción tiene comienzo a decaer aproximadamente en el 70% de dosis, pero si le otorga el 100% de dosis al 50% del tejido.

Por otro lado, se puede decir que la distribución en el recto es bastante aceptable para todas las predicciones, siempre cumpliendo la tendencia de que el tercer modelo tenga los peores resultados y el segundo los mejores, y notando que desde un principio el objetivo deposita bastante dosis en este OAR.

Por último, en la vejiga, esta vez no ocurre el fenómeno de que las predicciones adjudiquen menos dosis, si no que, por lo contrario, se encuentra la mayor discrepancia. Lo que más resalta es que la curva correspondiente a la tercera predicción está muy alejada de las otras y en vez de intentar asemejarse a una “L” tiende a ser un escalón, como si fuera para un PTV. Esto se entiende viendo la ilustración 51, donde la distribución de dosis de la tercera predicción asemeja más la silueta de la vejiga. Pareciera que la tercera red confunde este OAR con el PTV (probablemente porque la forma se asemeja más a lo que se usó para entrenar) e intenta atribuir la mayor cantidad de dosis a este órgano.

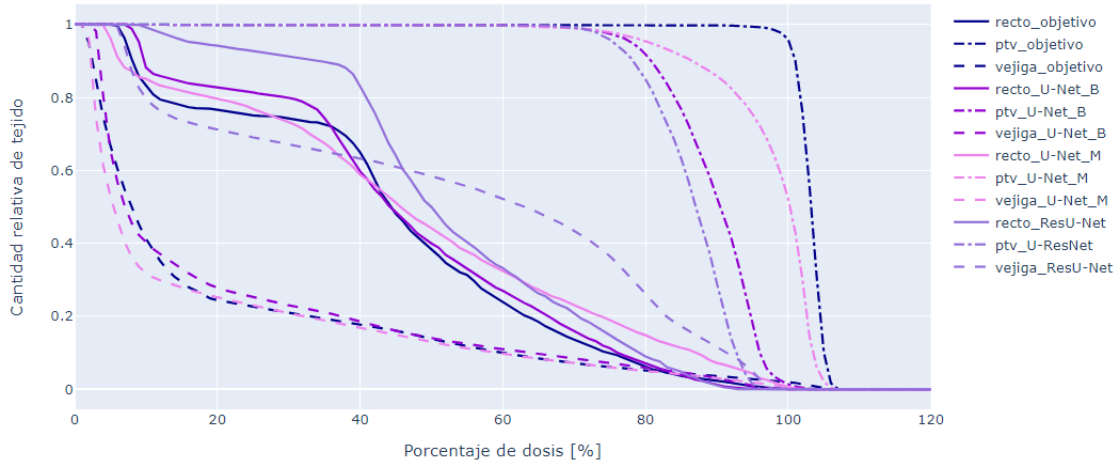


Ilustración 54: Gráfica DVH acumulado de todas las estructuras para todas las predicciones y el objetivo para el ejemplo 7

Un problema en las predicciones similar al anterior se da con el ejemplo 24 (Ilustración 55 y 56), que presenta la anomalía de tener la próstata muy adelante, esto se nota claramente al ver que en el corte 24 (Ilustración 57) de 64 ya no se puede ver su silueta. Además, pareciera que, en la anatomía de este paciente en particular, el recto tiene mucho contacto con el PTV, entonces las redes tienen complicaciones para distinguirlos y tienden a entregarle mucha dosis a dicho OAR. Esto se ve en la ilustración 56 que muestra las distribuciones de dosis y muestra cómo estas agregan una estructura a la izquierda del PTV, que no se encuentra en la distribución de dosis del objetivo.

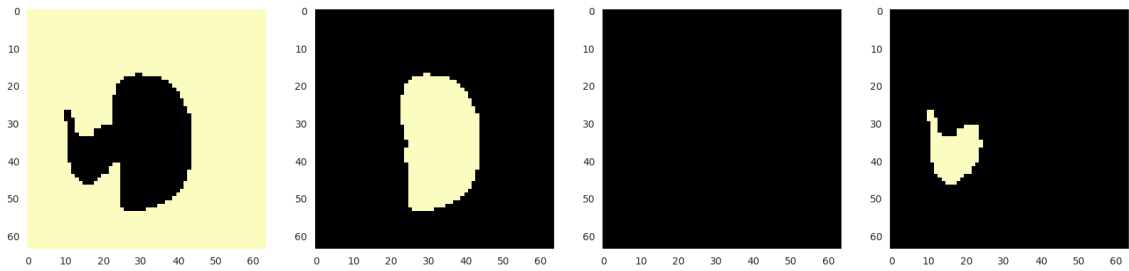


Ilustración 55: Corte transversal del one-hot encoding del ejemplo 24 a la altura de corte 20 de 64

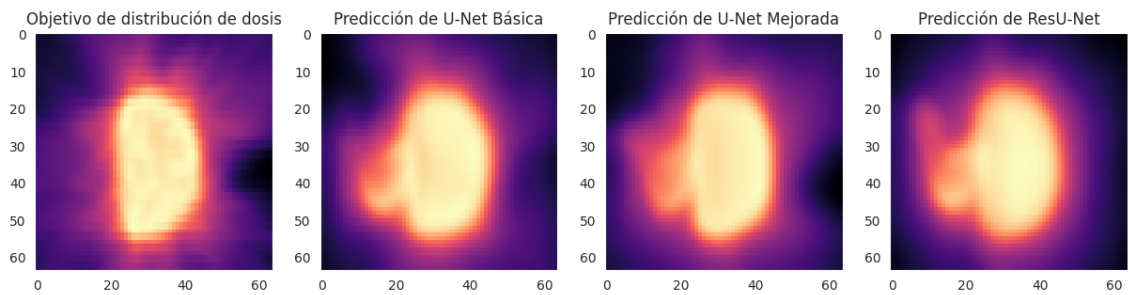


Ilustración 56: Corte transversal de la distribución de dosis objetivo y las predicciones de cada red para el ejemplo 24 del set de testeo, a la altura de corte 20 de 64

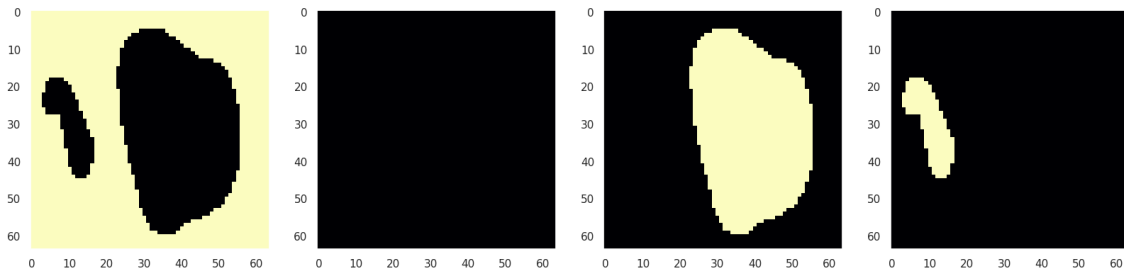


Ilustración 57: Corte transversal del one-hot encoding del ejemplo 24 a la altura de corte 24 de 64

Este problema con el recto queda más claro en los histogramas correspondientes a este órgano, ilustración 58, donde se ve como la tercera red genera una distribución más similar a un PTV que a un OAR. Para la predicción del primer modelo también se tiene un histograma bastante atípico en el que la distribución es prácticamente igual para todos los intervalos porcentuales de dosis. Y cabe destacar como siempre que el segundo modelo es el que más se acerca al objetivo.

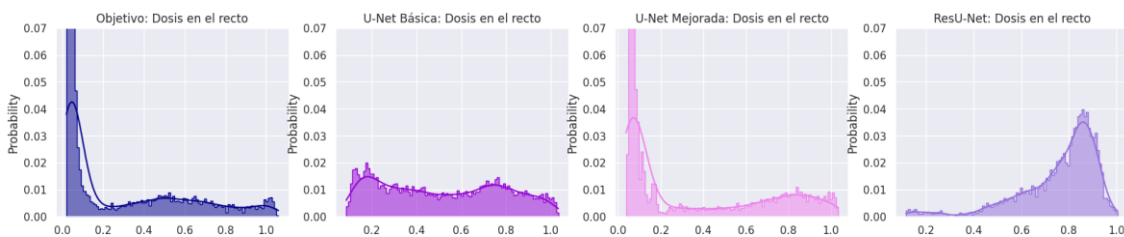


Ilustración 58: Histogramas de distribución de dosis del objetivo y las predicciones para el recto del ejemplo 24

En la gráfica de DVH acumulado se tiene el resultado esperado, la curva de la tercera predicción para el recto tiene forma de escalón, prácticamente una curva de PTV. Además, la curva de la primera predicción también discrepa bastante del objetivo y como se viene viendo, la curva de la segunda predicción es la más aceptable para todos los órganos.

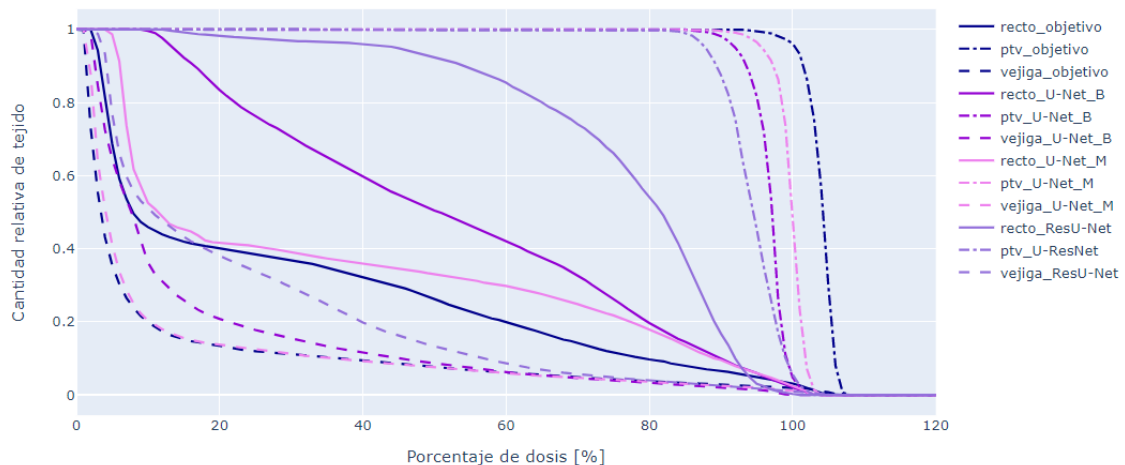


Ilustración 59: Gráfica DVH acumulado de todas las estructuras para todas las predicciones y el objetivo para el ejemplo 24

Finalmente, para terminar con este análisis de casos puntuales, también vale la pena destacar el ejemplo número 6 (Ilustración 60 y 61), que es el que genera los mejores resultados de predicción. Este ejemplo tiene una forma de próstata grande, ovalada y ubicada en una

posición central del volumen (como se puede ver en la ilustración 60 que es del corte transversal número 32 de 64), lo que pareciera que les permite a las redes realizar predicciones con mayor exactitud.

