Universidad Nacional Autónoma de México

Facultad de Ciencias

Segmentación de Estructuras Cardiovasculares mediante Redes Neuronales Convolucionales.

Reporte de Práctica Profesional

que para obtener el título de:

físico biomédico

P R E S E N T A :

Avendaño García Steve Alejandro

TUTOR

Dra. Jimena Olveres Montiel

Ciudad Universitaria, Cd. Mx., Enero 2020

Agradecimientos

Agradezco al apoyo de los programas UNAM-PAPIIT IA103119 e IN11691.

Índice general

Capítulo 1

Resumen

El área de imagenología médica consiste en el aprovechamiento de principios físicos y fisiológicos que ocurren en el organismo, tiene la finalidad obtener imágenes anatómicas estructurales y funcionales de un paciente, durante los últimos años esta área ha pasado por un gran desarrollo, permitiendo adquirir imágenes cada vez con una mayor nitidez, contraste y resolución, las técnicas mayormente empleadas para adquirir información tanto anatómica como funcional en forma de imágenes son Imagenología por Resonancia Magnética (MRI), Tomografía computarizada (CT), Tomografía por Emisión de Positrones (PET) y Ultrasonido (US).
Un caso particular del uso de estas técnicas es el diagnostico de enfermedades cardiovasculares, las cuales representan la primera causa de muerte a nivel mundial y en occidente. En las estadísticas de la Organización Mundial de la Salud se registra que simplemente en 2016 hubo más de 15 millones de muertes por este tipo de enfermedades, un ejemplo de esto son la cardiopatía isquémica y el accidente cerebro-vascular.

Debido a lo anterior, ha habido un aumento en la importancia del estudio del corazón y sus enfermedades, lo que ha provocado a su vez un aumento en la cantidad de información e imágenes que son generadas en el quehacer diario de la práctica clínica, siendo este superior al de cantidad de especialistas capaces de llevar a cabo evaluaciones en estas áreas, lo que limita el tiempo que el médico experto puede dedicar al diagnóstico de un caso particular, y afecta la calidad que el diagnostico pueda tener.

Con la finalidad de analizar lesiones, mal funcionamiento o algún tipo de malformación cardiaca, es necesaria la adquisición de una imagen médica y posterior segmentación de estructuras como: ventrículos, aurículas, aorta y venas pulmonares, además de que es un prerequisito para crear modelos anatómicos (fig. 1.2) que permitan planificar alguna intervención quirúrgica.

La segmentación manual de estas estructuras es el método más confiable, sin embargo, realizar esto en muchas imágenes resulta una tarea laboriosa que demanda mucho tiempo de un médico especialista, además de que este método está sujeto a una variabilidad inter- e intra- observador, ejemplo de esto es la imagen 1.3.

La segmentación de imágenes médicas es una tarea desafiante debido a características como la complejidad y variabilidad de la morfología, además de que se pueden presentar variaciones en las propiedades de la imagen debidas a la presencia de artefactos, o por el tipo de protocolo de adquisición, es por esto que la implementación de modelos semiautomáticos y automáticos aun presenta un problema abierto, sin embargo, lograr implementar estas herramientas de manera adecuada traería consigo ventajas como:

Figura 1.1: Estimaciones de salud global 2016. Organización Mundial de la Salud 2018 [1].

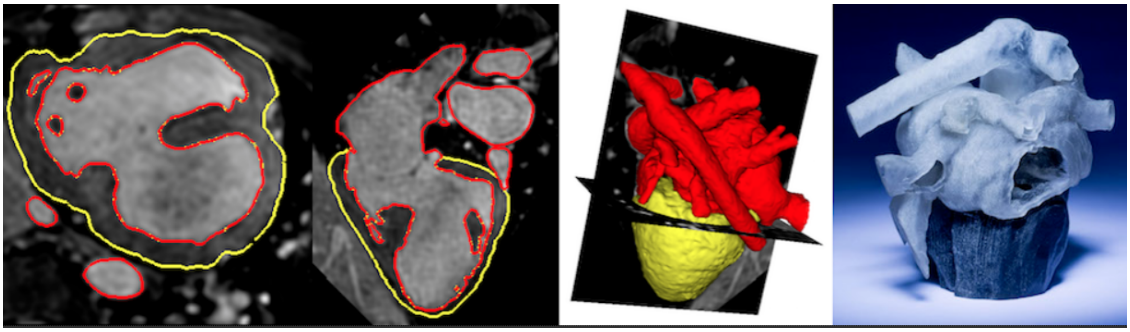


Figura 1.2: Segmentación y reconstrucción del volumen cardiaco [28].

- Facilitar la determinación y extracción de características estadísticamente relevantes que puedan ser de mayor importancia al momento de llevar a cabo el diagnóstico.
- Automatizar tareas repetitivas y apoyar el desarrollo de diagnóstico asistido por computador.
- Comenzar a marcar el camino hacia un análisis más cuantitativo y estandarizado, buscando disminuir la variabilidad y el error que puedan existir en un diagnóstico debido a la preparación, sobrecarga de trabajo y el tiempo del que dispone un médico radiólogo para llevar a cabo el diagnóstico.

1.1. Objetivo general

Desarrollar herramientas computacionales para asistir en el diagnóstico clínico en el área de imagenología médica.

1.2. Objetivos específicos

- Optimizar el flujo de trabajo en imagenología médica mediante la automatización de tareas como la segmentación de imágenes.

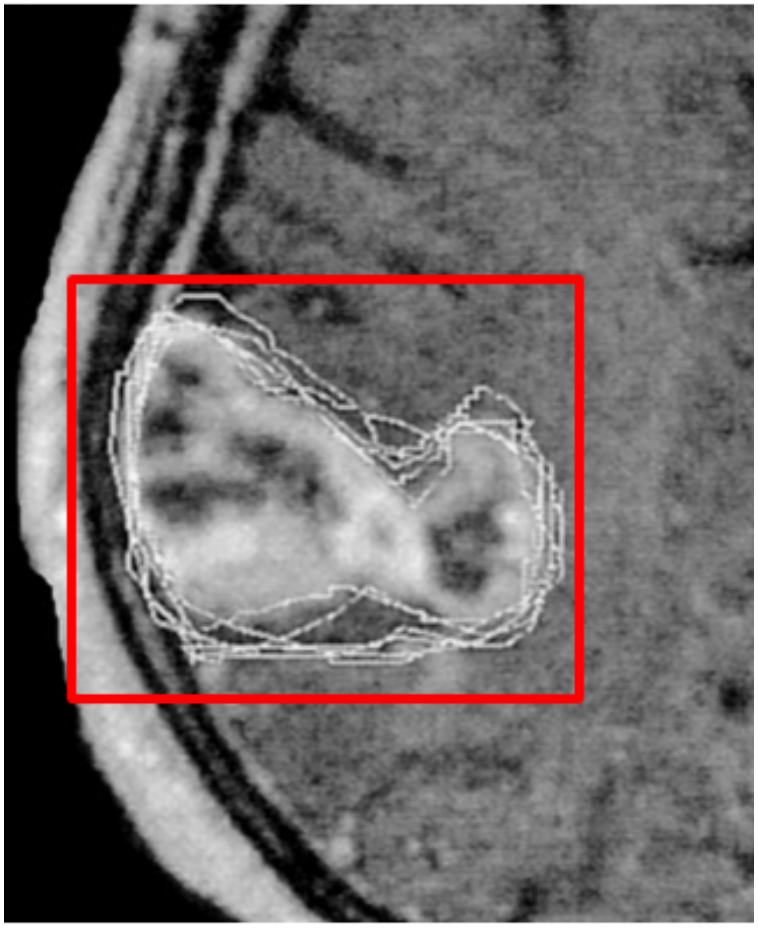


Figura 1.3: Imagen de cráneo en corte transversal obtenida por Resonancia Magnética T1, en ella se muestra la segmentación manual de un tumor cerebral, realizada por 9 especialistas independientes.

- Implementar algoritmos de Redes Neuronales Convolucionales que permita segmentar estructuras anatómicas a partir de imágenes adquiridas por Resonancia Magnética.
- Estudiar e implementar modificaciones en estos algoritmos para mejorar la eficiencia de la segmentación.

El desarrollo de técnicas como estas se ha venido realizando desde alrededor de 1950, particularmente el campo de visión por computadora, que busca automatizar las mismas tareas que puede llevar a cabo el sistema visual humano, hace uso de estas técnicas, en conjunto con herramientas de Inteligencia artificial, reconocimiento de patrones y más recientemente con algoritmos de aprendizaje profundo.
Tomando esto como motivación, a lo largo de este proyecto se implementó un algoritmo basado en aprendizaje profundo, más específicamente en **Redes Neuronales Convolucionales** (CNN), el cual tiene el propósito de identificar y segmentar estructuras anatómicas de interés, que en este caso correspondieron a las cámaras sanguíneas y al musculo ventricular del corazón.
Desde el punto de vista científico, la contribución de este proyecto radica en el desarrollo de métodos novedosos de segmentación. Desde el punto de vista tecnológico, la intención es crear herramientas de auxilio y apoyo al diagnóstico y seguimiento médico que, en este caso,

le brinden al médico información útil que le permita llevar a cabo un mejor diagnóstico, así como agilizar este proceso y obtener un ahorro significativo del tiempo invertido.

Capítulo 2

Introducción

Con la aparición de nuevo hardware en el área médica, la generación de datos se ha visto en aumento durante los últimos años, sin embargo, para crear un flujo de trabajo más eficiente, también es necesario el desarrollo de nuevo software, el cual permita procesar toda esta información y presentar al médico especialista la información más significativa para llevar a cabo el diagnostico.

Es posible representar de forma cualitativa la precisión del diagnóstico llevado a cabo por un médico experto como función del tiempo que este ha invertido en analizar la información con la que cuenta, del mismo modo, sería necesario representar la variabilidad que existe entre las decisiones de los distintos expertos debido a su preparación y experiencia en el ámbito clínico, este gráfico se muestra en la figura 2.1.

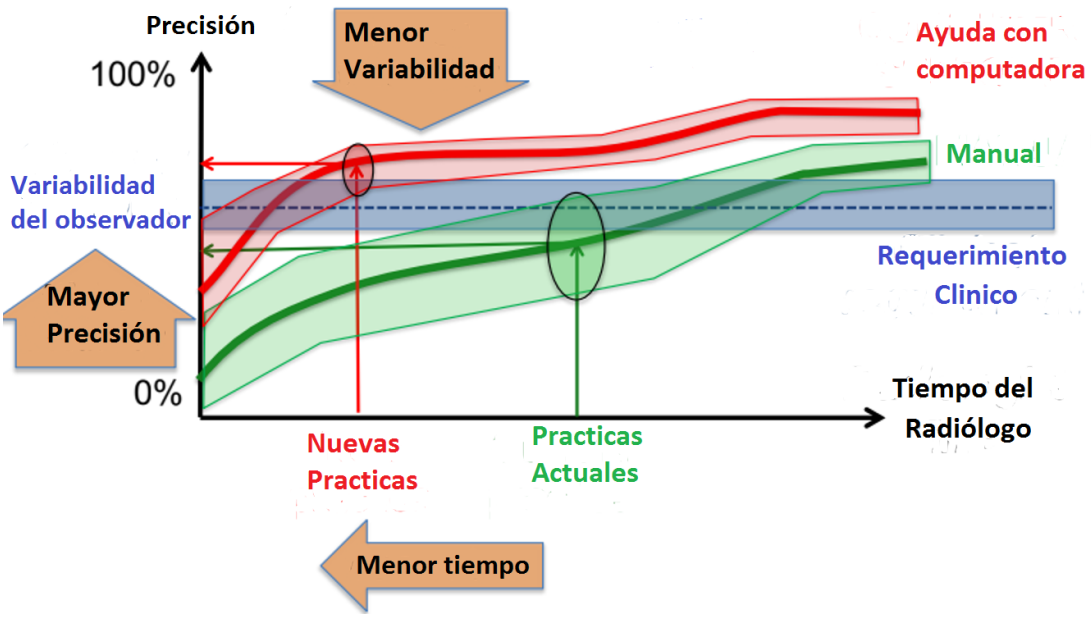


Figura 2.1: Precisión y variabilidad del diagnóstico. El uso de herramientas computacionales en el apoyo al diagnostico médico permite disminuir la variabilidad y el tiempo de este, así como alcanzar una buena precisión.

Las ventajas del desarrollo de estas herramientas de software permitirían no solo disminuir el tiempo que invierte un radiólogo en el proceso de diagnóstico médico, sino que

también permitiría alcanzar una mayor precisión. Estas herramientas también tienen la ventaja de facilitar la estandarización de estos procesos de evaluación, lo que resultaría en una disminución la variabilidad entre el diagnostico ofrecido por distintos expertos, así como facilitar el comienzo de una transición hacia un análisis más objetivo de estos datos.

Distintos métodos de segmentación han sido propuestas durante las últimas décadas, basándose cada vez técnicas más sofisticadas.

Una técnica propuesta en 1975 es la de umbralización, en la cual se toma un valor umbral de píxel, y los píxeles serán clasificados dependiendo si su valor está por encima o por debajo de este umbral, posteriormente se aplican operaciones morfológicas como erosión y dilatación con la finalidad de mejorar la calidad del segmentado.