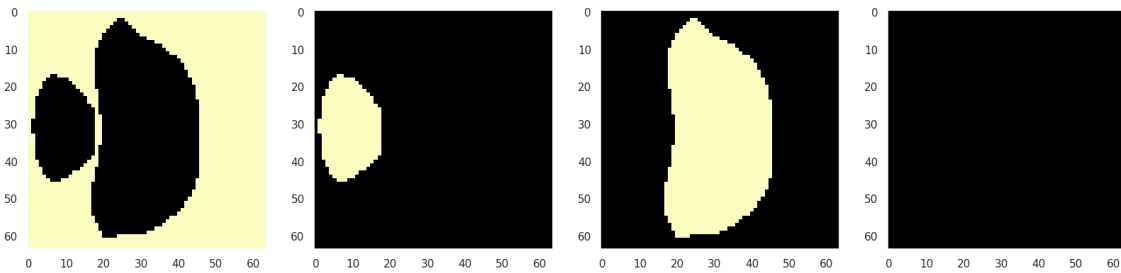


Ilustración 60: Corte transversal del one-hot encoding del ejemplo 6 a la altura de corte 32 de 64

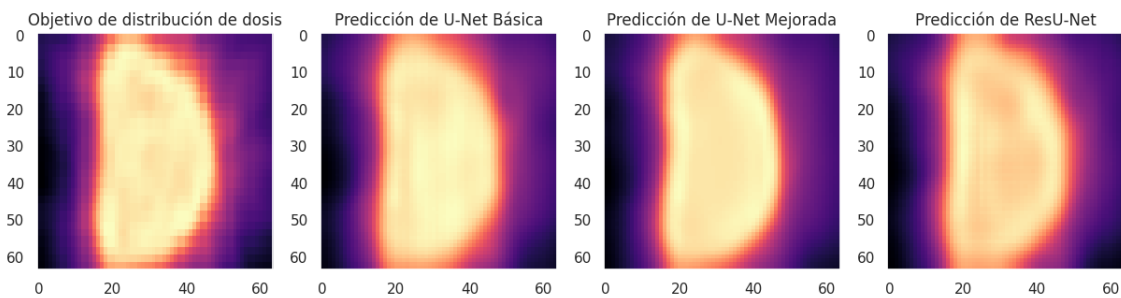


Ilustración 61: Corte transversal de la distribución de dosis objetivo y las predicciones de cada red para el ejemplo 6 del set de testeo, a la altura de corte 32 de 64

En la gráfica de índices de Dice resalta mucho como prácticamente todos los índices correspondientes a la primera y segunda predicción se encuentran por arriba del 0.6 y muchos están cerca de 0.9, siendo que esta vez es la primera red la que obtiene los resultados más constantes y satisfactorios. Como es habitual, la tercera red presenta muchas complicaciones especialmente en los en los valores más altos de dosis, donde fracasa rotundamente. También hay destacar como las dos primeras predicciones logran obtener algunos valores pertenecientes al intervalo porcentual del 110% de dosis.

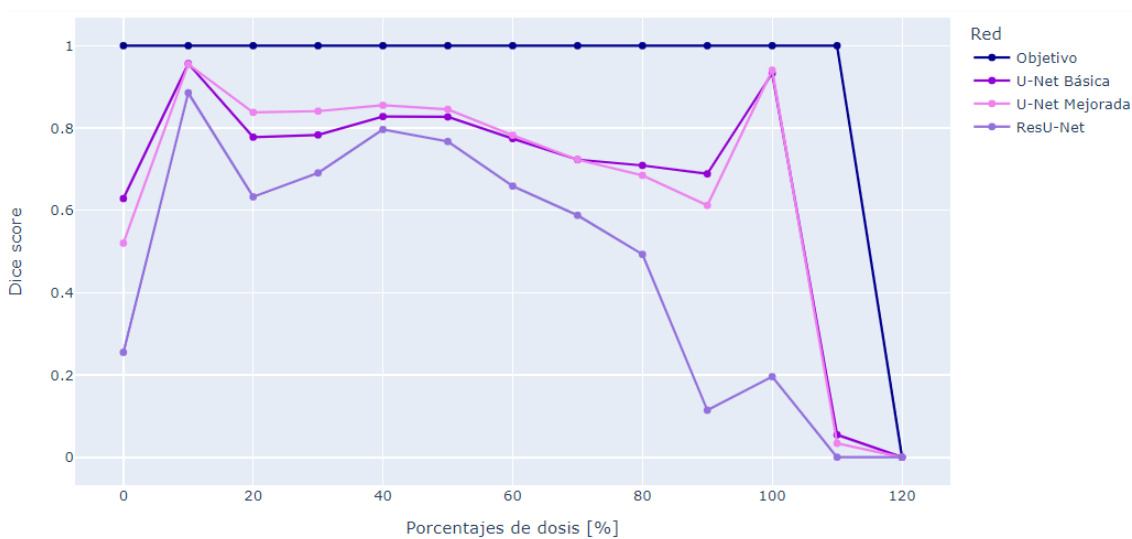


Ilustración 62: Gráfico de índice de Dice para los intervalos porcentuales de dosis del ejemplo 6

En los histogramas, resulta muy impresionante ver como la primera predicción emula casi perfectamente al objetivo. A diferencia de todos los casos anteriores, para este ejemplo se

logran predecir casi perfectamente, todos los vóxeles del PTV con valores de dosis mayores al 100%, con la primera red. Además, todas las distribuciones también son prácticamente idénticas para el recto y en la vejiga únicamente resalta que la tercera predicción presenta su pico a la derecha del 0.2.

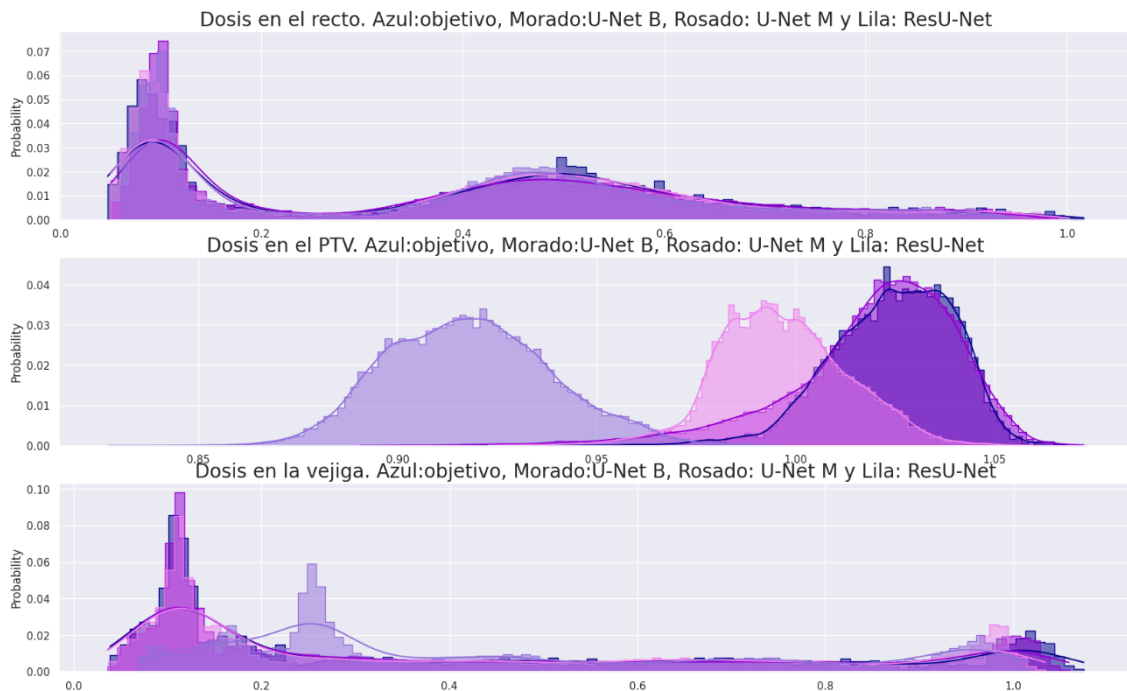


Ilustración 63: Histogramas superpuestos de las predicciones y el objetivo para cada órgano del ejemplo 6

Finalmente, en el DVH acumulado se puede ver como la primera predicción coincide perfectamente con el PTV del objetivo hasta casi el 80% del tejido, y en los dos OAR las dos primeras redes también obtienen resultados casi perfectos. Por otro lado, como se venía viendo, la predicción de la tercera red no llega a entregar la dosis completa al PTV y discrepa mucho en la curva de la vejiga en valores de dosis entre 10% y 50%, entregando dosis mucho mayores al OAR.

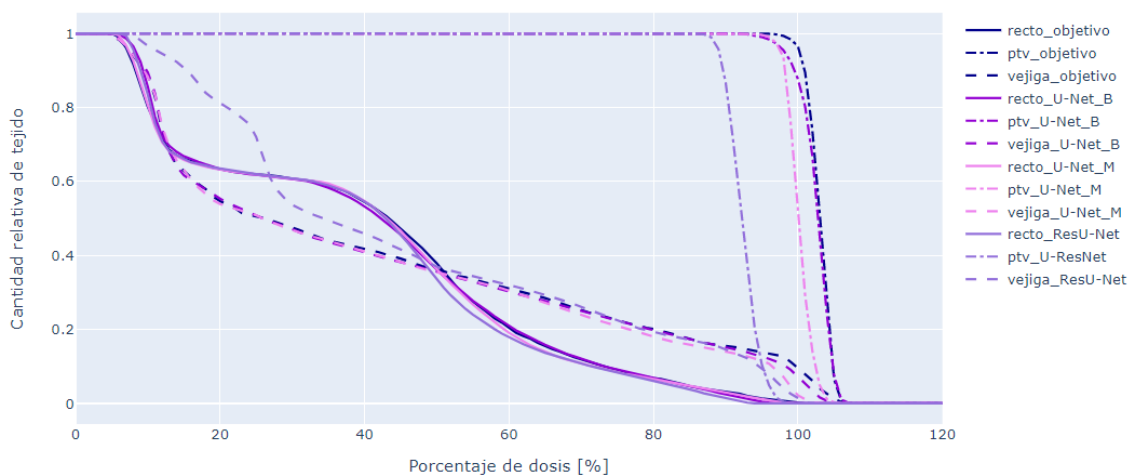


Ilustración 64: Gráfica DVH acumulado de todas las estructuras para todas las predicciones y el objetivo del ejemplo 6

10.2.2 Análisis global del desempeño de las predicciones para todo el set de testeo

Como último gráfico, se generó la ilustración de los índices de Dice promediados del conjunto de los 30 ejemplos del set de testeo (Ilustración 65).

Recordando que lo que se busca en esta gráfica es que todos los puntos sean lo más cercano a 1 posible, se puede decir que la segunda red es la que da las mejores predicciones, ya que la curva rosada se encuentra por arriba de las otras dos en todos los intervalos porcentuales de dosis. Las predicciones de la segunda red no solo son mejores que las otras dos, sino que también tiene un índice de Dice promedio siempre por arriba de la recta del 0.5. De igual forma se puede decir que las peores predicciones son las entregadas por la tercera red, ya que se encuentra en un valor menor en todas las dosis. También se puede ver como ninguna red es capaz de predecir consistentemente valores cercanos al 110%, aunque hay que tener en cuenta que muchos de los ejemplos que se usaron para entrenar y del mismo dataset de testeo directamente no presentan vóxeles con dosis de esos valores.

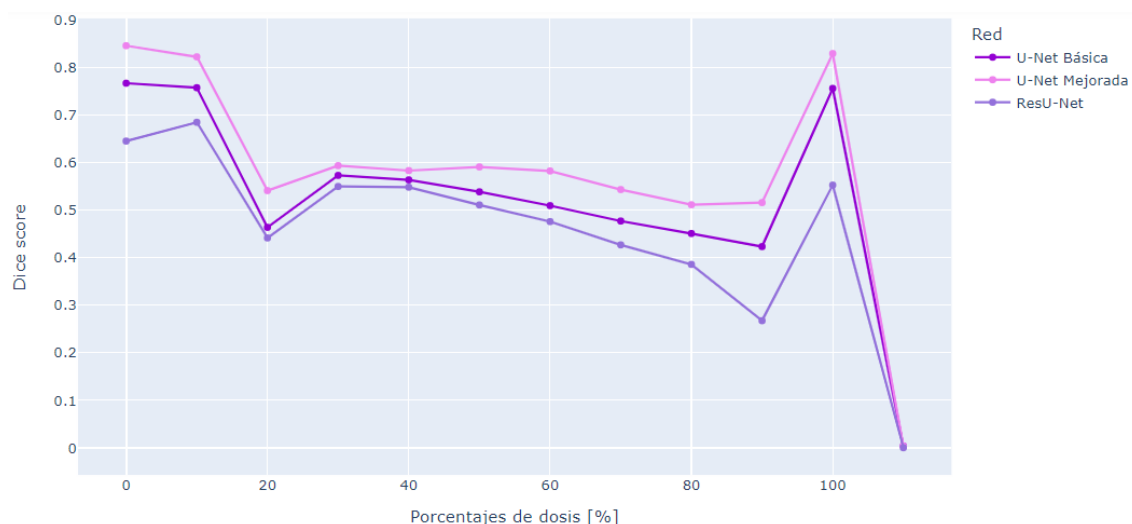


Ilustración 65: Gráfico de índice de Dice promedio para los intervalos porcentuales de dosis de los 30 ejemplos del set de testeo

A modo de conclusión final del desempeño de las redes, a nivel global en todo el set de testeo, la U-Net Mejorada es indiscutiblemente superior a las otras dos. Esto resalta especialmente en los intervalos porcentuales de dosis intermedios, entre 20% y 90%. Por otro lado, el ejemplo 6 demostró que para algunos casos la U-Net Básica logra ajustarse significativamente mejor al problema, lo que contrasta con las grandes carencias protegiendo los OAR que muestra con otros ejemplos, como en el 16. Entonces de estas dos redes se puede decir que la primera tiene picos muy altos, pero también picos muy bajos de desempeño, mientras que la segunda muestra una gran constancia de buenos resultados para un grupo ejemplo muy diversos, como los que tiene el set de testeo. Y, por último, las predicciones logradas con la ResU-Net únicamente son aceptables para el intervalo porcentual correspondiente al 0% de dosis, siendo para todos los otros casos regular, especialmente para las dosis más altas.

11 Discusiones

Los recientes avances tecnológicos han permitido la introducción de la IA como herramienta para mejorar varias áreas de la medicina, siendo la radio-oncología una de ellas, aplicándolas a diversas tareas en la ruta de la atención del paciente con cáncer. Si bien en la literatura relevante al tema se han planteado una serie de aplicaciones prometedoras, actualmente todavía queda afrontar los desafíos fundamentales de la aplicación en clínica cotidiana.

Un proceso de RT tradicional comienza con las imágenes simulación, luego el contorno de la estructura del órgano y las etapas de planificación del tratamiento para obtener la distribución de la dosis y evaluar la planificación. En este proyecto, se construyeron, evaluaron y compararon tres redes convolucionales volumétricas para predicción de distribución de dosis de SBRT con RapidArc™ para el cáncer de próstata, que parten de la arquitectura U-Net. Esta línea de investigación tiene como fin último omitir las etapas de planificación y evaluación para obtener la distribución de dosis. De esta manera se puede reducir las iteraciones y el tiempo de planificación.

Para este proyecto, el proceso de planificación incluyó dos arcos completos de VMAT y PTV con 36,25 Gy, que se usaron en los 135 pacientes para administrar radiación a la próstata. Esto le dio uniformidad al dataset, pero lo más seguro es que si se intenta implementar los modelos para predecir la distribución de dosis para pacientes tratados con otros protocolos, no se obtengan buenos resultados, incluso para pacientes con cáncer de próstata. También hay que tener en cuenta que todos los datos utilizados se obtuvieron de un solo centro de RT, lo que conlleva algún tipo de sesgo en los datos. La aplicación de los modelos con datos de otros centros probablemente no llegue a alcanzar el rendimiento obtenido, además que las características particulares pueden exigir otro tipo de preprocesamiento de los datos de entrada.

Un punto importante a tener en cuenta cuando se trabaja con DL, es que la cantidad de ejemplos usados como datos de entrenamiento influye en la precisión de la predicción en los algoritmos. En otras palabras, los resultados obtenidos en el presente proyecto no son el límite de capacidad de las tecnologías aplicadas, si no que, podrían mejorarse mucho aumentando el conjunto de datos de entrenamiento y su variedad.

Como se mencionaba en la sección anterior, la U-Net Mejorada dio los mejores resultados predictivos, mientras que el modelo ResU-Net dio los peores. Esto por supuesto, no descarta la gran utilidad de los métodos de transfer learning. Podría ser fructífero probar con otros tipos de redes previamente entrenadas para la ruta de codificación. Además, un detalle no menor respecto a estas redes preentrenadas, es qué tipo de imágenes se utilizaron para dicho entrenamiento. Si bien es cierto que en general se utilizan dataset masivos que buscan abarcar todos los tipos de bordes, texturas y toda información gráfica relevante, la realidad es que las imágenes médicas no necesitan muchos de estos detalles comunes en las imágenes estándares. Por lo tanto, podría ser oportuno investigar más a fondo la aplicación de redes preentrenadas únicamente en imágenes de estudios médicos. Estos sets de imágenes médicas masivos no son tan comunes y abundantes como los de imágenes comunes, pero en los últimos tiempos se han realizado grandes esfuerzos para su desarrollo, como por ejemplo con RadImageNet.

Otro punto que queda pendiente para estudiar es el compromiso entre definición de la imagen y la tridimensionalidad. Es sabido que mientras más pesados sean los datos de entrada, más exigencia computacional habrá. Por lo tanto, existe una relación de compromiso, a nivel

costo de cálculo y de tiempo, entre mantener la información de la tridimensionalidad y la información de la definición. En este proyecto se partió de la premisa de usar tensores volumétricos a costa de la definición, es decir, se decidió que la información espacial que otorgan las imágenes 3D es más valiosa que los detalles que se puedan perder cuando se disminuye la resolución. Entonces, se comenzó bajando la definición de los ejemplos a 64x64x64 vóxeles, cuando en un principio, los ejemplos exportados del sistema del centro tienen una definición en el rango de 300 a 700 vóxeles, por cada eje, aproximadamente. Sería interesante la comparación de los resultados del desempeño de este proyecto contra los de una red que trabaje con cortes bidimensionales con los mismos ejemplos, pero en cortes 2D de tamaño 512x512 (cada imagen tendría la misma cantidad de píxeles de entrada que los tensores tridimensionales utilizados en este proyecto, $2^6 \times 2^6 \times 2^6 = 2^9 \times 2^9$).

Siguiendo el tema de la cantidad de datos, en este proyecto se simplificó mucho el problema desde un principio. Se dio por hecho que con pocos datos se podían conseguir resultados aceptables, con el fin de evitar la sobrecarga computacional. Es por esto que se decidió trabajar únicamente con la información estructural de la próstata, vejiga y recto, cuando en realidad los datos exportados del sistema Aria también tienen disponible la información de las cabezas femorales, la uretra, el bulbo peneano y hasta el intestino, en algunos casos. Probablemente, incluyendo esta información extra se le estaría entregando a la red más herramientas para ajustar la dosis y así se podría conseguir un mejor desempeño final, pero a costa de más gasto computacional, claro está. Otra estrategia que se podría probar es modificar el preprocesamiento, en vez de recortar la distribución de dosis al tamaño justo que ocupan los órganos se podría dejar el tamaño original, es decir, todo el espacio ocupado por la pelvis. Así se le estaría entregando a la red más contexto de la naturaleza geométrica de los haces de radiación atravesando el cuerpo, y de esta forma podría llegar a mejores ajustes en la distribución de dosis.