Figura 2.2: Método de segmentación propuesto por N. Otsu et. al. 1975

También están los modelos activos de apariencia [29], en este caso se tiene un modelo geométrico predefinido, de algún órgano o estructura específica, que será ajustada de forma automática a una región de la imagen dependiendo de características como los valores de píxel.

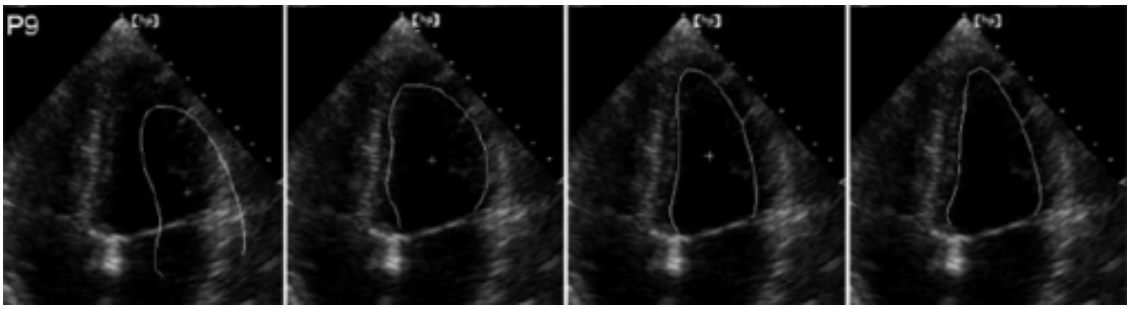


Figura 2.3: Método de segmentación implementado por J. Bosch et al 2002

Otra opción son los Contornos activos [34], algoritmos que se basan en contornos que pueden deformarse desde una posición inicial hasta una posición final, la evolución de este modelo está influenciada por fuerzas internas que imponen restricciones en la libertad de deformación de la curva y fuerzas externas que buscan minimizar una función de energía basada en las vecindades del contorno.

Existen muchos otros métodos de segmentación, sin embargo, en este trabajo se optó por emplear algoritmos de aprendizaje profundo, los cuales pertenecen al área del aprendizaje de maquina.

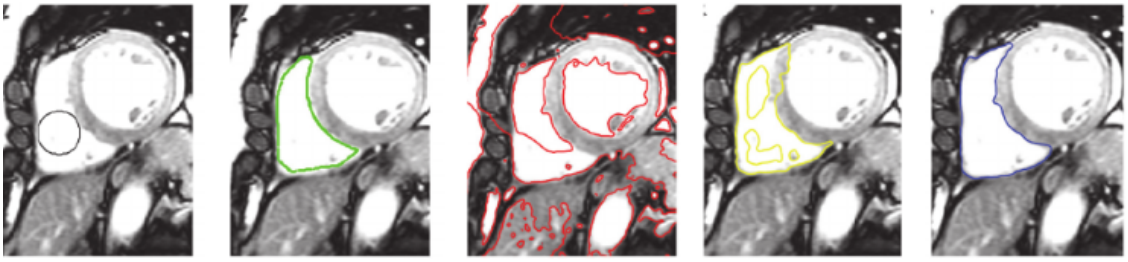


Figura 2.4: Método de segmentación implementado por Shafiullah et.al 2017.

2.1. Aprendizaje de Maquina

El aprendizaje de maquina (**Machine Learning**) o aprendizaje automático, es una rama de el campo de la Inteligencia Artificial, busca desarrollar modelos estadísticos basados en el reconocimiento automático de patrones,los algoritmos empleados aquí, tienen la característica de que permiten aprender a llevar a cabo una tarea a partir de un conjunto de datos u observaciones (experiencia). Tom M. Mitchell (1997) define el desarrollo de este tipo de algoritmos y el proceso de **aprendizaje** de la siguiente manera : 'Un programa de Computadora aprende de la experiencia E con respecto a alguna tarea T y con una medida de rendimiento P si su desempeño en la tarea T, medido por P, mejora con la experiencia E.' [12] El desarrollo de estos tipos de algoritmos se centran en determinar que métricas se pueden emplear para saber si la computadora esta mejorando o no y en dado caso, que es lo que debería cambiar para poder mejorar. En conjunto con lo anterior la variedad entre lo que puede ser considerado como tarea, una experiencia y una medida de desempeño da lugar a distintos tipos de aprendizaje, este trabajo se enfoco particularmente en el desarrollo de algoritmos de **Aprendizaje Supervisado**.

2.1.1. Aprendizaje Supervisado

En este tipo de aprendizaje se cuenta con un conjunto de datos u observaciones $X = \{x_1, x_2, ..., x_m\}$ y con un conjunto de etiquetas $Y = \{y_1, y_2, ..., y_m\}$, cada una asociada a una observación correspondiente. La idea de este tipo de aprendizaje es determinar la etiqueta que estaría asociada a una nueva observación x_k fuera del conjunto X, a partir de haber estimado la distribución condicional $p(y|x)$ para este tipo de datos, *i.e* que se busca generalizar para nuevas observaciones a partir de los datos conocidos. Adquiere este nombre porque en un inicio el etiquetado de las observaciones se lleva a cabo de forma manual por un 'instructor' o 'supervisor' y depende de estas etiquetas que es lo que el algoritmo va a aprender.

2.1.2. Estimación por Máxima Verosimilitud

El problema consiste entonces en estimar un modelo $p_{model}(y|x;\boldsymbol{\theta})$ con parámetros $\boldsymbol{\theta}$ que pueda representar con mayor fidelidad la distribución original $p(y|x)$. En lugar de adivinar alguna función que sea un buen estimador del modelo y posteriormente analizar su comportamiento, es posible recurrir a una herramienta matemática que nos permita generar funciones especificas que sean buenos estimadores para los diferentes modelos posibles.
La Estimación por Máxima Verosimilitud (EMV) es un método desarrollado en el área de estadística, que permite estimar los parámetros de una distribución probabilística, lo cual puede ser llevado a cabo mediante la maximización de una **función de verosimilitud**, de

modo que bajo el modelo estadístico propuesto, las observaciones obtenidas, representadas por los conjuntos X, Y sean las más probables. en este caso esto correspondería a que, dada una imagen x, el modelo genere o estime con la mayor probabilidad posible, la máscara que efectivamente le correspondería a esta imagen. Se trabaja con dos distribuciones diferentes, las cuales son $\widehat{p}_{data}(x)$ y $p_{model}(x;\boldsymbol{\theta})$, conformando la primera un modelo empírico, el cual es el resultado del conjunto de observaciones con el que se cuenta, ya que no se tiene acceso directo a $p(x)$ y solo se cuenta con un conjunto finito de observaciones de esta distribución, y por otro lado $p_{model}(x;\boldsymbol{\theta})$ es un modelo predefinido y que esta parametrizado por $\boldsymbol{\theta}$. Asociada con cada vector de parámetros $\boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_k]$ existe una distribución de probabilidad dentro de una familia parametrica $\{p_{model}(x;\boldsymbol{\theta})|\boldsymbol{\theta} \in \boldsymbol{\Theta}\}$ siendo $\boldsymbol{\Theta}$ el espacio de parámetros, un subconjunto del espacio Euclídeo.

El estimador de máxima verosimilitud para $\boldsymbol{\theta}$ puede definirse entonces como:

$$\boldsymbol{\theta}_{ML} = argmax_{\boldsymbol{\theta}} p_{model}(X;\boldsymbol{\theta}) \tag{2.1}$$

Siendo X el conjunto de observaciones $\{x_1, x_2, ..., x_m\}$

El lado derecho de la ecuación corresponde a la probabilidad conjunta de las observaciones y que bajo las condición de que estas observaciones sean consideradas **independientes e idénticamente distribuidas** (i.i.d), puede expresarse como:

$$\boldsymbol{\theta}_{ML} = argmax_{\boldsymbol{\theta}} \prod_{i=1}^{m} p_{model}(x^i;\boldsymbol{\theta}) \tag{2.2}$$

Una forma más conveniente pero equivalente de resolver este problema de optimización se obtiene al tomar el logaritmo de la función a optimizar, pues nos permite convertir el producto en suma y dado que esta es una función monótona creciente, el máximo ocurre para el mismo valor de $\boldsymbol{\theta}$, por lo que

$$\boldsymbol{\theta}_{ML} = argmax_{\boldsymbol{\theta}} \sum_{i=1}^{m} \log p_{model}(x^{(i)};\boldsymbol{\theta}) \tag{2.3}$$

Es posible entonces dividir por m, el número de observaciones, para obtener una nueva versión de la ecuación, la cual puede expresarse como el valor esperado de la distribución del modelo parametrizado, tomada con respecto a la distribución empírica definida por el conjunto de observaciones.

$$\boldsymbol{\theta}_{ML} = argmax_{\boldsymbol{\theta}} E_{x\sim\widehat{p}_{data}} \log p_{model}(x^{(i)};\boldsymbol{\theta}) \tag{2.4}$$

Una forma de interpretar la EMV es al verla como una forma de minimizar la diferencia que existe entre la distribución empírica $\widehat{p}_{data}$ y la distribución generada por el modelo propuesto p_{model}, una forma de medir la diferencia entre estas dos distribuciones es mediante la **divergencia de Kullback-Leibler** (K-L), que viene dada por:

$$D_{KL}(\widehat{p}_{data}||p_{model}) = E_{x\sim\widehat{p}_{data}}[\log \widehat{p}_{data}(x) - \log p_{model}(x;\boldsymbol{\theta})] \tag{2.5}$$

El primer termino al lado derecho de la ecuación tiene un valor constante, por lo que para disminuir la diferencia que hay entre las dos distribuciones probabilísticas, la única cantidad que sera necesario minimizar es:

$$-E_{x\sim\widehat{p}_{data}}[\log p_{model}(x;\boldsymbol{\theta})] \tag{2.6}$$

Este termino también es conocido como la **entropía cruzada** y la minimización de este, es análogo a la maximización de la ecuación (2.4).

La divergencia K-L tiene la propiedad de que dadas dos distribuciones P y Q

$$D_{KL}(P||Q) \geq 0 \tag{2.7}$$

Por lo que tiene un valor mínimo definido, el cual puede ser alcanzado cuando las distribuciones son idénticas, los que en este caso correspondería a obtener $\widehat{p}_{data} = p_{model}$.

En el caso de que la ecuación (2.6) describa una función convexa, y que esta sea diferénciale en $\boldsymbol{\theta}$, encontrar el mínimo se reduce al criterio:

$$\bigtriangledown_{\boldsymbol{\theta}}(-E_{x\sim\widehat{p}_{data}} \log p_{model}(x;\boldsymbol{\theta})) = \bigtriangledown_{\boldsymbol{\theta}} l(x;\boldsymbol{\theta}) = 0 \tag{2.8}$$

Donde

$$\bigtriangledown_{\boldsymbol{\theta}} = \left(\frac{\partial}{\partial\theta_1}, \frac{\partial}{\partial\theta_2}, ..., \frac{\partial}{\partial\theta_n}\right) \tag{2.9}$$

Esta ecuación puede llegar a tener una solución analítica, sin embargo, esto depende de la complejidad del modelo y muchas veces en necesario recurrir a otros métodos de optimización, como los métodos iterativos, un ejemplo de esto es el gradiente descendente, en el cual los parámetros del modelo pueden ser modificados a partir de la siguiente relación de recurrencia:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \epsilon \bigtriangledown_{\boldsymbol{\theta}} l(x;\boldsymbol{\theta}) \tag{2.10}$$

Donde ϵ se denomina regularmente como la **taza de aprendizaje**. El uso de las derivadas parciales permite tomar en cuanta el efecto que tiene cada uno de los parámetros en el resultado final y así modificar los parámetros para poder llevar a cabo el proceso de optimización. Es necesario notar que en el caso que se cumpla la condición de la ecuación (2.8), los parámetros dejaran de ser modificados.

2.2. Visión por Computadora

El campo de visión por computadora existe desde alrededor de 1960, toma ideas del sistema visual humano con el propósito de automatizar las misma tareas que este puede realizar, las cuales serían: adquirir, procesar, analizar y comprender imágenes.

Para el área de imagenología médica, los principales objetivos consisten en poder llevar a cabo la detección, localización y segmentación de distintas estructuras anatómicas o posibles patologías, el acercamiento principal para lograr esto se basa en la extracción y análisis de diferentes característica que pueden ser obtenidas de una imagen, particularmente en el área de la imagen médica estas características pueden tener diferentes niveles de complejidad, llegando a variar desde la geometría de la región anatómica de interés, el histograma de los valores de píxel, hasta el arreglo espacial especifico de ciertos valores de píxeles en la imagen (*texturas*). Estas características, también conocidas como descriptores pueden ser extraídos directamente de la imagen o después de haber aplicado diferentes filtros o transformaciones, lo que permite clasificarlas dentro de los siguientes grupos [35]:

- **Características de forma**: Describen la geometría de la región de interés (ROI), y propiedades como el volumen o la superficie, la longitud máxima entre diferentes

direcciones ortogonales, esfericidad, etc. Por ejemplo, cantidades como la razón entre volumen y la superficie de un tumor pueden ser de utilidad para determinar la progresión (estadiaje) que le corresponda.

- **Características estadísticas de primer orden**: Permiten describir la distribución de los valores de píxeles sin tomar en cuenta ningún tipo de información espacial. Estas cantidades se basan en el histograma de los valores de píxeles de una imagen, siendo algunas: la varianza, el promedio, los valores mínimo y máximo, la curtosis, la asimetría y la entropía.