Para ir cerrando con esta sección, se decidió hacer una breve alusión a un par de trabajos de temática similar realizados en los últimos años. Es necesario mencionar que nunca fue uno de los objetivos de este proyecto equiparar o mejorar otros trabajos similares. Esos proyectos en general obtienen mejores resultados, pero también parten de una base con muchos más recursos a nivel técnico y de conocimientos. Por ejemplo, uno de los trabajos más citados es "A feasibility study for predicting optimal radiation therapy dose distributions of prostate cancer patients" de Nguyen et al. [4]. En este trabajo del 2019 se utilizó una red U-Net para predecir distribuciones de dosis de IMRT y se consiguió un MSE de entrenamiento final de 0.0000102. Este valor de MSE es dos órdenes de magnitud menor que el 0.00216 final que se obtuvo con la U-Net Mejorada, lo que es una muestra de la diferencia de nivel con dicho trabajo. Pero, sería erróneo compara directamente este proyecto con dicho paper, en principio las imágenes que usaron corresponden a planificaciones IMRT y no VMRT como las de este proyecto, es decir, sus datos son esencialmente diferentes, hasta se podría decir que más simples, ya que en IMRT los rayos de radiación se puede distinguir claramente. Particularmente en el trabajo de Nguyen et al. los ejemplos utilizados se componen de 7 ángulos de haz y en su trabajo se menciona que el modelo aprendió a predecir sólo la dosis que proviene de aproximadamente las mismas orientaciones, y es posible que no pueda manejar geometrías de haz más complejas. Hablando del modelo, su arquitectura U-Net tiene 7 niveles de profundidad (6 skip-connections, el doble de las que se utilizan en este proyecto), lo que es viable porque su entrada son 6 canales de imágenes bidimensionales de 256x256. En otras palabras, cómo se parte de imágenes con más definición, se pueden aplicar más capas de codificación, en el cuello de botella sus mapas de características tienen un tamaño de 4x4, mismo tamaño que tienen los de la ResU-Net y la U-Net Mejorada en sus cuellos de botella, por haber comenzado con una definición de 64x64x64. Otra diferencia a tener en cuenta es que se está hablando de una red 2D entrenada con 80

ejemplos (72 para entrenamiento y 8 para validación) durante 1000 épocas, lo que en total tardaron 6 días para terminar el entrenamiento en 5 tarjetas gráficas NVIDIA Tesla K80 dual-GPU (10 chips GPU en total). Claramente, en ese proyecto se aplicó mucha más técnica en todos los niveles, incluso los resultados fueron más complejos de interpretar porque se utilizó validación cruzada con 10 agrupaciones, en las que cada una produce un modelo de predicción distinto, de los que se debe tener una estadística. A fin de cuentas, obtuvieron un índice de Dice promedio de 0.91 para los datos de testeo, con todos los índices de los intervalos porcentuales por arriba de 0.85 (aunque no se aclara con cuántos dígitos después de la coma se calcula).

Otro trabajo, un poco más parecido a este proyecto, es el que se presenta en el paper “A convolutional neural network approach for IMRT dose distribution prediction in prostate cancer patients” de Kajikawa et al. [38]. En ese trabajo se realizó una CNN 3D para predecir a partir de contornos la distribución de dosis IMRT, en el cáncer de próstata. Kajikawa et al. evaluaron el rendimiento del modelo comparándolo con planificaciones realizadas con RapidPlan™ y concluyeron que las predicciones logradas con la CNN son superiores o como mínimo comparables con las generadas con RapidPlan™. Este trabajo, también del 2019, utilizó una U-Net en las que se ingresaban imágenes de 64x64x64 con cuatro capas correspondientes a los órganos (agregando la uretra) y se generaron 5 grupos de cross-validation y cada uno se entrenó durante 250 épocas. De igual forma que en el trabajo anteriormente mencionado, la base de datos utilizada en el trabajo está formada por imágenes de pacientes tratados con IMRT (de ocho campos). Aunque en este trabajo no se mencionan métricas para imágenes como Dice, se puede ver en una de las imágenes de entrenamiento que se alcanzaron valores de MSE cercanos a 0.0005, que sirve como punto de referencia para contrastar con el presente proyecto. En conclusión, este trabajo también muestra resultados superiores, teniendo menos épocas de entrenamiento con datos tridimensionales como los de este proyecto.

De todas formas, sigue siendo muy aventurado querer comparar este tipo de trabajos a partir de una sola métrica o gráfico. Es erróneo esperar que la misma red con el mismo entrenamiento vaya a tener el mismo desempeño con datos que varían desde el tipo de RT hasta el lugar de donde se extrajeron. A pesar de esto, se pueden extraer características útiles que se tendrían que tener en cuenta para el futuro perfeccionamiento de estas tecnologías. Por ejemplo, una de estas características es la técnica de cross-validation, utilizada en los dos trabajos mencionados, para compensar la falta de ejemplos.

Para cerrar, solo queda recordar que este tipo de trabajos no tienen límite, se puede estar años realizando cambios, probando diferentes estrategias de preprocesamiento, nuevas arquitecturas, técnicas y funciones. Además, el avance de la tecnología tampoco parece tener un techo, por lo que continuamente aparecen nuevas herramientas que dan lugar a mejores resultados, especialmente considerando que gran parte de estos trabajos es realizar un proceso iterativo de prueba y error con todas estas variaciones que influyen en el resultado final.

12 Conclusiones

En este proyecto integrador, se construyeron, entrenaron y compararon tres modelos predictivos para el mapeo de distribución de dosis 3D de SBRT con RapidArc™ para tratamiento de cáncer de próstata. Las redes fueron adaptadas de la U-Net 2D de Ronneberger et al. [7], originalmente empleada para segmentación semántica. Los datos implementados como entrada para entrenamiento y testeo de las CNN fueron los contornos de la planificación de TAC de la próstata (PTV), recto y vejiga (OAR). En una visión global del proyecto, se buscó simplificar al máximo posible los datos para facilitar los procesos de entrenamiento, y a partir de eso ver qué calidad de resultados se obtenía.

El segundo modelo realizado, la U-Net Mejorada, presentó un desempeño superior a los otros dos, mostrando una mayor precisión y robustez, mejorando alrededor de un 0.05 el índice de Dice promedio, respecto al segundo mejor desempeño. Esta red utiliza varias técnicas que buscan disminuir el ruido y optimizar las operaciones que se realizan, pero a costa de aumentar la cantidad de cálculos que se realizan. En líneas generales, las predicciones conseguidas con las tres redes tienden a ser destacables para los vóxeles que reciben el 0%, 10% y 100% de la dosis total prescrita, con valores de índice de Dice cercanos a 0.8 (especialmente en la primera y segunda red). Por otro lado, para los intervalos porcentuales de dosis intermedios la tendencia de la calidad de los resultados es de regular a mala, ya que se consiguieron valores de índice de Dice alrededor de 0.5 y 0.6.

Los resultados y avances generados durante el desarrollo de este trabajo cumplen ampliamente los objetivos planteados un principio. Además, su realización dejó invaluable aprendizajes y experiencias sobre todos los temas que se abarcaron durante el proceso, especialmente sobre radioterapia y Deep Learning. Por otro lado, estos resultados no son aceptables a nivel clínico y todavía se necesitarían ajustes manuales para poder ser utilizados en tratamientos reales, pero son un punto de partida para continuar esta línea de investigación y desarrollo de estas tecnologías. Esta temática de investigación se encuentra en auge, particularmente, una versión preliminar de este trabajo se presentó como trabajo en formación en el congreso SABI2022 y la versión final será enviada al AAPM (American Association of Physicists in Medicine) para participar del proceso de selección para el 65th Annual Meeting & Exhibition (2023).

En un futuro, estos modelos podrían proporcionar datos de dosis de alta precisión en poco tiempo para cualquier paciente, por lo tanto, ahorra un tiempo valioso, principalmente en el proceso de optimización de dosis en la clínica. Finalmente, el tiempo total de planificación del tratamiento podría acortarse considerablemente, permitiendo a los médicos y físicos o dosimetristas centrar sus esfuerzos en casos más desafiantes.

13 Trabajos futuros

Como se mencionó en secciones anteriores, desde un principio, para la ejecución de este proyecto, se partió de acotar rigurosamente la base de datos, formada por pacientes con una enfermedad y un tratamiento muy específicos. Entonces, podría ser oportuno extender las técnicas aplicadas a más tipos de cánceres, por ejemplo, el cáncer de mamá izquierda, que requiere muchas consideraciones particulares por cada paciente porque está ubicado cerca de estructuras sumamente delicadas como lo son el corazón y el pulmón.

Además, para prevenir el desbordamiento de la capacidad de los procesadores, desde un principio se planteó que la base de datos que se iba a utilizar estaría formada por volúmenes que incluían únicamente tres de las estructuras involucradas, cuando por lo general, en las planificaciones de tratamiento se consideran muchos más órganos (por ejemplo, las cabezas femorales). Es por esto que una posible ampliación de este trabajo podría ser incluir una mayor cantidad de estructuras disponibles en la demarcación realizada por los médicos. También, cabe reiterar que, se partió de redimensionar todos los volúmenes a un tamaño común de 64x64x64 píxeles ajustados a los órganos considerados, lo que pudo haber afectado negativamente el contexto necesario para la predicción de los modelos. Por lo que, otra extensión de este trabajo sería entrenar los modelos con otros redimensionamientos del dataset de entrada. Se tendría que idear una estrategia para que la red trabaje con un dataset con diferentes longitudes en cada eje de los volúmenes, es decir, con arreglos que no sean cubos si no prismas cuadrangulares (por lo general, las imágenes NIfTI tiene un eje transversal bastante mayor al anteroposterior y al longitudinal), y así utilizar datos más fieles a los generados en la tomografía inicial.

Finalmente, a nivel arquitectura, falta experimentar con varias redes que se mencionan recurrentemente en la bibliografía relevante al tema, como en [3] y [33], como lo son las redes GAN y ResNet-antiResNet, que también demostraron alcanzar muy buenos resultados y podrían llegar a tener un desempeño más satisfactorio que el que se obtuvo con las redes que se implementaron en este proyecto. Por la misma línea, se debe tener presente que no se explotó al máximo la capacidad de los métodos de transfer learning, que pueden llegar a ser una técnica clave para mejorar todavía más la calidad de las predicciones generadas, comenzando por usar modelos más grandes (por ejemplo, ResNet50) y posteriormente probando con otras redes preentrenadas disponibles.

14 Bibliografía y Referencias

- [1] A. Jemal, R. Siegel, and E. M. Ward, "Cancer facts and figures 2011," Atlanta: American Cancer Society, 2011.
- [2] H. Sung *et al.*, "Global Cancer Statistics 2020: GLOBOCAN Estimates of Incidence and Mortality Worldwide for 36 Cancers in 185 Countries," *CA Cancer J Clin*, vol. 71, no. 3, pp. 209–249, May 2021, doi: 10.3322/caac.21660.
- [3] S. Momin *et al.*, "Knowledge-based radiation treatment planning: A data-driven method survey," *Journal of Applied Clinical Medical Physics*, vol. 22, no. 8. John Wiley and Sons Ltd, pp. 16–44, Aug. 01, 2021. doi: 10.1002/acm2.13337.
- [4] D. Nguyen *et al.*, "A feasibility study for predicting optimal radiation therapy dose distributions of prostate cancer patients from patient anatomy using deep learning," 2019.
- [5] J. Deng and L. Xing, *Big Data in Radiation Oncology*. 2019. [Online]. Available: <https://www.crcpress.com/Series-in-Optics-and-Optoelectronics/book-series/TFOPTICSOPT>
- [6] E. Shelhamer, J. Long, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," May 2016, [Online]. Available: <http://arxiv.org/abs/1605.06211>
- [7] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Verlag, 2015, pp. 234–241. doi: 10.1007/978-3-319-24574-4_28.
- [8] Organización Mundial de la Salud, "Cáncer," Feb. 02, 2022. <https://www.who.int/es/news-room/fact-sheets/detail/cancer>
- [9] NCCN, "NCCN Guidelines Prostate Cancer." 2020.
- [10] P. Rawla, "Epidemiology of Prostate Cancer," *World J Oncol*, vol. 10, no. 2, pp. 63–89, 2019, doi: 10.14740/wjon1191.
- [11] R. Baskar, K. A. Lee, R. Yeo, and K. W. Yeoh, "Cancer and radiation therapy: Current advances and future directions," *International Journal of Medical Sciences*, vol. 9, no. 3. pp. 193–199, Feb. 27, 2012. doi: 10.7150/ijms.3635.
- [12] M. Koukourakis and Stavros Touloupidis, "External Beam Radiotherapy for Prostate Cancer: Current Position and Trends," *Anticancer Res*, vol. 26, pp. 485–494, 2006.
- [13] E. Rosenblatt and E. Zubizarreta, Eds., *Radiotherapy in cancer care: facing the global challenge*. Vienna, Austria: International Atomic Energy Agency, 2017.
- [14] "Introduction to Radiation Therapy." <https://training.seer.cancer.gov/treatment/radiation/>
- [15] D. Palma *et al.*, "Volumetric Modulated Arc Therapy for Delivery of Prostate Radiotherapy: Comparison With Intensity-Modulated Radiotherapy and Three-Dimensional Conformal Radiotherapy," *Int J Radiat Oncol Biol Phys*, vol. 72, no. 4, pp. 996–1001, Nov. 2008, doi: 10.1016/j.ijrobp.2008.02.047.

- [16] K. T. Afrin and S. Ahmad, "Clinical Differences Among Volumetric Modulated Arc Therapy, Intensity Modulated Radiation Therapy, and 3D- Conformal Radiation Therapy in Prostate Cancer: A Brief Review Study," *Multidisciplinary Cancer Investigation*, vol. 5, no. 4, pp. 1–7, Oct. 2021, doi: 10.30699/mci.5.4.522-1.
- [17] International Commission on Radiation Units and Measurements, "ICRU Report 62. Prescribing, recording, and reporting photon beamtherapy (Supplement to ICRU Report 50)," Bethesda, 1999.
- [18] B. E. Nelms *et al.*, "Variation in external beam treatment plan quality: An inter-institutional study of planners and planning systems," *Pract Radiat Oncol*, vol. 2, no. 4, pp. 296–305, Oct. 2012, doi: 10.1016/j.prro.2011.11.012.
- [19] S. H. Benedict *et al.*, "Stereotactic body radiation therapy: The report of AAPM Task Group 101," *Medical Physics*, vol. 37, no. 8. John Wiley and Sons Ltd, pp. 4078–4101, 2010. doi: 10.1118/1.3438081.
- [20] A. M. Barragán-Montero *et al.*, "Three-Dimensional Dose Prediction for Lung IMRT Patients with Deep Neural Networks: Robust Learning from Heterogeneous Beam Configurations," pp. 3679–3691, 2019.
- [21] C. Schubert *et al.*, "Intercenter validation of a knowledge based model for automated planning of volumetric modulated arc therapy for prostate cancer. The experience of the German RapidPlan Consortium," *PLoS One*, vol. 12, no. 5, May 2017, doi: 10.1371/journal.pone.0178034.
- [22] Y. Murakami, T. Magome, K. Matsumoto, T. Sato, Y. Yoshioka, and M. Oguchi, "Fully automated dose prediction using generative adversarial networks in prostate cancer patients," *PLoS One*, vol. 15, no. 5, May 2020, doi: 10.1371/journal.pone.0232697.
- [23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MA: MIT Press,. 2016.
- [24] N. Ketkar and J. Moolayil, *Deep Learning with Python*, 2nd ed. 2021.
- [25] J. Schmidhuber, "Deep Learning in neural networks: An overview," *Neural Networks*, vol. 61. Elsevier Ltd, pp. 85–117, Jan. 01, 2015. doi: 10.1016/j.neunet.2014.09.003.
- [26] H. Wang and B. Raj, "On the Origin of Deep Learning," Mar. 2017.
- [27] X. Bai, J. Zhang, B. Wang, S. Wang, Y. Xiang, and Q. Hou, "Sharp loss: a new loss function for radiotherapy dose prediction based on fully convolutional networks," *Biomed Eng Online*, vol. 20, no. 1, Dec. 2021, doi: 10.1186/s12938-021-00937-w.
- [28] A. A. Taha and A. Hanbury, "Metrics for evaluating 3D medical image segmentation: Analysis, selection, and tool," *BMC Med Imaging*, vol. 15, no. 1, Aug. 2015, doi: 10.1186/s12880-015-0068-x.
- [29] A. Rahman and Y. Wang, "Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation."
- [30] P. Shi, M. Duan, L. Yang, W. Feng, L. Ding, and L. Jiang, "An Improved U-Net Image Segmentation Method and Its Application for Metallic Grain Size Statistics," *Materials*, vol. 15, no. 13, Jul. 2022, doi: 10.3390/ma15134417.

- [31] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations," Jul. 2017, doi: 10.1007/978-3-319-67558-9_28.
- [32] F. Milletari, N. Navab, and S. A. Ahmadi, "V-Net: Fully convolutional neural networks for volumetric medical image segmentation," in *Proceedings - 2016 4th International Conference on 3D Vision, 3DV 2016*, Institute of Electrical and Electronics Engineers Inc., Dec. 2016, pp. 565–571. doi: 10.1109/3DV.2016.79.
- [33] C.-W. Wang, M.-A. Khalil, and N. P. Firdi, "A Survey on Deep Learning for Precision Oncology," *Diagnostics*, vol. 12, no. 6, p. 1489, Jun. 2022, doi: 10.3390/diagnostics12061489.
- [34] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," Mar. 2016, [Online]. Available: <http://arxiv.org/abs/1603.07285>
- [35] Z. Wojna *et al.*, "The Devil is in the Decoder: Classification, Regression and GANs," Jul. 2017, [Online]. Available: <http://arxiv.org/abs/1707.05847>
- [36] Y. Dai, H. Lu, and C. Shen, "Learning Affinity-Aware Upsampling for Deep Image Matting *."
- [37] S. B. Blumberg, D. Raví, M.-C. Xu, M. Figini, I. Kokkinos, and D. C. Alexander, "Deformably-Scaled Transposed Convolution," Oct. 2022, [Online]. Available: <http://arxiv.org/abs/2210.09446>
- [38] T. Kajikawa *et al.*, "A convolutional neural network approach for IMRT dose distribution prediction in prostate cancer patients," *J Radiat Res*, vol. 60, no. 5, pp. 685–693, Oct. 2019, doi: 10.1093/jrr/rrz051.
- [39] S. Chen, K. Ma, and Y. Zheng, "Med3D: Transfer Learning for 3D Medical Image Analysis," Apr. 2019, [Online]. Available: <http://arxiv.org/abs/1904.00625>
- [40] D. Cheng and E. Y. Lam, "Transfer Learning U-Net Deep Learning for Lung Ultrasound Segmentation."
- [41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015, [Online]. Available: <http://arxiv.org/abs/1512.03385>
- [42] "Computadora Nabucodonosor (ML)."
<https://ccad.unc.edu.ar/equipamiento/computadora-nabucodonosor/>
- [43] "Cluster Mendieta Fase 2." <https://ccad.unc.edu.ar/equipamiento/cluster-mendieta/cluster-mendieta-fase-2/>
- [44] I. Sumida *et al.*, "A convolution neural network for higher resolution dose prediction in prostate volumetric modulated arc therapy," *Physica Medica*, vol. 72, pp. 88–95, Apr. 2020, doi: 10.1016/j.ejmp.2020.03.023.

15 Anexos

15.1 Anexo I: Resumen de arquitecturas

15.1.1 U-Net Básica

En la siguiente tabla se puede observar una descripción detallada, capa por capa, de la primera arquitectura implementada, que se explica en la sección [U-Net Básica](#).