- **Características estadísticas de segundo orden**: Estas incluyen las denominadas características de *textura*, las cuales pueden ser obtenidas calculando las interrelaciones estadísticas entre píxeles vecinos [4].

- **Características estadísticas de mayor orden**: Estas son obtenidas después de haber aplicado algún tipo de filtro o transformación matemática a las imágenes, por ejemplo, con el propósito de identificar patrones repetitivos, suprimir ruido o resaltar estructuras. Algunos ejemplos pueden ser las transformadas de Wavelet, Hermite, Gabor, etc.

Algunos ejemplos de algoritmos que se han empleado como herramientas para el desarrollo del campo de visión por computadora son: máquinas de soporte vectorial (SVM), arboles de decisión, análisis de componentes principales, Redes Neuronales Artificiales (ANN), Modelos estadísticos de forma, Contornos activos deformables, Redes Neuronales Convolucionales (CNN), Redes Neuronales Recurrentes (RNN), etc.

2.3. Aprendizaje Profundo

El aprendizaje profundo es un conjunto de algoritmos del área del aprendizaje de maquina (Fig. 2.5), que buscan generar representaciones complejas de los datos a partir de representaciones más sencillas. Este tipo de algoritmos se pueden representar como una estructura jerárquica de capas, donde para una tarea como la clasificación de imágenes, dados los píxeles de una imagen, la primera capa pude fácilmente detectar líneas al comparar valores de intensidad en píxeles vecinos, a partir de este resultado la siguiente capa pude buscar bordes y esquinas (intersecciones de estas rectas), a partir de aquí las siguientes capas pude detectar estructuras más complejas, como rostros u objetos, al encontrar la colección especifica de contornos que lo constituyen, esto se muestra en la figura 2.6.

Algunas de las principales ventajas de esta área en particular se presentan a continuación:

- El aprendizaje profundo tiene la característica de que permite 'aprender' directamente de los ejemplos, sin necesidad de llevar a cabo un pre-procesamiento para extraer de forma manual la características que sean más importantes para la tarea a realizar.

- Este tipo de algoritmos han superado el rendimiento de aquellos existentes hasta el momento.

- La cantidad de datos existente, así como la generada día a día aumenta cada vez más, lo que propicia la adopción de métodos que permitan llevar a cabo este aprendizaje basado en ejemplos.

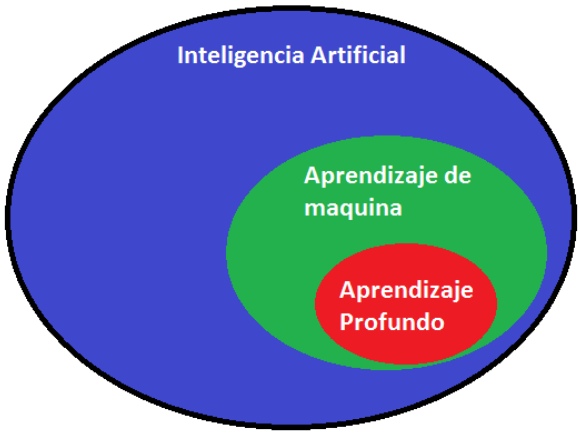


Figura 2.5: Campos de la Inteligencia Artificial.

- El continuo desarrollo de dispositivos con una alta capacidad y velocidad de computo, siendo algunos ejemplos las unidades de procesamiento gráfico (GPU), unidades de procesamiento tensorial (TPU) y chips especializados que se han estado desarrollando ha facilitado el acceso a este tipo de herramientas a un mayor publico.

Este tipo de algoritmos se basa en el uso de unidades individuales denominadas **neuronas artificiales**, representadas en la figura 2.7, las cuales pueden aceptar valores numéricos de entrada y llevar a cabo operaciones con ellos, llegando a ser estas suma, multiplicación o incluso transformaciones no lineales como: sigmoide, exponencial, logarítmica, rectificadora (ReLU), etc. En las técnicas de aprendizaje profundo estas unidades son empleadas en arreglos específicos que han sido desarrollados para llevar a cabo alguna tarea en particular.

Para poder llevar a cabo la extracción y aprendizaje de las características que serán empleadas para llevar a cabo la tarea especifica, se hace el uso de estructuras jerárquicas, las cuales permitan construir conceptos complejos definidos a su vez a partir de conceptos más sencillos, al representar esta estructura jerárquica en forma de gráfica se obtiene una serie consecutiva de capas, como se muestra en la figura 2.8, cuya profundidad pude relacionarse con la capacidad de llevara a cabo representaciones con un mayor nivel de complejidad. Las arquitecturas (o arreglos) pueden ser formadas con estas unidades son denominadas **Redes Neuronales Artificiales**. A los distintos niveles de esta estructura se les denomina *capas*, siendo la primera la *capa de entrada*, la ultima la *capa de salida* y la capas intermedias se denominan como *capas ocultas*.

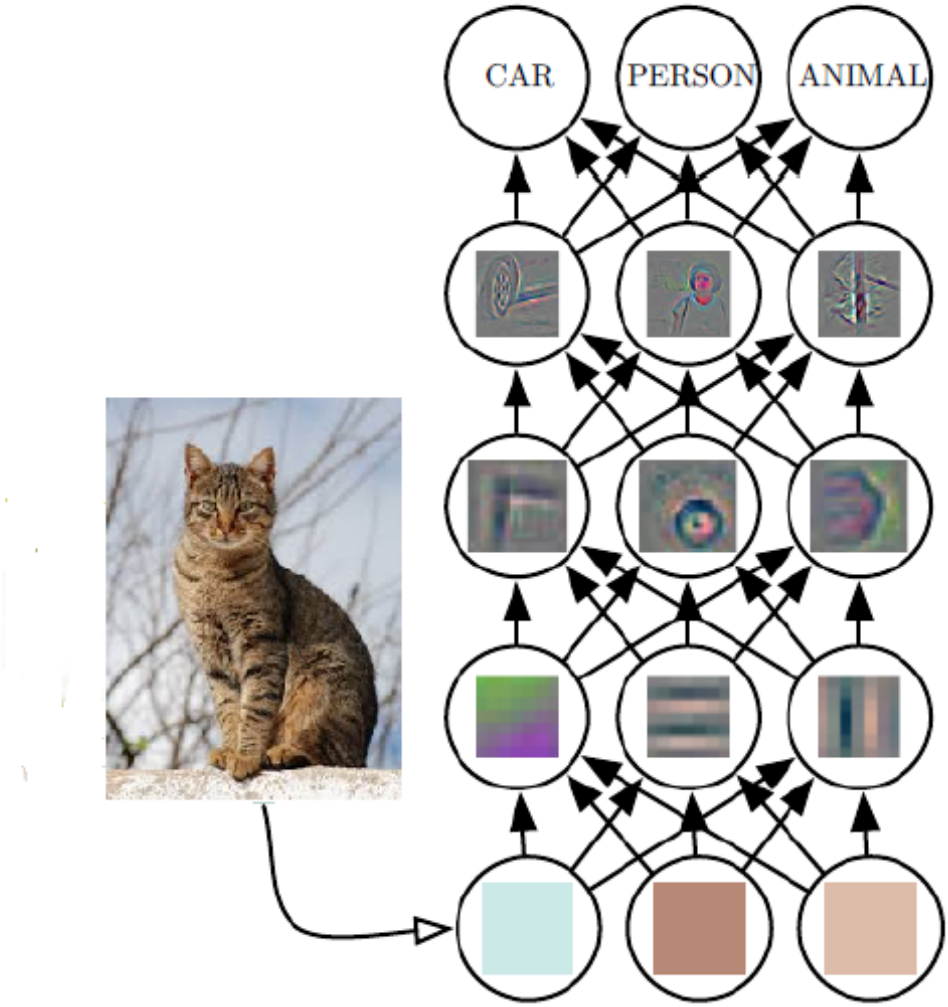


Figura 2.6: Aprendizaje profundo en el reconocimiento de imágenes.

2.3.1. Retro-Propagación

Las unidades operacionales de una arquitectura en particular pueden visualizarse como nodos en un gráfico computacional, cada uno de estos nodos con parámetros modificables que influyen en el resultado final u^n, de modo que para llevar a cabo el proceso de optimización es necesario tomar en cuenta el cambio en el resultado final que seria generado por el cambio en los parámetros de cada uno de estos nodos u^i, esto puede expresarse como:

$$\frac{\partial u^{(n)}}{\partial u^{(j)}} = \sum_{i:j} \frac{\partial u^{(n)}}{\partial u^{(i)}} \frac{\partial u^{(i)}}{\partial u^{(j)}} \tag{2.11}$$

donde i es un índice que corre desde n hasta j y la suma va sobre todos los nodos que se encuentren en una misma capa.

Una forma óptima de calcular las derivadas parciales necesarias para la optimización del modelo, es mediante el uso del algoritmo de retro-propagación, se dice de retro-propagación porque para llevar a cabo una predicción, la información fluye de la capa $i - sima$ a la capa $(i + 1)$ hasta obtener un resultado, mientras que para el calculo de la corrección a los

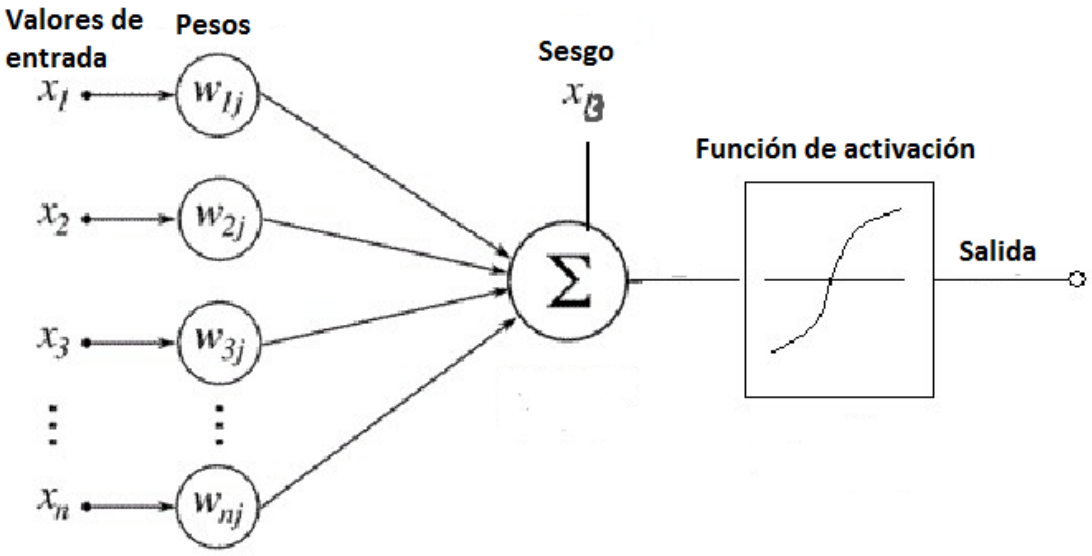


Figura 2.7: Representación de las operaciones llevadas a cabo por en una neurona artificial. Los parámetros aprendidos por este tipo de modelos son los **pesos** por los cuales se pueden multiplicar los valores de entrada, así como un valor de **sesgo** que se sumara directamente al resultado.

parámetros, la derivada de la $i-sima$ capa es empleada para el calculo de la derivada en la capa $(i-1)$, de modo que la información fluye en dirección contraria a la que normalmente lo haría, como se muestra en la figura 2.9.

El uso de este algoritmo permite evitar el uso cálculos redundantes y facilita la obtención de la derivadas con respecto a cada uno de los parámetros.

2.3.2. Redes Neuronales Convolucionales

En el área del aprendizaje profundo existe una gran variedad de arquitecturas de redes neuronales, cada una de ellas siendo especifica para alguna tarea en particular. En el caso de visión por computadora, las arquitecturas mayormente empleadas son las Redes Neuronales Convolucionales, este tipo de arquitecturas se diferencian de las demás debido al tipo de operación, ya que este tipo de operaciones se ajusta mejor a patrones de datos que tengan una estructura de tipo malla como es el caso de las imágenes 2D, además de que se da prioridad a las posibles interacciones que puedan existir entre los elementos de vecindades locales de los datos.

La operación de Convolución

La operación de convolución dadas dos funciones continuas $f(x), w(x)$ se define como:

$$s(t) = (f * w)(x) = \int f(x)w(t-x)dx \tag{2.12}$$

En este caso $w(x)$ se conoce como el kerner de convolución y $w(t-x)$ representa este mismo kernel pero con un offset de t con respecto a la función $f(x)$.