Tabla 3: Summary de la U-Net Básica

Layer (type)	Output Shape	Param #
Conv3d-1	[-1, 64, 64, 64, 64]	6,976
BatchNorm3d-2	[-1, 64, 64, 64, 64]	128
ReLU-3	[-1, 64, 64, 64, 64]	0
Conv3d-4	[-1, 64, 64, 64, 64]	110,656
BatchNorm3d-5	[-1, 64, 64, 64, 64]	128
ReLU-6	[-1, 64, 64, 64, 64]	0
MaxPool3d-7	[-1, 64, 32, 32, 32]	0
Conv3d-8	[-1, 128, 32, 32, 32]	221,312
BatchNorm3d-9	[-1, 128, 32, 32, 32]	256
ReLU-10	[-1, 128, 32, 32, 32]	0
Conv3d-11	[-1, 128, 32, 32, 32]	442,496
BatchNorm3d-12	[-1, 128, 32, 32, 32]	256
ReLU-13	[-1, 128, 32, 32, 32]	0
MaxPool3d-14	[-1, 128, 16, 16, 16]	0
Conv3d-15	[-1, 256, 16, 16, 16]	884,992
BatchNorm3d-16	[-1, 256, 16, 16, 16]	512
ReLU-17	[-1, 256, 16, 16, 16]	0
Conv3d-18	[-1, 256, 16, 16, 16]	1,769,728
BatchNorm3d-19	[-1, 256, 16, 16, 16]	512
ReLU-20	[-1, 256, 16, 16, 16]	0
MaxPool3d-21	[-1, 256, 8, 8, 8]	0
Conv3d-22	[-1, 512, 8, 8, 8]	3,539,456
BatchNorm3d-23	[-1, 512, 8, 8, 8]	1,024
ReLU-24	[-1, 512, 8, 8, 8]	0
Conv3d-25	[-1, 512, 8, 8, 8]	7,078,400
BatchNorm3d-26	[-1, 512, 8, 8, 8]	1,024
ReLU-27	[-1, 512, 8, 8, 8]	0
ConvTranspose3d-28	[-1, 256, 16, 16, 16]	1,048,832
Conv3d-29	[-1, 256, 16, 16, 16]	3,539,200
BatchNorm3d-30	[-1, 256, 16, 16, 16]	512
ReLU-31	[-1, 256, 16, 16, 16]	0
Conv3d-32	[-1, 256, 16, 16, 16]	1,769,728
BatchNorm3d-33	[-1, 256, 16, 16, 16]	512
ReLU-34	[-1, 256, 16, 16, 16]	0
deconv-35	[-1, 256, 16, 16, 16]	0
ConvTranspose3d-36	[-1, 128, 32, 32, 32]	262,272
Conv3d-37	[-1, 128, 32, 32, 32]	884,864
BatchNorm3d-38	[-1, 128, 32, 32, 32]	256
ReLU-39	[-1, 128, 32, 32, 32]	0
Conv3d-40	[-1, 128, 32, 32, 32]	442,496
BatchNorm3d-41	[-1, 128, 32, 32, 32]	256
ReLU-42	[-1, 128, 32, 32, 32]	0
deconv-43	[-1, 128, 32, 32, 32]	0
ConvTranspose3d-44	[-1, 64, 64, 64, 64]	65,600
Conv3d-45	[-1, 64, 64, 64, 64]	221,248
BatchNorm3d-46	[-1, 64, 64, 64, 64]	128
ReLU-47	[-1, 64, 64, 64, 64]	0

Conv3d-48	[-1, 64, 64, 64, 64]	110,656
BatchNorm3d-49	[-1, 64, 64, 64, 64]	128
ReLU-50	[-1, 64, 64, 64, 64]	0
deconv-51	[-1, 64, 64, 64, 64]	0
Conv3d-52	[-1, 1, 64, 64, 64]	65

Total params: 22,404,609
 Trainable params: 22,404,609
 Non-trainable params: 0

Input size (MB): 4.00
 Forward/backward pass size (MB): 2387.00
 Params size (MB): 85.47
 Estimated Total Size (MB): 2476.47

15.1.2 U-Net Mejorada

En la siguiente table se puede observar una descripción detallada, capa por capa, de la segunda arquitectura implementada, que se explica en la sección [U-Net Mejorada](#).

Tabla 4: Summary de la segunda U-Net implementada

Layer (type)	Output Shape	Param #
ReflectionPad3d-1	[-1, 4, 66, 66, 66]	0
Conv3d-2	[-1, 64, 64, 64, 64]	6,976
BatchNorm3d-3	[-1, 64, 64, 64, 64]	128
PReLU-4	[-1, 64, 64, 64, 64]	1
ReflectionPad3d-5	[-1, 64, 66, 66, 66]	0
Conv3d-6	[-1, 64, 64, 64, 64]	110,656
PReLU-7	[-1, 64, 64, 64, 64]	1
BatchNorm3d-8	[-1, 64, 64, 64, 64]	128
PReLU-9	[-1, 64, 64, 64, 64]	1
encoding_block-10	[-1, 64, 64, 64, 64]	0
MaxPool3d-11	[-1, 64, 32, 32, 32]	0
ReflectionPad3d-12	[-1, 64, 34, 34, 34]	0
Conv3d-13	[-1, 128, 32, 32, 32]	221,312
BatchNorm3d-14	[-1, 128, 32, 32, 32]	256
PReLU-15	[-1, 128, 32, 32, 32]	1
ReflectionPad3d-16	[-1, 128, 34, 34, 34]	0
Conv3d-17	[-1, 128, 32, 32, 32]	442,496
PReLU-18	[-1, 128, 32, 32, 32]	1
BatchNorm3d-19	[-1, 128, 32, 32, 32]	256
PReLU-20	[-1, 128, 32, 32, 32]	1
encoding_block-21	[-1, 128, 32, 32, 32]	0
MaxPool3d-22	[-1, 128, 16, 16, 16]	0
ReflectionPad3d-23	[-1, 128, 18, 18, 18]	0
Conv3d-24	[-1, 256, 16, 16, 16]	884,992
BatchNorm3d-25	[-1, 256, 16, 16, 16]	512
PReLU-26	[-1, 256, 16, 16, 16]	1
ReflectionPad3d-27	[-1, 256, 18, 18, 18]	0
Conv3d-28	[-1, 256, 16, 16, 16]	1,769,728
PReLU-29	[-1, 256, 16, 16, 16]	1
BatchNorm3d-30	[-1, 256, 16, 16, 16]	512
PReLU-31	[-1, 256, 16, 16, 16]	1
encoding_block-32	[-1, 256, 16, 16, 16]	0
MaxPool3d-33	[-1, 256, 8, 8, 8]	0
ReflectionPad3d-34	[-1, 256, 10, 10, 10]	0
Conv3d-35	[-1, 512, 8, 8, 8]	3,539,456
BatchNorm3d-36	[-1, 512, 8, 8, 8]	1,024
PReLU-37	[-1, 512, 8, 8, 8]	1
ReflectionPad3d-38	[-1, 512, 10, 10, 10]	0
Conv3d-39	[-1, 512, 8, 8, 8]	7,078,400
PReLU-40	[-1, 512, 8, 8, 8]	1

BatchNorm3d-41	[-1, 512, 8, 8, 8]	1,024
PReLU-42	[-1, 512, 8, 8, 8]	1
encoding_block-43	[-1, 512, 8, 8, 8]	0
MaxPool3d-44	[-1, 512, 4, 4, 4]	0
ReflectionPad3d-45	[-1, 512, 6, 6, 6]	0
Conv3d-46	[-1, 1024, 4, 4, 4]	14,156,800
BatchNorm3d-47	[-1, 1024, 4, 4, 4]	2,048
PReLU-48	[-1, 1024, 4, 4, 4]	1
ReflectionPad3d-49	[-1, 1024, 6, 6, 6]	0
Conv3d-50	[-1, 1024, 4, 4, 4]	28,312,576
PReLU-51	[-1, 1024, 4, 4, 4]	1
BatchNorm3d-52	[-1, 1024, 4, 4, 4]	2,048
PReLU-53	[-1, 1024, 4, 4, 4]	1
Dropout-54	[-1, 1024, 4, 4, 4]	0
encoding_block-55	[-1, 1024, 4, 4, 4]	0
Upsample-56	[-1, 1024, 8, 8, 8]	0
Conv3d-57	[-1, 512, 8, 8, 8]	524,800
decoding_block-58	[-1, 512, 8, 8, 8]	0
ReflectionPad3d-59	[-1, 1024, 10, 10, 10]	0
Conv3d-60	[-1, 512, 8, 8, 8]	14,156,288
BatchNorm3d-61	[-1, 512, 8, 8, 8]	1,024
PReLU-62	[-1, 512, 8, 8, 8]	1
ReflectionPad3d-63	[-1, 512, 10, 10, 10]	0
Conv3d-64	[-1, 512, 8, 8, 8]	7,078,400
PReLU-65	[-1, 512, 8, 8, 8]	1
BatchNorm3d-66	[-1, 512, 8, 8, 8]	1,024
PReLU-67	[-1, 512, 8, 8, 8]	1
Dropout-68	[-1, 512, 8, 8, 8]	0
encoding_block-69	[-1, 512, 8, 8, 8]	0
Upsample-70	[-1, 512, 16, 16, 16]	0
Conv3d-71	[-1, 256, 16, 16, 16]	131,328
decoding_block-72	[-1, 256, 16, 16, 16]	0
ReflectionPad3d-73	[-1, 512, 18, 18, 18]	0
Conv3d-74	[-1, 256, 16, 16, 16]	3,539,200
BatchNorm3d-75	[-1, 256, 16, 16, 16]	512
PReLU-76	[-1, 256, 16, 16, 16]	1
ReflectionPad3d-77	[-1, 256, 18, 18, 18]	0
Conv3d-78	[-1, 256, 16, 16, 16]	1,769,728
PReLU-79	[-1, 256, 16, 16, 16]	1
BatchNorm3d-80	[-1, 256, 16, 16, 16]	512
PReLU-81	[-1, 256, 16, 16, 16]	1
Dropout-82	[-1, 256, 16, 16, 16]	0
encoding_block-83	[-1, 256, 16, 16, 16]	0
Upsample-84	[-1, 256, 32, 32, 32]	0
Conv3d-85	[-1, 128, 32, 32, 32]	32,896
decoding_block-86	[-1, 128, 32, 32, 32]	0
ReflectionPad3d-87	[-1, 256, 34, 34, 34]	0
Conv3d-88	[-1, 128, 32, 32, 32]	884,864
BatchNorm3d-89	[-1, 128, 32, 32, 32]	256
PReLU-90	[-1, 128, 32, 32, 32]	1
ReflectionPad3d-91	[-1, 128, 34, 34, 34]	0
Conv3d-92	[-1, 128, 32, 32, 32]	442,496
PReLU-93	[-1, 128, 32, 32, 32]	1
BatchNorm3d-94	[-1, 128, 32, 32, 32]	256
PReLU-95	[-1, 128, 32, 32, 32]	1
Dropout-96	[-1, 128, 32, 32, 32]	0
encoding_block-97	[-1, 128, 32, 32, 32]	0
Upsample-98	[-1, 128, 64, 64, 64]	0
Conv3d-99	[-1, 64, 64, 64, 64]	8,256
decoding_block-100	[-1, 64, 64, 64, 64]	0
ReflectionPad3d-101	[-1, 128, 66, 66, 66]	0
Conv3d-102	[-1, 64, 64, 64, 64]	221,248
BatchNorm3d-103	[-1, 64, 64, 64, 64]	128

PReLU-104	[-1, 64, 64, 64, 64]	1
ReflectionPad3d-105	[-1, 64, 66, 66, 66]	0
Conv3d-106	[-1, 64, 64, 64, 64]	110,656
PReLU-107	[-1, 64, 64, 64, 64]	1
BatchNorm3d-108	[-1, 64, 64, 64, 64]	128
PReLU-109	[-1, 64, 64, 64, 64]	1
Dropout-110	[-1, 64, 64, 64, 64]	0
encoding_block-111	[-1, 64, 64, 64, 64]	0
Conv3d-112	[-1, 1, 64, 64, 64]	65

```
=====
Total params: 85,435,420
Trainable params: 85,435,420
Non-trainable params: 0
```

```
-----
Input size (MB): 4.00
Forward/backward pass size (MB): 4412.13
Params size (MB): 325.91
Estimated Total Size (MB): 4742.04
-----
```

15.1.3 ResU-Net

En la siguiente tabla se puede observar una descripción detallada, capa por capa, de la última arquitectura implementada, donde hasta la capa 65 se hace uso de capas preentrenadas, lo que se explica en la sección [ResU-Net](#).

Tabla 5: Summary de la tercera U-Net implementada

Layer (type)	Output Shape	Param #
Conv3d-1	[-1, 64, 32, 32, 32]	87,808
BatchNorm3d-2	[-1, 64, 32, 32, 32]	128
ReLU-3	[-1, 64, 32, 32, 32]	0
Conv3DSimple-4	[-1, 64, 32, 32, 32]	110,592
BatchNorm3d-5	[-1, 64, 32, 32, 32]	128
ReLU-6	[-1, 64, 32, 32, 32]	0
Conv3DSimple-7	[-1, 64, 32, 32, 32]	110,592
BatchNorm3d-8	[-1, 64, 32, 32, 32]	128
ReLU-9	[-1, 64, 32, 32, 32]	0
BasicBlock-10	[-1, 64, 32, 32, 32]	0
Conv3DSimple-11	[-1, 64, 32, 32, 32]	110,592
BatchNorm3d-12	[-1, 64, 32, 32, 32]	128
ReLU-13	[-1, 64, 32, 32, 32]	0
Conv3DSimple-14	[-1, 64, 32, 32, 32]	110,592
BatchNorm3d-15	[-1, 64, 32, 32, 32]	128
ReLU-16	[-1, 64, 32, 32, 32]	0
BasicBlock-17	[-1, 64, 32, 32, 32]	0
Conv3DSimple-18	[-1, 128, 16, 16, 16]	221,184
BatchNorm3d-19	[-1, 128, 16, 16, 16]	256
ReLU-20	[-1, 128, 16, 16, 16]	0
Conv3DSimple-21	[-1, 128, 16, 16, 16]	442,368
BatchNorm3d-22	[-1, 128, 16, 16, 16]	256
Conv3d-23	[-1, 128, 16, 16, 16]	8,192
BatchNorm3d-24	[-1, 128, 16, 16, 16]	256
ReLU-25	[-1, 128, 16, 16, 16]	0
BasicBlock-26	[-1, 128, 16, 16, 16]	0
Conv3DSimple-27	[-1, 128, 16, 16, 16]	442,368
BatchNorm3d-28	[-1, 128, 16, 16, 16]	256
ReLU-29	[-1, 128, 16, 16, 16]	0
Conv3DSimple-30	[-1, 128, 16, 16, 16]	442,368
BatchNorm3d-31	[-1, 128, 16, 16, 16]	256
ReLU-32	[-1, 128, 16, 16, 16]	0
BasicBlock-33	[-1, 128, 16, 16, 16]	0
Conv3DSimple-34	[-1, 256, 8, 8, 8]	884,736

BatchNorm3d-35	[-1, 256, 8, 8, 8]	512
ReLU-36	[-1, 256, 8, 8, 8]	0
Conv3DSimple-37	[-1, 256, 8, 8, 8]	1,769,472
BatchNorm3d-38	[-1, 256, 8, 8, 8]	512
Conv3d-39	[-1, 256, 8, 8, 8]	32,768
BatchNorm3d-40	[-1, 256, 8, 8, 8]	512
ReLU-41	[-1, 256, 8, 8, 8]	0
BasicBlock-42	[-1, 256, 8, 8, 8]	0
Conv3DSimple-43	[-1, 256, 8, 8, 8]	1,769,472
BatchNorm3d-44	[-1, 256, 8, 8, 8]	512
ReLU-45	[-1, 256, 8, 8, 8]	0
Conv3DSimple-46	[-1, 256, 8, 8, 8]	1,769,472
BatchNorm3d-47	[-1, 256, 8, 8, 8]	512
ReLU-48	[-1, 256, 8, 8, 8]	0
BasicBlock-49	[-1, 256, 8, 8, 8]	0
Conv3DSimple-50	[-1, 512, 4, 4, 4]	3,538,944
BatchNorm3d-51	[-1, 512, 4, 4, 4]	1,024
ReLU-52	[-1, 512, 4, 4, 4]	0
Conv3DSimple-53	[-1, 512, 4, 4, 4]	7,077,888
BatchNorm3d-54	[-1, 512, 4, 4, 4]	1,024
Conv3d-55	[-1, 512, 4, 4, 4]	131,072
BatchNorm3d-56	[-1, 512, 4, 4, 4]	1,024
ReLU-57	[-1, 512, 4, 4, 4]	0
BasicBlock-58	[-1, 512, 4, 4, 4]	0
Conv3DSimple-59	[-1, 512, 4, 4, 4]	7,077,888
BatchNorm3d-60	[-1, 512, 4, 4, 4]	1,024
ReLU-61	[-1, 512, 4, 4, 4]	0
Conv3DSimple-62	[-1, 512, 4, 4, 4]	7,077,888
BatchNorm3d-63	[-1, 512, 4, 4, 4]	1,024
ReLU-64	[-1, 512, 4, 4, 4]	0
BasicBlock-65	[-1, 512, 4, 4, 4]	0
ConvTranspose3d-66	[-1, 256, 8, 8, 8]	1,048,832
ReflectionPad3d-67	[-1, 512, 10, 10, 10]	0
Conv3d-68	[-1, 256, 8, 8, 8]	3,539,200
BatchNorm3d-69	[-1, 256, 8, 8, 8]	512
PReLU-70	[-1, 256, 8, 8, 8]	1
Dropout-71	[-1, 256, 8, 8, 8]	0
ReflectionPad3d-72	[-1, 256, 10, 10, 10]	0
Conv3d-73	[-1, 256, 8, 8, 8]	1,769,728
BatchNorm3d-74	[-1, 256, 8, 8, 8]	512
PReLU-75	[-1, 256, 8, 8, 8]	1
Dropout-76	[-1, 256, 8, 8, 8]	0
deconv-77	[-1, 256, 8, 8, 8]	0
ConvTranspose3d-78	[-1, 128, 16, 16, 16]	262,272
ReflectionPad3d-79	[-1, 256, 18, 18, 18]	0
Conv3d-80	[-1, 128, 16, 16, 16]	884,864
BatchNorm3d-81	[-1, 128, 16, 16, 16]	256
PReLU-82	[-1, 128, 16, 16, 16]	1
Dropout-83	[-1, 128, 16, 16, 16]	0
ReflectionPad3d-84	[-1, 128, 18, 18, 18]	0
Conv3d-85	[-1, 128, 16, 16, 16]	442,496
BatchNorm3d-86	[-1, 128, 16, 16, 16]	256
PReLU-87	[-1, 128, 16, 16, 16]	1
Dropout-88	[-1, 128, 16, 16, 16]	0
deconv-89	[-1, 128, 16, 16, 16]	0
ConvTranspose3d-90	[-1, 64, 32, 32, 32]	65,600
ReflectionPad3d-91	[-1, 128, 34, 34, 34]	0
Conv3d-92	[-1, 64, 32, 32, 32]	221,248
BatchNorm3d-93	[-1, 64, 32, 32, 32]	128
PReLU-94	[-1, 64, 32, 32, 32]	1
Dropout-95	[-1, 64, 32, 32, 32]	0
ReflectionPad3d-96	[-1, 64, 34, 34, 34]	0
Conv3d-97	[-1, 64, 32, 32, 32]	110,656

BatchNorm3d-98	[-1, 64, 32, 32, 32]	128
PReLU-99	[-1, 64, 32, 32, 32]	1
Dropout-100	[-1, 64, 32, 32, 32]	0
deconv-101	[-1, 64, 32, 32, 32]	0
ConvTranspose3d-102	[-1, 64, 64, 64, 64]	32,832
ReflectionPad3d-103	[-1, 64, 66, 66, 66]	0
Conv3d-104	[-1, 64, 64, 64, 64]	110,656
BatchNorm3d-105	[-1, 64, 64, 64, 64]	128
PReLU-106	[-1, 64, 64, 64, 64]	1
Dropout-107	[-1, 64, 64, 64, 64]	0
Conv3d-108	[-1, 1, 64, 64, 64]	65
out_conv-109	[-1, 1, 64, 64, 64]	0