Por otro lado, una imagen se representa como un arreglo discreto tipo malla de valores, siendo estos el valor de píxel. La imagen puede estar descrita por tres dimensiones, representando cada una respectivamente el alto, ancho y numero de canales (e.j. rojo, verde y azul)

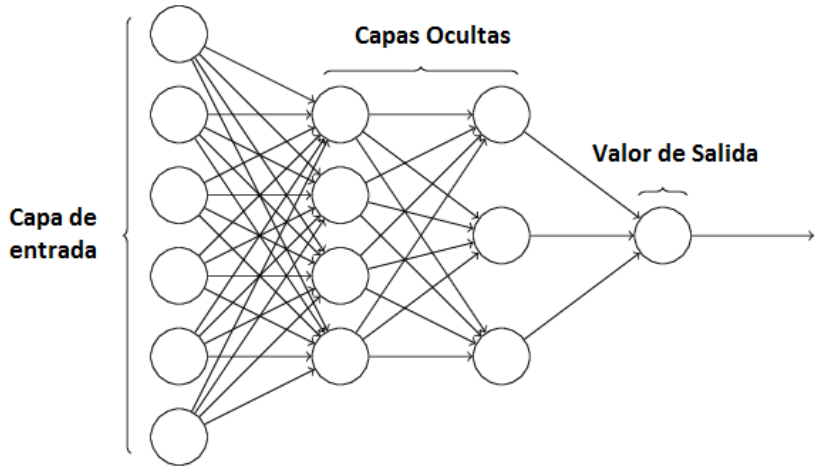


Figura 2.8: Representación gráfica de una Red Neuronal conformada por distintas capas y cada una de estas con un número determinado de neuronas artificiales.

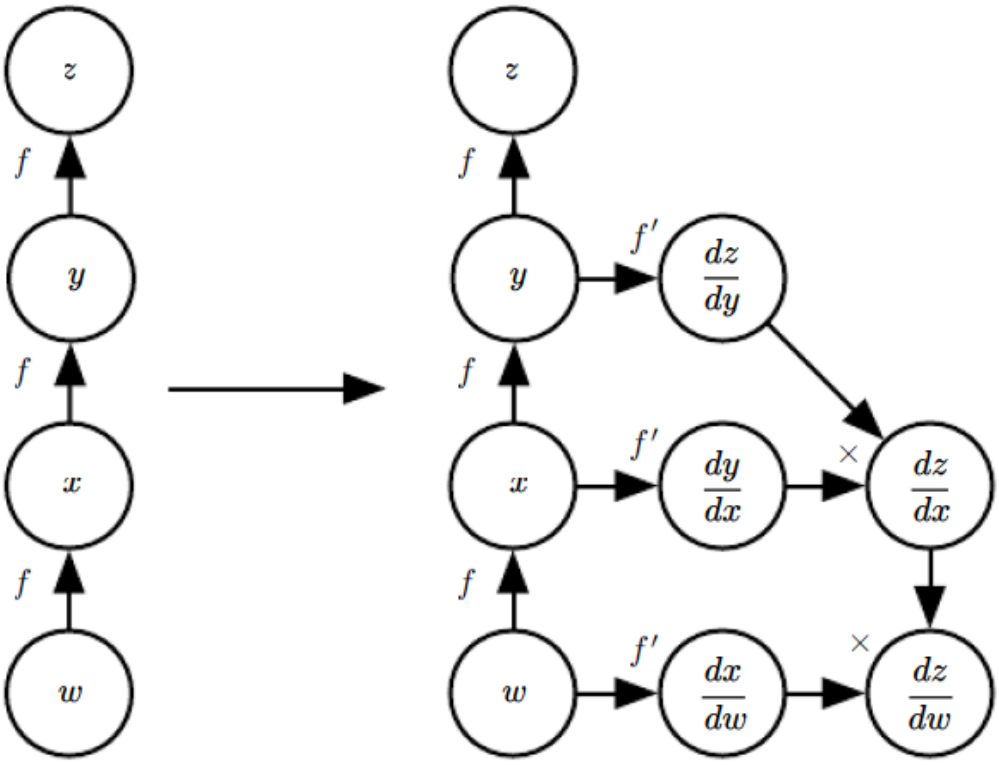


Figura 2.9: Representación gráfica algoritmo de retro-propagación.

de la imagen, de modo que si se considera una imagen como una función sobre la cual se puede llevar a cabo una operación de convolución, la operación podría representarse como:

$$Z_{i,j} = (V * K)_{i,j} = \sum_{m,n,l} V_{i,j,k} K_{i-m,j-n,k-l} \tag{2.13}$$

Donde $Z_{i,j}$ representa el renglón i, columna j del arreglo resultante de aplicar la convolución del kerner K sobre la imagen V, ambos con k número de canales. Esta es la forma convencional en la que se lleva a cavo la operación en el campo del aprendizaje profundo. Al aplicar un mayor número de kernels de convolución, el número de canales del arreglo Z resultante aumentara en la misma cantidad.

Esta operación también puede representarse de forma gráfica para una imagen inicial y una nueva imagen resultante con dimensiones:

- N_m: Altura y anchura inicial de la Imagen
- D_m: Número de canales de la Imagen inicial y del kernel
- Q_m: Altura y anchura del kernel
- N_{m+1}: Altura y anchura final de la Imagen
- D_{m+1}: Número de canales de la Imagen final y número de kernels empleados

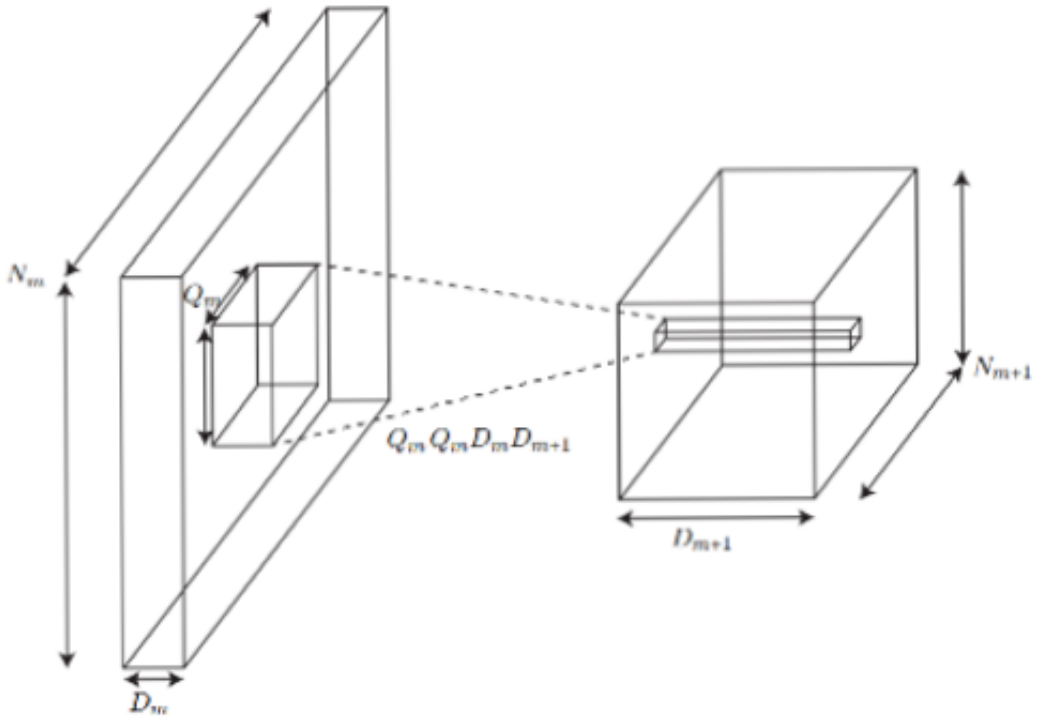


Figura 2.10: Representación grafica de la operación de convolución en un tensor.

De modo que el numero de operaciones que se llevan a cabo para pasar del arreglo **m** al arreglo **m+1** seria:

$$N = Q_m^2 D_m D_{m+1} N_{m+1}^2 \tag{2.14}$$

Redes Neuronales Convolucionales desde una base biológica

Este tipo de arquitecturas toma inspiración de modelos como el 'Neurocognitron' [10], el cual a su vez se basa en modelos propuestos por David Hubel y Torsten Wiesel, quienes colaboraron por muchos años [8] para determinar algunos de los mecanismos de funcionamiento del sistema visual en mamíferos como el mono [16] y el gato [15]. En 1981, su trabajo fue reconocido con el Premio Nobel en Fisiología o Medicina

Sus trabajos describen principalmente el sistema de visión como una estructura jerárquica conformada por células simples y células complejas las cuales son las encargadas de procesar las imágenes una vez estas eran adquiridas, siendo las células simples, encontradas en las primeras etapas de esta estructura, las cuales respondían a patrones bien definidos como bordes, esquinas y barras como dimensiones y orientaciones especificas, que las representaciones de objetos con un mayor grado de complejidad, como rostros, estructuras u otros objetos pueden ser formados a partir de componentes más simples como lineas, intersecciones de estas, etc, principio en el que se basa el desarrollo de las Redes Neuronales Convolucionales.

Capítulo 3

Metodología

El objetivo de esta practica fue desarrollar algoritmos basados en aprendizaje profundo que permitan llevar a cabo la segmentación de regiones anatómicas especificas, *i.e*, que se generara una nueva imagen donde cada valor de píxel representara la pertenencia a alguna clase en particular. Para lograr esto se tomó como base el trabajo llevado a cabo por Ronneberger Olaf et al.[31], desarrollado en 2015. Se llevó a cabo el entrenamiento de esta red, así como una posterior comparación de su desempeño con una variante de la arquitectura en la cual se buscó alcanzar una mayor eficiencia.

3.1. Base de Datos

Las imágenes junto con sus etiquetas fueron tomadas de un conjunto de datos de uso libre [28], el cual consiste de imágenes correspondientes en un cierto volumen a nivel del tórax, estas fueron adquiridas por Resonancia Magnética Cardiovascular durante una práctica clínica en el Hospital de Niños de Boston, Boston, MA,USA. Los casos incluyen una variedad de enfermedades congénitas del corazón, algunos de los pacientes has sido sometidos a intervenciones.

Las posibles clases a las que puede ser asignado cada píxel son:

- Volumen de Sangre: Esta clase incluye las aurículas izquierda y derecha, los ventrículos izquierdo y derecho, y se extiende unos pocos centímetros fuera del origen de la aorta, venas pulmonares, vena cava superior e inferior y arterias pulmonares.
- Miocardio: La clase asociada al miocardio incluye la pared muscular que rodea los dos ventrículos y el septum que los separa.
- Fondo: A esta clase se asignó todo píxel que no correspondiera a las primeras dos.

En este caso, la tarea consiste en determinar la categoría a la que pertenece cada píxel de la imagen, dada la imagen entera como observación, de modo que la distribución de probabilidad vendría representada como $p(y|x)$ donde x corresponde a la imagen médica, mientras que y corresponde a la **mascara de segmentación**, una nueva imagen donde cada píxel consistirá ahora de un vector de tres entradas, cada una asociada a la probabilidad de que el píxel corresponda a una categoría en particular (fondo, volumen sanguíneo, pared ventricular).

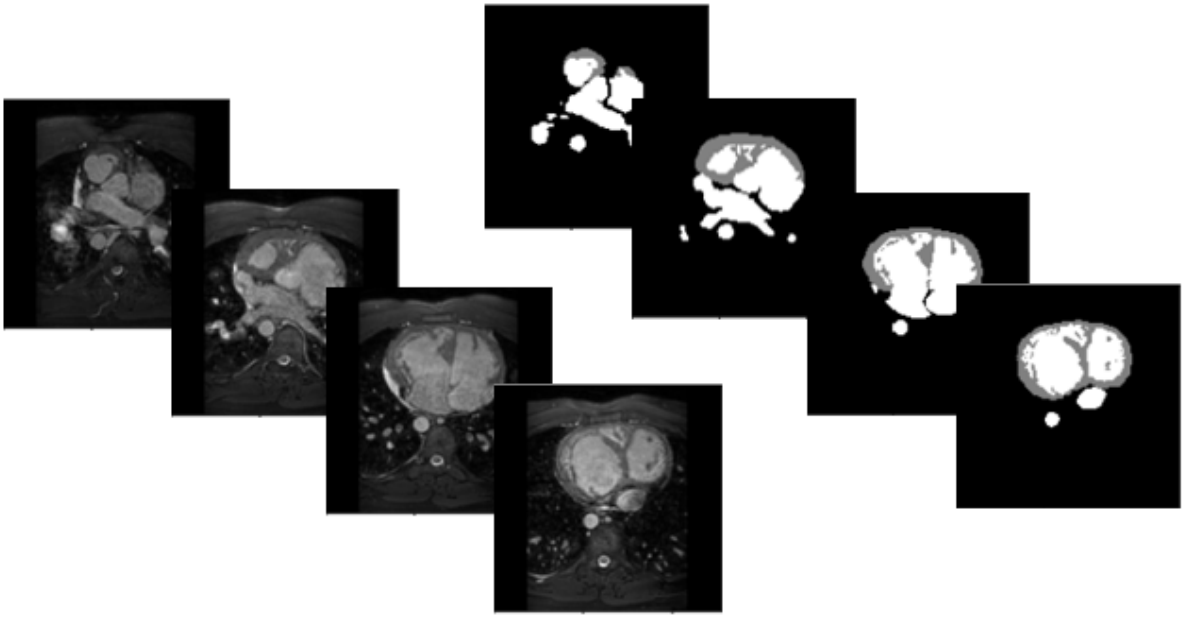


Figura 3.1: Ejemplos de algunas Imágenes que podían encontrarse en el conjunto de datos [28].

3.1.1. Partición de los datos

Una practica común en el aprendizaje de maquina es dividir el conjunto datos en tres sub-conjuntos mutuamente excluyentes, los cuales son:

- **Entrenamiento**: En este sub-conjunto se encontraran los datos u observaciones que serán empleados para llevar a cabos las modificaciones a los parámetros de nuestro modelo.
- **Validación**: Los datos en este sub-conjunto no son empleados directamente para la optimización de los parámetros del modelo, sino que son empleados para evaluar durante el proceso de entrenamiento el desempeño que tiene el modelo con datos que no se le habían presentado antes.
- **Evaluación**: Los datos en este sub-conjunto están completamente desacoplados del proceso de entrenamiento y son empleados para evaluar el desempeño final del modelo.