=====
Total params: 41,716,232

Trainable params: 41,716,232

Non-trainable params: 0

Input size (MB): 4.00

Forward/backward pass size (MB): 1430.90

Params size (MB): 159.13

Estimated Total Size (MB): 1594.03

15.2 Anexo II: Transfer learning: ResNet18 3D

Descripción detallada de las capas de la red preentrenada para transfer learning, utilizada en la arquitectura detallada en [ResU-Net](#).

```
VideoResNet(
  (stem): BasicStem(
    (0): Conv3d(3, 64, kernel_size=(3, 7, 7), stride=(1, 2, 2), padding=(1, 3, 3), bias=False)
    (1): BatchNorm3d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
  )
  (layer1): Sequential(
    (0): BasicBlock(
      (conv1): Sequential(
        (0): Conv3DSimple(64, 64, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
        (1): BatchNorm3d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (2): ReLU(inplace=True)
      )
      (conv2): Sequential(
        (0): Conv3DSimple(64, 64, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
        (1): BatchNorm3d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
      (relu): ReLU(inplace=True)
    )
    (1): BasicBlock(
      (conv1): Sequential(
        (0): Conv3DSimple(64, 64, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
        (1): BatchNorm3d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (2): ReLU(inplace=True)
      )
      (conv2): Sequential(
        (0): Conv3DSimple(64, 64, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
        (1): BatchNorm3d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
  )
)
```



```

    )
    (relu): ReLU(inplace=True)
  )
)
(layer2): Sequential(
  (0): BasicBlock(
    (conv1): Sequential(
      (0): Conv3DSimple(64, 128, kernel_size=(3, 3, 3), stride=(2, 2, 2), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): ReLU(inplace=True)
    )
    (conv2): Sequential(
      (0): Conv3DSimple(128, 128, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
    (relu): ReLU(inplace=True)
    (downsample): Sequential(
      (0): Conv3d(64, 128, kernel_size=(1, 1, 1), stride=(2, 2, 2), bias=False)
      (1): BatchNorm3d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
  )
  (1): BasicBlock(
    (conv1): Sequential(
      (0): Conv3DSimple(128, 128, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): ReLU(inplace=True)
    )
    (conv2): Sequential(
      (0): Conv3DSimple(128, 128, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
    (relu): ReLU(inplace=True)
  )
)
(layer3): Sequential(
  (0): BasicBlock(
    (conv1): Sequential(
      (0): Conv3DSimple(128, 256, kernel_size=(3, 3, 3), stride=(2, 2, 2), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): ReLU(inplace=True)
    )
    (conv2): Sequential(
      (0): Conv3DSimple(256, 256, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
    (relu): ReLU(inplace=True)
    (downsample): Sequential(
      (0): Conv3d(128, 256, kernel_size=(1, 1, 1), stride=(2, 2, 2), bias=False)

```

```
(1): BatchNorm3d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
)
)
(1): BasicBlock(
  (conv1): Sequential(
    (0): Conv3DSimple(256, 256, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
    (1): BatchNorm3d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
  )
  (conv2): Sequential(
    (0): Conv3DSimple(256, 256, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
    (1): BatchNorm3d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (relu): ReLU(inplace=True)
)
)
(layer4): Sequential(
  (0): BasicBlock(
    (conv1): Sequential(
      (0): Conv3DSimple(256, 512, kernel_size=(3, 3, 3), stride=(2, 2, 2), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): ReLU(inplace=True)
    )
    (conv2): Sequential(
      (0): Conv3DSimple(512, 512, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
    (relu): ReLU(inplace=True)
    (downsample): Sequential(
      (0): Conv3d(256, 512, kernel_size=(1, 1, 1), stride=(2, 2, 2), bias=False)
      (1): BatchNorm3d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
  )
  (1): BasicBlock(
    (conv1): Sequential(
      (0): Conv3DSimple(512, 512, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): ReLU(inplace=True)
    )
    (conv2): Sequential(
      (0): Conv3DSimple(512, 512, kernel_size=(3, 3, 3), stride=(1, 1, 1), padding=(1, 1, 1), bias=False)
      (1): BatchNorm3d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
    (relu): ReLU(inplace=True)
  )
)
)
(avgpool): AdaptiveAvgPool3d(output_size=(1, 1, 1))
(fc): Linear(in_features=512, out_features=400, bias=True)
)
```

15.3 Anexo III: Nabucodonosor vs Mendieta

Solo a modo de ilustrativo, a continuación, se muestra un paralelismo entre la misma notebook (entrenamiento) corrida en ambas computadoras:

La primera mejora que se puede ver con Mendieta es que, ahora si se permite entrenar con un tamaño de batch mucho mayor, 16 en vez del límite de 2 que se tenía con Nabu. Además de esto, el tiempo de entrenamiento total requerido es aproximadamente la mitad, siendo en total para 100 épocas 44 minutos con 28 segundos contra 21 minutos con 14 segundos, como se puede ver en la ilustración 66 y 67.

```

model = UNet()
hist, met = fit(model, dataloader, epochs=100)

loss 0.00083 iou 0.667 dice 0.800: 100% | ██████████ | 90/90 [00:22<00:00, 4.07it/s]
test_loss 0.00126 iou 0.597 dice 0.747: 100% | ██████████ | 30/30 [00:03<00:00, 8.51it/s]

Epoch 99/100
loss 0.00087 iou 0.658 dice 0.794: 100% | ██████████ | 90/90 [00:23<00:00, 3.91it/s]
test_loss 0.00123 iou 0.602 dice 0.752: 100% | ██████████ | 30/30 [00:03<00:00, 8.42it/s]

Epoch 100/100
loss 0.00076 iou 0.713 dice 0.832: 100% | ██████████ | 90/90 [00:22<00:00, 3.93it/s]
test_loss 0.00135 iou 0.580 dice 0.734: 100% | ██████████ | 30/30 [00:04<00:00, 7.43it/s]

Entrenamiento de 100 epocas finalizado en 44m 28s

```

Ilustración 66: Ejemplificación de un entrenamiento de 100 épocas de la red U-Net Básica con función de pérdida MSELoss y batchsize de 2 en la computadora Nabucodonosor.

```

model = UNet()
hist, met = fit(model, dataloader, epochs=100)

loss 0.00113 iou 0.391 dice 0.563: 100% | ██████████ | 12/12 [00:11<00:00, 1.06it/s]
test_loss 0.00144 iou 0.630 dice 0.773: 100% | ██████████ | 4/4 [00:01<00:00, 2.35it/s]

Epoch 99/100
loss 0.00114 iou 0.559 dice 0.717: 100% | ██████████ | 12/12 [00:11<00:00, 1.09it/s]
test_loss 0.00140 iou 0.609 dice 0.757: 100% | ██████████ | 4/4 [00:01<00:00, 2.40it/s]

Epoch 100/100
loss 0.00121 iou 0.292 dice 0.452: 100% | ██████████ | 12/12 [00:11<00:00, 1.08it/s]
test_loss 0.00146 iou 0.602 dice 0.752: 100% | ██████████ | 4/4 [00:01<00:00, 2.32it/s]

Entrenamiento de 100 epocas finalizado en 21m 14s

```

Ilustración 67: Ejemplificación de un entrenamiento de 100 épocas de la red U-Net Básica con función de pérdida MSELoss y batchsize de 16 en la computadora Mendieta.

Con los gráficos de progreso del valor de la función de pérdida (Ilustración 68), se puede ver que con Mendieta hay una estabilidad mucho mayor y menor sobreajuste en el entrenamiento que se debe principalmente al cambio de del tamaño del batchsize.

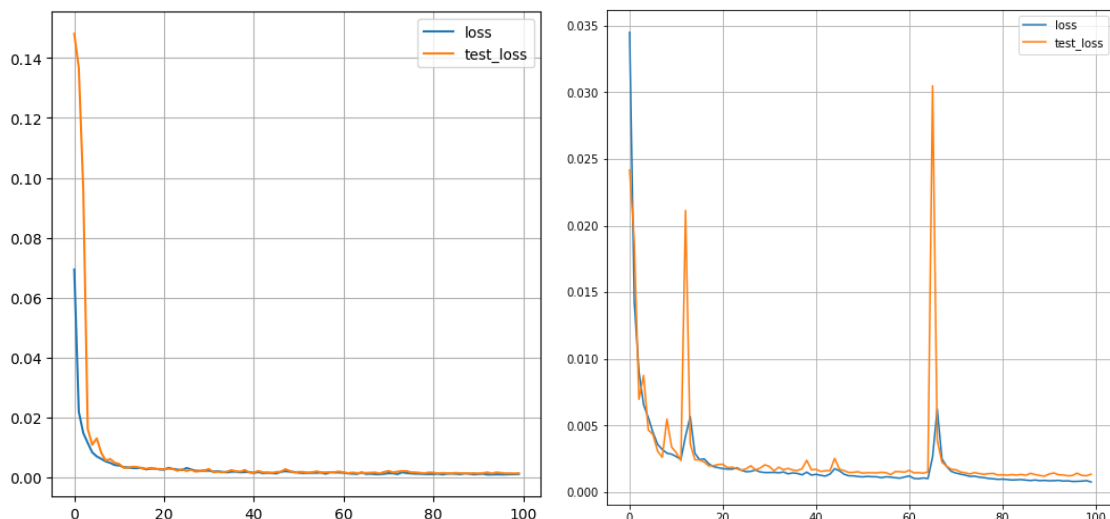


Ilustración 68: Gráficos de la evolución del valor de pérdida en entrenamiento y en validación de la res U-Net Básica con MSELoss en 100 épocas, con Mendieta (izquierda) y con Nabucodonosor (derecha).

15.4 Anexo IV: Códigos

A continuación, se facilitan los links de todos los códigos que se utilizaron en el desarrollo de este trabajo. Todos los códigos se encuentran en el siguiente repositorio: <https://github.com/InesSadir/PI-U-Net->

- Preparación del dataset: <https://github.com/InesSadir/PI-U-Net-/blob/main/1%C2%B0%20Datos%20PI%20-%20Preprocesamiento%20y%20Data%20augmentation.ipynb>
- Primer etapa de entrenamiento de la U-Net Básica: <https://github.com/InesSadir/PI-U-Net-/blob/main/2%C2%B0%20U-Net%20B%C3%A1sica.ipynb>
- Primer etapa de entrenamiento de la U-Net Mejorada: <https://github.com/InesSadir/PI-U-Net-/blob/main/3%C2%B0%20U-Net%20Mejorada.ipynb>
- Primer etapa de entrenamiento de la ResU-Net: <https://github.com/InesSadir/PI-U-Net-/blob/main/4%C2%B0%20Res-U-Net.ipynb>
- Etapa final de entrenamiento y testeos: <https://github.com/InesSadir/PI-U-Net-/blob/main/5%C2%B0%20Redes%20U-Net%20finales.ipynb>

Por causa del peso total del archivo, en las notebooks anteriores no se pueden mostrar las imágenes generadas con la librería "plotly".