En cada caso es posible cuantificar un error entre el resultado esperado y el obtenido a partir del modelo, siendo este el *error de entrenamiento* para el conjunto de entrenamiento y el *error de generalización* para los conjuntos de validación y evaluación.

El conjunto de datos consiste de nueve volúmenes de imágenes, cada uno de un paciente diferente, sumando un total de 2348 imágenes, con sus respectivas mascaras de segmentación.

En este caso se opto por una partición aleatoria de cada uno de los primeros ocho volúmenes, con una relación 80-20, para generar el conjunto de entrenamiento y el de validación, esto se llevo a cabo usando el método de *train test split*, de la biblioteca de **Scikit-Learn**, con una valor para la semilla de inicialización de 1. El conjunto de evaluación de conformó de todas las imágenes del noveno volumen. Como resultado se obtuvieron 1535 imágenes de entrenamiento, 389 imágenes de validación y 212 imágenes de evaluación.

3.1.2. Pre-procesamiento

Aumento de Datos

Se llevo a cabo un proceso de Aumento de datos ('data augmentation') en el cual se aplicaron transformaciones afines como desplazamiento, rotación, magnicación y shearing a las imágenes y mascaras en conjunto, estas nuevas imágenes eran posteriormente introducidas a la red para su entrenamiento, esto se hace con la finalidad de aumentar el número de ejemplos disponibles para llevar a cabo el entrenamiento del modelo, así como para volverlo robusto ante variaciones especificas en las imágenes de entrada, las cuales pudieran ser ocasionadas por una variación en la calidad de la imagen, o en las proporciones anatómicas del paciente. Estas transformaciones fueron llevadas a cabo de manera aleatoria sobre las imágenes de entrenamiento y de validación a medida que estas eran ingresadas al modelo para llevar a cabo el proceso de entrenamiento, lo que se logró usando la clase *Image Data Generator* de la biblioteca de **Keras**.

los parámetros empleados para llevar a cabo las transformaciones sobre las imágenes fueron:

- Rango de Rotación: 10 grados
- Traslación lateral: $\pm 5\,\%$ de la imagen
- Traslación vertical: $\pm 5\,\%$ de la imagen
- Rango de shearing = 0.02
- Rango de Zoom = $\pm 15\,\%$
- Imagen horizontal en espejo

Este procedimiento solo se llevó a cabo en la imagen médica y en el mapa de segmentación en conjunto.

Normalización de la Imagen

A diferencia de protocolos de adquisición de imágenes médicas como la Tomografía Axial Computarizada, en el que se cuenta con una escala definida para los valores de cada vóxel, conocida como escala de unidades Hounsfield, donde cada región anatómica esta caracterizada por un rango particular de valores en esta escala, la adquisición de imágenes por Resonancia Magnética presenta una mayor variación en los valores de vóxel que pueden ser obtenidos y la evaluación de regiones anatómicas es llevada con un enfoque más relativo a los valores de vóxel de las regiones vecinas a la de interés. Debido a la variabilidad presente en los datos, se opto por normalizar cada imagen para que tuviera media de cero y desviación estándar de uno, al igual que en [38, 27].

$$\boldsymbol{I} = \frac{\boldsymbol{I} - \mu}{\sigma} \tag{3.1}$$

Donde la resta y división se lleva a cabo sobre cada uno de los vóxeles de la imagen, restringiendo así el rango de variabilidad entre los datos de entrada y facilitando el proceso de aprendizaje. Este procedimiento solo se llevó a cabo en la imagen medica.

3.1.3. Conversión a variable categórica.

La segmentación de una imagen consiste en asignar cada uno de los píxeles de la imagen a alguna de las posibles clases, para esto es necesario manejar la probabilidad de que pertenezca a cada una de ellas. El píxel es asignado a la clase con mayor probabilidad. A continuación se muestra una representación de la probabilidad que tiene un píxel dado de pertenecer a alguna clase en particular. En la figura 3.2 se muestra, del lado izquierdo la distribución categórica de probabilidades que seria generada por la distribución empírica de los datos, mientras que del lado derecho se muestra la distribución que seria generada, dada una imagen, por un modelo parametrizado, la idea durante el proceso de optimización seria modificar los parámetros del modelo para minimizar la diferencia que hay entre estas dos distribuciones.

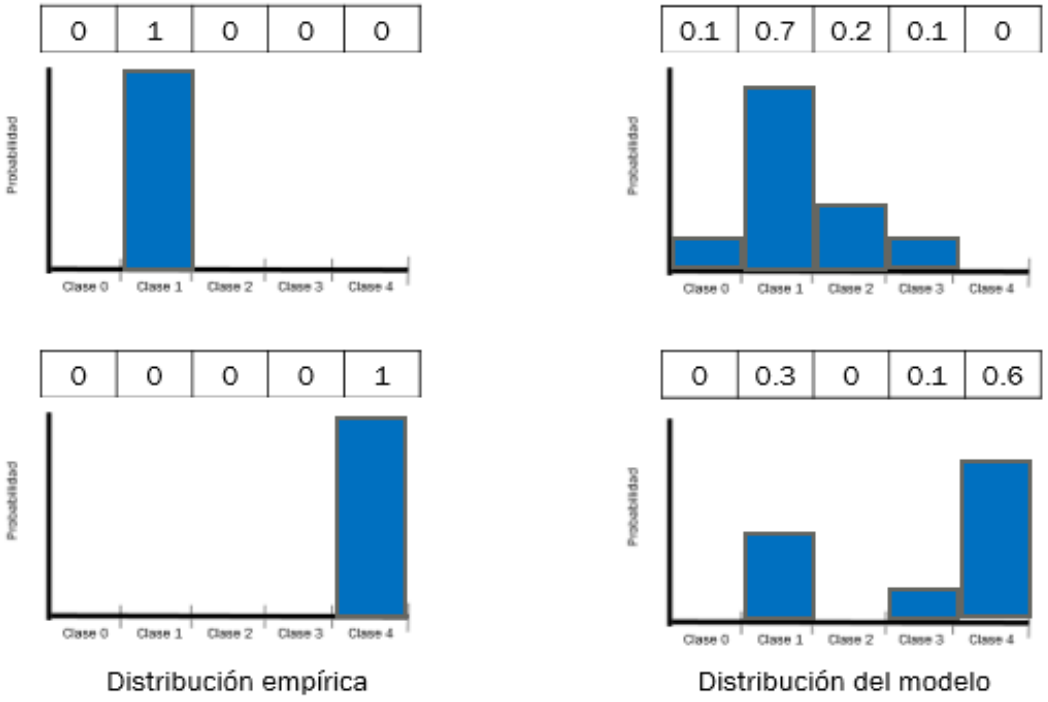


Figura 3.2: **Izquierda)** Distribución de probabilidad verdadera de que el píxel pertenezca a una de las clases, **Derecha)** Distribución de probabilidad generada por el modelo.

Para formar esta distribución categórica, se tomaron los mapas de segmentación de la imagen y se separaron en tres canales binarios, uno para cada clase, donde el valor del píxel en cada canal representa la probabilidad de pertenecer a esa clase en particular, esto se muestra en la figura 3.3.

3.2. La Arquitectura

Esta red se basa en una arquitectura tipo autocodificador ('encoder-decoder'), la cual esta constituida por dos etapas, la primera etapa siendo la de codificación ('encoding'), la cual consiste en series consecutivas de operaciones de convolución seguidas cada una por un proceso particular de reescalamiento ('downsampling'), esta ultima generalmente disminuye las dimensiones de la imagen a la mitad y es llevado a cabo mediante la aplicación de filtros

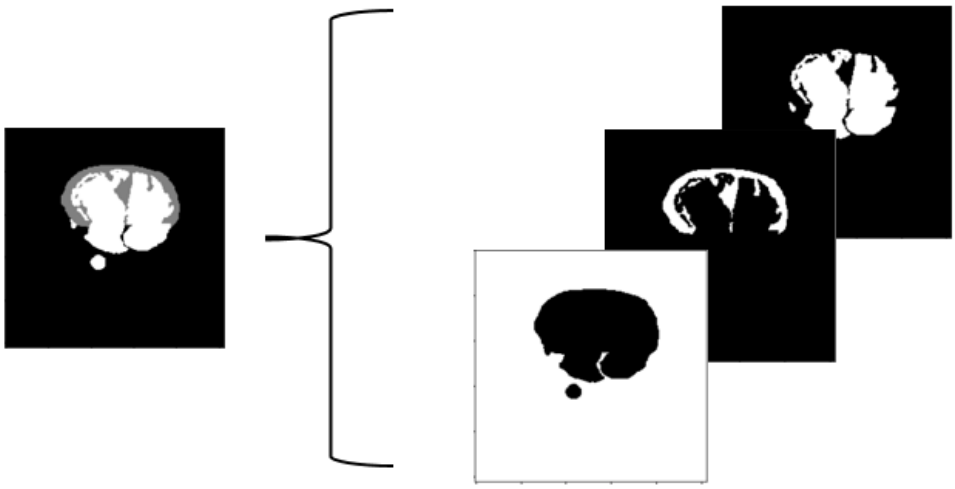


Figura 3.3: Mascara de segmentación separada en tres canales, cada uno correspondiente a una clase en particular.

no lineales como max-pooling. La etapa de codificación permite extraer las características principales de la imagen al mismo tiempo que va disminuyendo las dimensiones de esta, lo que permite extraer continuamente las características en las diferentes escalas, eliminar las variables que no contribuyan al proceso de segmentación y mantener aquellas que tengan una mayor importancia para esta tarea, alcanzando así representaciones más complejas e informativas de los datos. En el trabajo original la imagen inicial pasa por cuatro etapas de codificación (lo que corresponde a una disminución en las dimensiones de la imagen por una razón de 2^4), hasta llegar a lo que se conoce como cuello de botella, el nombre se debe a que es la etapa donde la información se encuentra más compacta.

Lo siguiente es una etapa de decodificación ('decoding'), la cual consiste nuevamente en series consecutivas de operaciones de convolución, seguidas cada una por un proceso particular de reescalamiento ('upsampling'), pero en esta ultima generalmente se duplican las dimensiones de la imagen, lo que es llevado a cabo mediante la aplicación operaciones como la deconvolución o convolución transpuesta (Esto también puede llevarse a cabo mediante métodos como interpolación bilineal, cubica, etc...). La ventaja del uso de la operación de deconvolución en el proceso del reescalamiento, es que los valores del kernel que sera empleado también serán generados durante el proceso de entrenamiento. Este proceso de decodificación permite tomar la nueva representación de los datos generada a partir de las características extraídas en la etapa de codificación y gradualmente ir formando el mapa de segmentación correspondiente a la imagen inicial.

El proceso de codificación y posterior decodificación tiene la ventaja de permitir analizar la imagen en distintas resoluciones, sin embargo esta disminución en la resolución puede generar una perdida de los detalles o estructuras más pequeñas de la imagen, de modo que con la finalidad de solucionar esto se emplean conexiones saltadas ('skip conections'), las cuales permiten conectar la información de la parte de codificación a la parte de decodificación antes de llegar al cuello de botella, de modo que las características extraídas de la imagen en sus distintas resoluciones y representaciones pueden ser recuperadas para influir en el proceso de generación de la mascara de segmentación.

Como entrada a este modelo se da la imagen médica, de un solo canal, y como salida se

obtiene una imagen de tres canales, que corresponde al mapa de segmentación con el primer, segundo y tercer canal correspondiendo a las clases de fondo, volumen sanguíneo y musculo ventricular respectivamente.

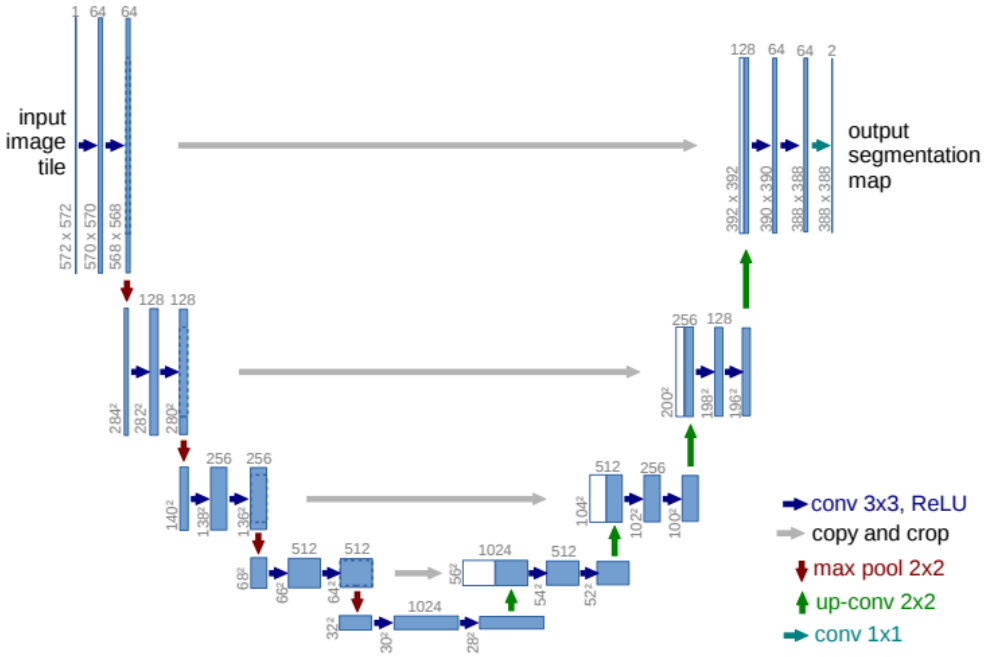


Figura 3.4: Arquitectura de la red U-NET.

Esta arquitectura se ha empleado en otros trabajos y se le han realizado modificaciones particulares en cada uno de ellos, con la finalidad obtener el mejor desempeño para el problema que buscaban resolver. [6, 41, 19, 42, 17, 26].

En este proyecto se llevo a cabo una comparación entre la arquitectura original y una variante de esta, a la nueva arquitectura se le llevaron a cabo modificaciones tomando en cuenta los trabajos llevados a cabo en [6, 7, 33, 37, 14], estas modificaciones consisten en la sustitución de muchas de las operaciones de convolución convencional por un nuevo tipo de operación denominada convolución separable, así como la incorporación de etapas para controlar las características de las distribuciones de datos que maneja el modelo, como Batch Normalization [18], y métodos de regularización durante el proceso de entrenamiento, como 2D Spatial Dropout [37].

Convolución separable

Las diferencia principal entre la arquitectura empleada en este trabajo y la original consiste en el uso de convoluciones separables, esta operación permite desacoplar el proceso de aprendizaje de la correlación entre píxeles vecinos y entre los píxeles de distintos canales. La sustitución de la convolución convencional por la convolución separable, o su uso en conjunto con esta ha demostrado una mejora en el desempeño de las Redes Neuronales Convolucionales, mostrando no solo una disminución en el número de parámetros empleado y operaciones realizadas, si no que también mejorando la precisión del modelo, volviéndolo así más eficiente [6, 7, 33, 37, 14].

La convolución separable consiste en separar el proceso de convolución en dos etapas, como se ilustra en la figura 3.5.

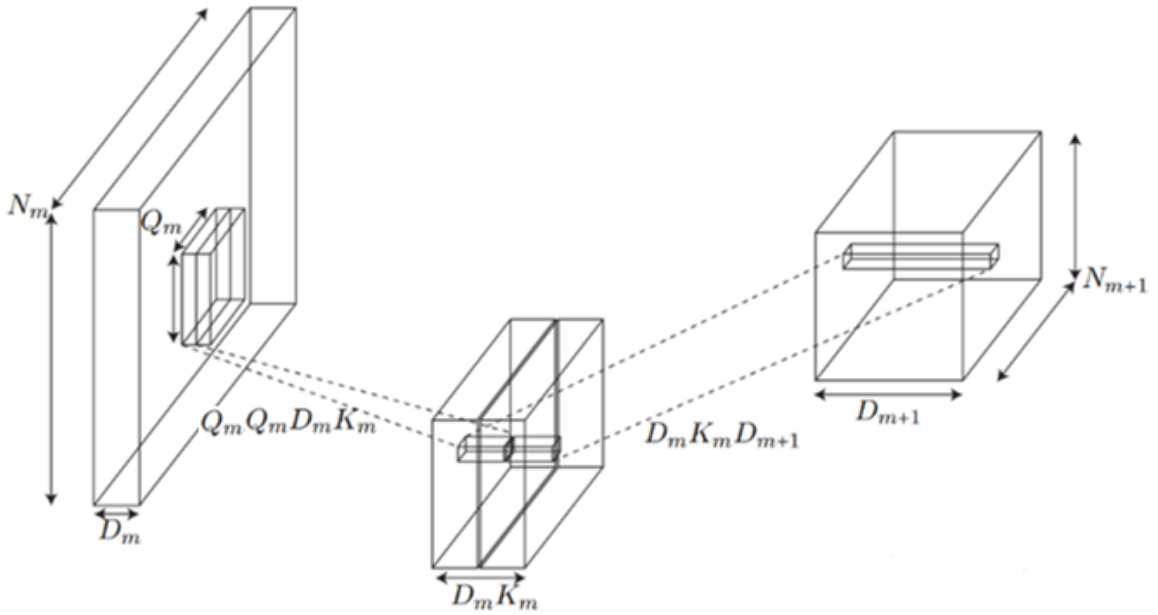


Figura 3.5: Convolución separable [33]

- **Convolución por canal** (Depthwise convolution): Esta es la primera etapa de la convolución separable. En este caso, la convolución no ocurre sobre las tres dimensiones de la imagen, si no que ocurre en cada canal por separado y se tiene tantos kernels bidimensionales como canales en la imagen de entrada. $K_{i,j,k}$ representa el elemento del renglón i, columna j del k-ésimo kernel, mientras que $G_{i,j,k}$ representa el elemento del renglón i, columna j del k-ésimo canal del arreglo resultante. Esta primera etapa puede representarse con la ecuación:

$$G_{i,j,k} = (V * K)_{i,j,k} = \sum_{m,n} V_{i,j,k} K_{i-m,j-n,k} \tag{3.2}$$

  Por lo tanto, el número de multiplicaciones que son llevadas a cabo durante esta primera etapa de la convolución separable ilustrada en la Figura 3.5 vendría dada por:

$$N_D = Q_m^2 D_m K_m N_{m+1}^2 \tag{3.3}$$

- **Convolución puntual** (Pointwise convolution):

  Esta es la segunda etapa de la convolución separable. En este caso, la convolución ocurre sobre la dimensión asociada a los canales de la imagen. $H_{i,j,k}$ representa el elemento del renglón i, columna j, canal k-ésimo kernel, mientras que $G_{i,j,k}$ representa el elemento del renglón i, columna j del k-ésimo canal del arreglo inicial y el arreglo resultante solo tiene dos dimensiones. Esta segunda etapa puede representarse con la ecuación:

$$Z_{i,j} = (G * H)_{i,j} = \sum_{l} G_{i,j,k} H_{i,j,k-l} \tag{3.4}$$

De lo anterior que el número de multiplicaciones que son llevadas a cabo durante esta segunda etapa de la convolución separable ilustrada en la Figura 3.5 vendría dada por:

$$N_P = K_m D_m N_{m+1}^2 D_{m+1} \tag{3.5}$$

El número total de operaciones que se llevan a cabo en la convolución separable seria entonces:

$$N_S = N_D + N_P = (Q_m^2 + D_{m+1}) D_m N_{m+1}^2 K_m \tag{3.6}$$

Comparación con la operación estándar de convolución

Es posible comparar cualitativamente la modificación en el costo computacional que implica la sustitución de la operación de convolución.

Convencionalmente se toma $K_m = 1$, de modo que:

$$\frac{N.mul.conv.separable}{N.mul.conv.convencional} = \frac{(Q_m^2 + D_{m+1}) D_m N_{m+1}^2}{Q_m^2 D_m D_{m+1} N_{m+1}^2} \tag{3.7}$$

$$\frac{N.mul.conv.separable}{N.mul.conv.convencional} = \frac{(Q_m^2 + D_{m+1})}{Q_m^2 D_{m+1}} \tag{3.8}$$

$$\frac{N.mul.conv.separable}{N.mul.conv.convencional} = \frac{1}{Q_m^2} + \frac{1}{D_{m+1}} \tag{3.9}$$

$$\frac{No.params.separable.conv.}{No.params.standard.conv.} = K_m \left(\frac{1}{Q_m^2} + \frac{1}{D_{m+1}} \right) \tag{3.10}$$

En esta ultima ecuación se obtiene la razón entre el número de multiplicaciones que se llevan a cabo en la convolución separable con respecto a la convolución convencional, que coincide con la razón entre el número de parámetros (sin contar los parámetros de sesgo) de ambas operaciones, de modo que cuando $D_{m+1} > 1$ tanto el número de operaciones como de parámetros sera menor en el caso de la convolución separable.

3.3. Función de Salida

Como salida de la red se obtiene una imagen de tres canales, cada canal esta asociado a una categoría distinta, correspondiendo el primer canal al fondo, el segundo canal a la pared ventricular y el tercer canal al volumen sanguíneo. Para una posición de píxel en particular, el valor que tenga este en cada uno de los canales, corresponderá a la probabilidad de que el píxel pertenezca esa categorías en particular, i.e que un píxel pude representarse como un vector $\widehat{\boldsymbol{y}}$ con entradas $\widehat{y}_i = P(y = i|x)$, esto describe una **distribución categórica**. Por otro lado al ser deseable una distribución sobre la probabilidad de pertenecer a una categoría en particular, se debe de cumplir que:

$$\sum_{i}^{N} P(y_i|x) = 1 \tag{3.11}$$

Es decir que la probabilidad esté normalizada. Una forma que se ha demostrado útil para realizar esto es empleando la función *softmax*:

$$softmax(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \tag{3.12}$$

Donde z_k representa el valor de la probabilidad (sin normalizar) de pertenecer a la categoría k. Esta re-normalización es lo que nos permite continuar usando el principio de máxima verosimilitud para encontrar el modelo de los datos.

3.4. Función de Costo

Como se menciono en la sección de Introducción, el desarrollo de algoritmos de aprendizaje de maquina nos permite aproximar la distribución de una variable dada a partir del conjunto de observaciones o datos de entrenamiento, y el principio de máxima verosimilitud proporciona una forma de cuantificar la diferencia entre la distribución empírica generada por los datos observados y la distribución aprendida por nuestro modelo. También se llegó al resultado de que la maximización de la función de verosimilitud era un proceso análogo a la minimización de la entropía cruzada. Para una distribución condicional que permite clasificar en C categorías diferentes, la función de entropía cruzada es:

$$CCE = -\sum^{C}\sum_{i}^{N} \widehat{p}_{data}(y_i|x_i)[\log p_{model}(y_i|x_i)] \tag{3.13}$$

La entropía cruzada, o bien entropía categórica cruzada, como seria conocida en este caso es una función que nos permite determinar la diferencia entre dos distribuciones, a su vez, existen muchas otras funciones de costo diferentes, las cuales son empleadas para modificar los parámetros del modelo. Una de las funciones de costo comúnmente empleadas en el área de visión por computadora esta asociada con la cantidad conocida como **coeficiente de Dice**, el cual es una medida de que tanto se traslapan dos estructuras sobrepuestas, lo que permite medir la similitud entre el mapa de segmentación generado por el modelo y aquel generado por un médico experto. El cálculo de este coeficiente se representa gráficamente en la figura 3.6 y mediante la ecuación:

$$DC = \frac{2\sum_N^i p_{model}(y_i|x_i)\widehat{p}_{data}(y_i|x_i)}{\sum_N^i p_{model}(y_i|x_i)^2 + \sum_N^i \widehat{p}_{data}(y_i|x_i)^2} \tag{3.14}$$

Con $0 \geq DC \geq 1$, donde 1 representa una superposición exacta entre las estructuras.
La función de costo para el coeficiente de dice seria entonces:

$$DL = 1 - DC \tag{3.15}$$

Como función de costo total se tomó la suma de la entropía categórica cruzada y el error en el coeficiente de Dice:

$$Loss = CCE + DL \tag{3.16}$$

Esto pues, la entropía categórica cruzada es una función general que surge de manera natural a partir del principio de máxima verosimilitud, que permite llevar a cabo la tarea de clasificación y que ofrece una opción robusta para emplear en el aprendizaje de maquina,

mientras que el coeficiente de dice ofrece una opción para incorporar la información de la calidad del mapa de segmentación de manera global.

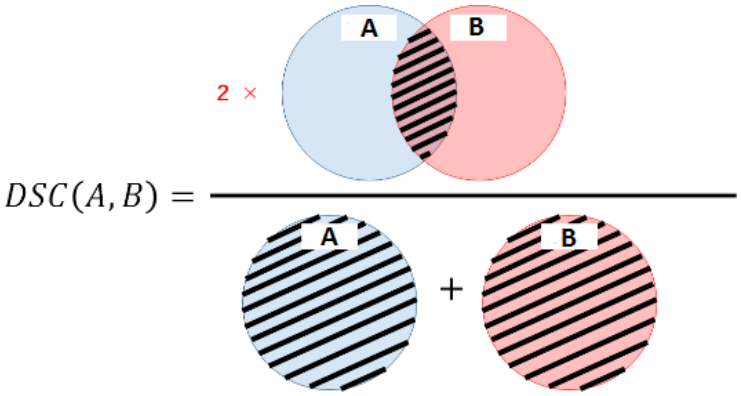


Figura 3.6: Representación gráfica del coeficiente de Dice.

3.5. Entrenamiento

El entrenamiento se llevó a cabo en una unidad de GPU, empleada en la plataforma de Google Colab. La evaluación del desempeño final en un conjunto de prueba fue llevada a cabo empleando el coeficiente de dice, el cual se calcula tomando en cuenta las tres categorías: fondo, volumen sanguíneo, paredes ventriculares.

GPU NVIDIA	Version CUDA	GPU RAM	chip CPU	Max. tiempo de sesión
Tesla K80	10.0.130	12	Intel Xeon 2.30GHz	12 horas

Tabla 3.1: Información relativa al ambiente virtual de trabajo.

Las características del entorno virtual empleado son las siguientes:

- Tensorflow Versión 2.0
- Keras Versión 2.2.4
- Back-end de Tensorflow

Keras y Tensorflow

Keras es una biblioteca de código abierto para el desarrollo de Redes Neuronales Artificiales escrita en Python. Esta biblioteca es empleada como interfaz de programación de aplicaciones, es de carácter modular, de modo que la construcción de un modelo consiste en conectar bloques configurables entre si, esto tiene la finalidad de acelerar y facilita el desarrollo y la evaluación de distintos modelos. Keras puede trabajar sobre Tensorflow, Theano y Microsoft Cognitive Toolkit, siendo estas las opciones de back-end.

Tensorflow es una biblioteca de código abierto desarrollada por el equipo de Google Brain, se basa en el uso de calculo simbólico a partir de la generación de grafos computacionales donde se registra el flujo de datos entre operaciones, esto a su vez permite llevar a cabo el

proceso de auto-diferenciación, esto es de gran utilidad para el desarrollo y optimización de modelos de Redes Neuronales Artificiales.

Estrategia de Optimización

Hay diferentes estrategias de optimización basadas en el gradiente descendente, dos de las que han mostrado un buen desempeño en una gran variedad de trabajos son RMSPorp y Adam. En este caso la estrategia empleada fue Adam ('Adaptative moments'), estrategia que ha mostrado un buen desempeño en el proceso de optimización de funciones no convexas [20, 30], y la cual hace una combinación de RMSProp y Momento, que consisten respectivamente en el uso de una taza de aprendizaje adaptativa y una versión exponencialmente ponderada de los valores del gradiente obtenidos anteriormente (momento), esto se lleva a cabo con la finalidad de acelerar y facilitar la convergencia del modelo a un punto óptimo.

El algoritmo es descrito a continuación [12].

Requiere: Taza de aprendizaje ϵ (Valor sugerido por defecto 0.001)

Requiere: Tazas de decaimiento para el decaimiento de la estimación de los momentos ρ_1 y ρ_2 en [0,1). (Valores sugeridos por defecto 0.9 y 0.999 respectivamente)

Requiere: Constante pequeña δ usado para estabilidad numérica (Valor sugerido por defecto 10^{-8})

Requiere: Parámetros iniciales $\boldsymbol{\theta}$, valores iniciales para el primer y segundo momento $\boldsymbol{s} = 0, \boldsymbol{r} = 0$

while no se ha cumplido el criterio de paro **do**

Seleccionar un lote de m ejemplos del conjunto de entrenamiento $\boldsymbol{X}, \boldsymbol{Y}$

Calcular el gradiente: $\boldsymbol{g} \leftarrow \frac{1}{m} \bigtriangledown_{\boldsymbol{\theta}} \sum_i L(p_{model}(x^{(i)}; \theta), y^{(i)})$

Actualizar el valor de la estimación del primer y segundo momento:

$$\boldsymbol{s} \leftarrow \rho_1 \boldsymbol{s} + (1 - \rho_1)\boldsymbol{g} \qquad \boldsymbol{r} \leftarrow \rho_2 \boldsymbol{r} + (1 - \rho_2)\boldsymbol{g} \odot \boldsymbol{g}$$

Corregir el sesgo en el primer y segundo momento:

$$\widehat{\boldsymbol{s}} \leftarrow \frac{\boldsymbol{s}}{1-\rho_1^t} \qquad \widehat{\boldsymbol{r}} \leftarrow \frac{\boldsymbol{r}}{1-\rho_2^t}$$

Calcular el cambio en los parámetros $\Delta\boldsymbol{\theta} = -\epsilon \frac{\widehat{\boldsymbol{s}}}{\sqrt{\widehat{\boldsymbol{r}}}+\delta}$

Modificar los parametros: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$

Capacidad y Regularización

Lo más deseable en el desarrollo de este tipo de algoritmos es que tanto el error de entrenamiento, como el de generalización disminuyan durante el proceso de entrenamiento, sin embargo, muchas veces puede darse el caso de que el error de entrenamiento disminuye, mientras que el de generalización permanece constante o inclusive aumenta, esto se conoce como **sobreajuste**, ilustrado en las figuras 3.7 y 3.8, y esta relacionado con la **capacidad** que tiene el modelo para ajustar una gran cantidad de distribuciones.

Un modelo con una alta capacidad (generalmente aquellos que cuentan con una gran cantidad de parametros) es más susceptible a llevar a cabo un sobreajuste de los datos, por lo que muchas veces se recurre a lo que se conoce como técnicas de **regularización**, las cuales pueden llegar a limitar el sobreajuste de los datos, disminuyendo así el error de generalización, con la desventaja de que esto se logra disminuyendo también la capacidad del modelo.

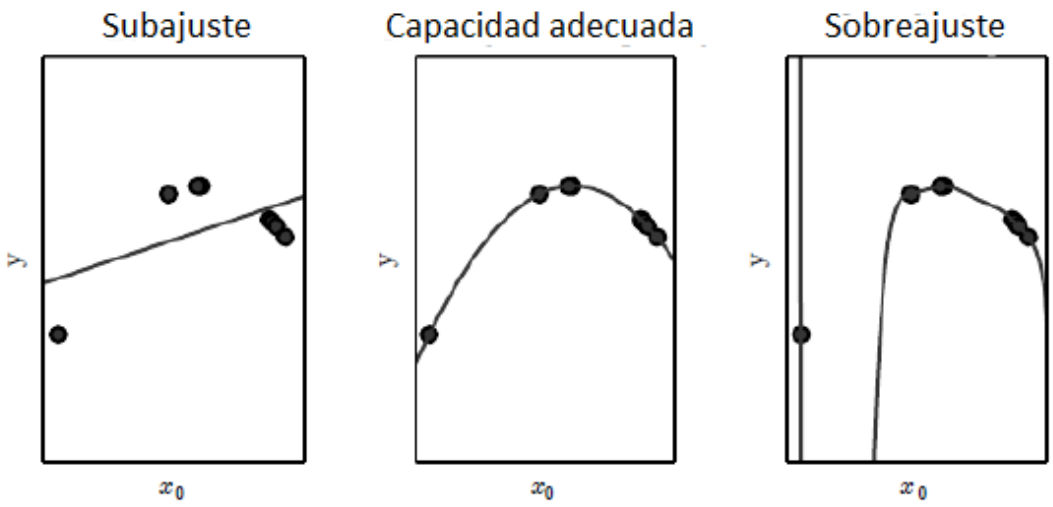


Figura 3.7: Ajuste de a la distribución de los datos dependiente de la capacidad del modelo.

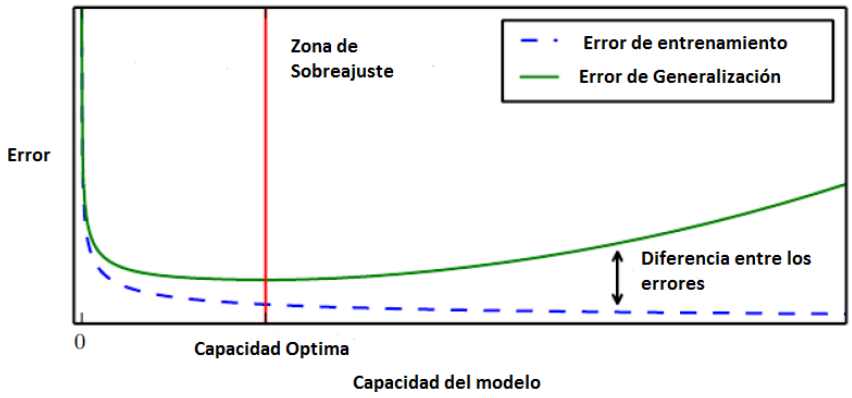


Figura 3.8: Se representa el ajuste de los modelos a los datos.

En este caso, la tecnica de regularización empleada fue **Spatial DropOout** el cual consiste en desactivar canales completos de la imagen en forma aleatoria. En este caso se tomo un porcentaje de DropOut de 0.2 en cada capa.

3.6. Función de Activación

Existen diferentes tipos de funciones de activación empleadas en el desarrollo de este tipo de algoritmos, algunas de estas se ilustran el la figura 3.9.

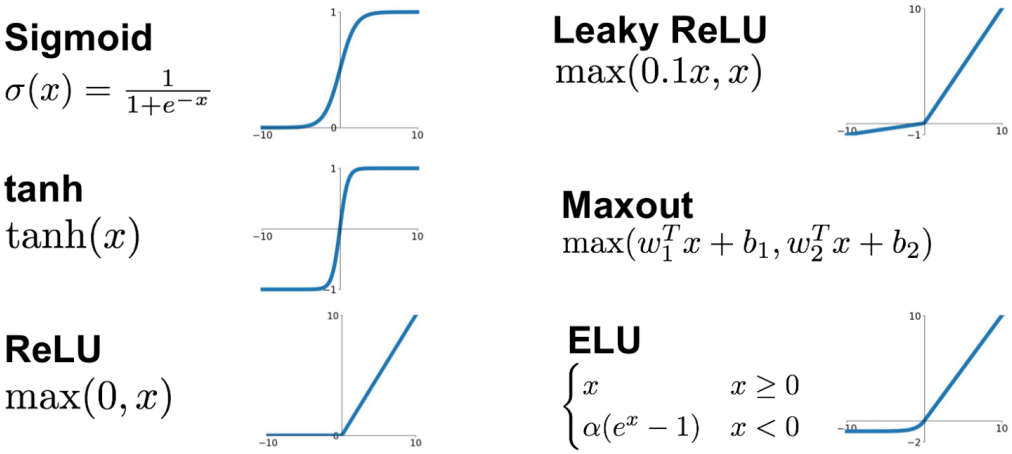


Figura 3.9: Algunas de las funciones de activación comúnmente empleadas en la Redes Neuronales Artificiales.

En este caso como función de activación se empleo la función rectificadora **ReLu**, que ha mostrado un buen desempeño en distintos trabajos [25, 24, 40, 13].

3.7. Batch Normalization

Otra cosa a tomar en cuenta es el desplazamiento interno de la covariable (**internal covariate shift**), en el cual, dado un modelo, la distribución de datos generada en las primeras etapas determina la información que será recibida por etapas posteriores, durante el proceso de entrenamiento, todos los parámetros del modelo están siendo modificados al mismo tiempo, por lo que la distribución de datos generada por una capa en particular en la iteración $i - sima$ no necesariamente será la misma que en la iteración $(i + 1)$, por lo que los parámetros de etapas posteriores estarían siendo modificados para distribuciones diferentes en cada iteración , lo que ralentiza y desestabiliza el proceso de aprendizaje, para resolver esto se recurre a técnicas como 'Batch Normalization'[18] , que permite mejorar el desempeño, rapidez de entrenamiento y estabilidad de las Redes Neuronales al disminuir esta variación.

Suponiendo que se hace pasar por el modelo un lote de m datos $B = \{x_1, ..., x_m\}$, el proceso de normalización consiste en: Calculo del valor promedio del lote

$$\mu_B = \frac{1}{m} \sum_{i=1}^{m} x_i \tag{3.17}$$

Calculo de la varianza de los datos del lote

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_B)^2 \tag{3.18}$$

Normalización de los datos

$$\widehat{x}^k = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{\sigma^{(k)2} + \epsilon}} \tag{3.19}$$

Una vez que los datos has sido normalizados, hay una ultima etapa de re-escalamiento, esto es llevado a cabo mediante el uso de los parámetros γ y β, los cuales pueden ser obtenidos durante el proceso de entrenamiento, esto ultimo se lleva a cabo con la finalidad de que en modelo no pierda la capacidad de modificar la representación de los datos.

$$y_i = \gamma\widehat{x_i} + \beta \equiv \boldsymbol{BN}_{\gamma,\beta}(x_i) \tag{3.20}$$

3.8. Inicialización de parámetros

Antes de comenzar el entrenamiento de uno de estos modelos es necesario contar con valores iniciales para los parámetros de la red, una practica común es que estos sean tomados de forma aleatoria de una distribución gaussiana con media de cero y un valor constante para la desviación estándar, sin embargo se han diseñado distintos métodos de inicialización con la finalidad de felicitar la convergencia del modelo [32, 11, 22]. Un caso particular de esto es [13], trabajo en el que se propone un método de inicialización que toma en cuenta los efectos de las no linealidades generadas por la función de activación RELU, y que alcanzo un buen desempeño en la tarea de clasificación en el conjunto de datos de ImageNet en 2015 . La derivación de este método de inicialización toma en cuanta el numero de valores de entrada que hay en cada capa, así como el efecto que tiene la profundidad del modelo en la varianza de los resultados de las funciones de activación.
El método de inicialización empleado en este trabajo fue *he uniform* de la biblioteca de Keras, este método esta basado en [13] y consiste en tomar valores de forma aleatoria en el intervalo $[-\phi, \phi]$, con:

$$\phi = \sqrt{\frac{6}{N_{in}}} \tag{3.21}$$

donde N_{in} el numero de entradas que hay en la capa donde se llevará a cabo la inicialización de parámetros.

Capítulo 4

Resultados

Como resultado fue posible obtener dos modelos capaces de llevar a cabo la tarea de segmentación de las regiones anatómicas de interés, las características de cada uno de estos modelos son descritas en la tabla 1, donde el modelo *U-net 1* corresponde a aquel en el que se empleó la operación de convolución convencional, mientras que el modelo *U-net 2* corresponde a aquel en el que se empleó la operación de convolución separable.

Características de los modelos

Modelo	Parámetros	Tamaño de lote	Imagen de Entrada	Imagen de Salida
U-net 1	31,125,987	4	(256,256,1)	(256,256,3)
U-net 2	6,030,156	4	(256,256,1)	(256,256,3)

Tabla 4.1: Información relativa a los modelos empleados.

Es posible observar que como uno de los resultados principales, fue posible disminuir el número de parámetros empleados en una razón de ~ 5 al emplear la operación de convolución separable, mostrando que el uso de esta operación permite disminuir significativamente el número de parámetros empleados por la misma arquitectura.

El tamaño de lote de entrenamiento fue el mismo para ambos modelos, esto se llevo a cabo con la finalidad de eliminar la influencia que el tamaño de lote pudiera tener en el proceso de entrenamiento de los modelos. El entrenamiento se llevó a cabo por 50 épocas.

Para la evaluación, el índice de Dice fue calculado para cada una de las clases en cada una de las 212 imágenes del conjunto de evaluación. En la siguiente tabla se muestra el valor promedio, así como la desviación estándar, correspondientes a cada una de las clases.

Índice de Dice

Modelo	Fondo	Miocardio Ventricular	Volumen sanguíneo	Promedio
U-net 1	0.990	0.748	0.747	0.829
U-net 2	0.992	0.793	0.751	0.845

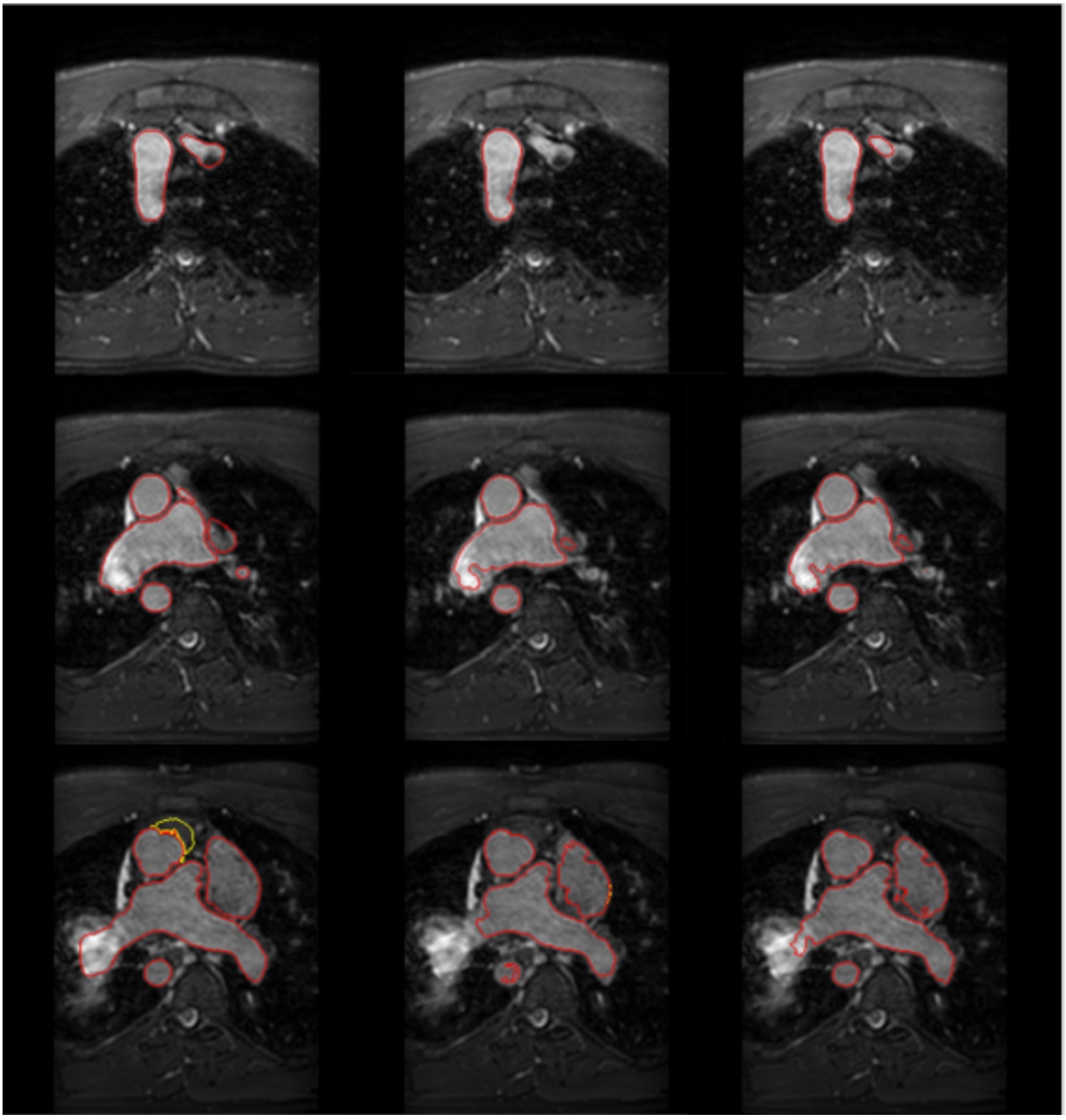
Tabla 4.2: Desempeño alcanzado en las distintas clases.

Tomando como referencia los resultados de las tablas 4.1 y 4.2, se observa que en el caso de esta arquitectura tipo auto-encoder, es posible sustituir el tipo de operación de convolución

empleada, resultando en un modelo permite alcanzar un desempeño similar o incluso superior al llevar a cabo la tarea de segmentación y que a su vez permite llevar esto a cabo con un menor número de parámetros, lo que afecta directamente el tiempo de entrenamiento, de inferencia y el consumo de memoria.

4.0.1. Comparaciones cualitativas

Con la finalidad de contar con un análisis más cualitativo del desempeño de estos modelos, se emplearon ambos modelos para generar las mascaras segmentaciones de las imágenes del conjunto de evaluación, haciendo uso las mascaras de segmentación es posible delimitar el borde de las estructuras anatómicas de interés, como en la figura 4.1, donde se muestran algunas comparaciones de los resultados de segmentación de cada uno de los modelos, así como del resultado esperado. El contorno rojo rodea las regiones de la imagen que corresponden a la clase de volumen sanguíneo, mientras que el contorno amarillo rodea las regiones de la imagen que corresponden a la clase asociada al miocardio.



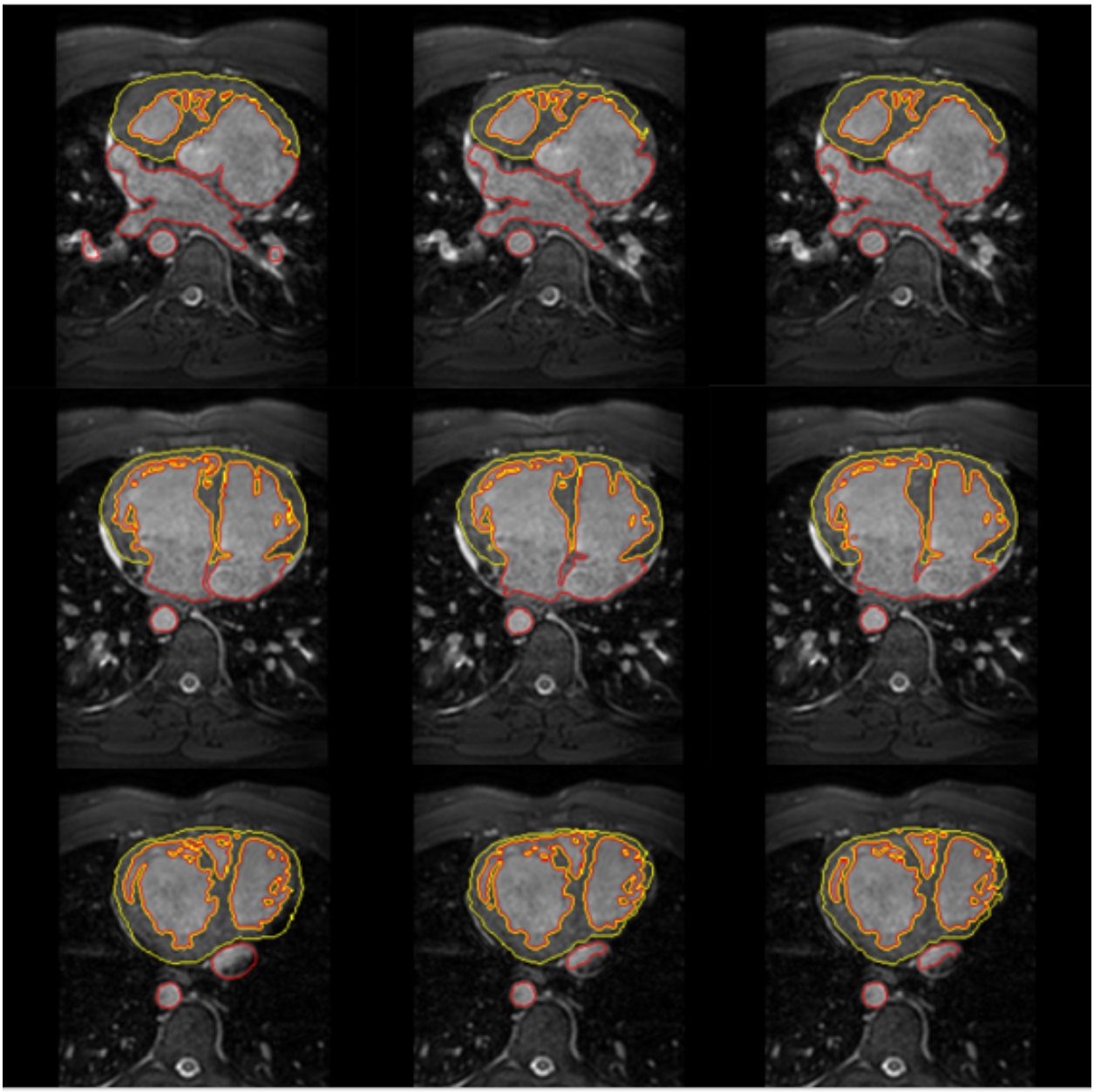


Figura 4.1: Resultado de segmentación. **Columna izquierda)** Mascara de segmentación original. **Columna Central)** Mascara de segmentación U-net 1, **Columna Derecha)** Mascara de segmentación U-net 2

Capítulo 5

Conclusión

En este proyecto se llevo a cabo una comparación de dos modelos diferentes basados en la misma arquitectura tipo auto-encoder, para llevar a cabo la tarea de segmentación de imágenes médicas, la arquitectura original fue propuesta en el trabajo llevado a cabo por *Ronneberger et al.* en 2015 [31], por otro lado a la nueva variante de esta arquitectura, diseñada a lo largo de este proyecto, se le realizaron distintas modificaciones tomando en cuenta los trabajos llevados a cabo en [6, 7, 33, 37, 14], estas modificaciones consisten en la sustitución de muchas de las operaciones de convolución convencional por una variante de esta operación, denominada convolución separable, así como la incorporación de etapas para controlar los tipos de distribuciones que maneja el modelo, como Batch Normalization [18], y métodos de regularización de este, como 2D Spatial Dropout [37].

Fue posible observar que la sustitución de la operación de convolución convencional por convolución separable permite obtener un modelo que alcanza un desempeño comparable e incluso superior en esta tarea, pero disminuyendo significativamente (en una razón de ~ 5) la cantidad de parámetros empleados por la red, lo que puede atribuirse a un uso más óptimo estos. A pesar de que los tiempos del proceso de entrenamiento, así como de inferencia no fueron registrados en este trabajo pues muchas veces esto puede depender en gran medida de la conexión de internet, sin embargo los periodos de tiempo empleados por la nueva red fueron consistentemente menores que aquellos correspondientes a la arquitectura original.

5.0.1. Discusión

Existe una gran cantidad de combinaciones y modificaciones que se pueden aplicar en modelos como estos para poder alcanzar un rendimiento óptimo, los principales cambios en el diseño de este tipo de arquitecturas pueden ser clasificados en tres categorías diferentes:

- Características de las capas de modelo: Tamaño del kernel de convolución, tipo de convolución (convencional, separable, dilatada, etc...), funciones de activación, tipo de normalización y estrategias para disminuir o aumentar el tamaño de la imagen (max pooling, striding, de-convolucion, etc...).
- El proceso de optimización: algoritmo de gradiente de descenso, inicialización de los parámetros, función de costo, tamaño de lote, métodos de regularización, restricciones impuestas y criterio de convergencia.
- Manejo de los datos: Preprocesamiento de las imágenes (Remoción artefactos presentes en la imagen, reescalamiento, aplicación de transformadas de Gabor, Laplace, Hermi-

te, etc...), aumento y selección de datos (Transformaciones afines, agregado de ruido, recorte de imágenes, subsampleo, etc...).

Estas modificaciones pueden ser aplicadas en esté y otros modelos. Muchas de las modificaciones que se realizaron en este modelo fueron seleccionadas por separado de distintos artículos científicos donde se ha mostrado su eficiencia por separado, permitiendo incluso que algunos modelos lograran el mejor desempeño en competencias como ImageNet.

5.0.2. Trabajo a Futuro

A pesar de que fue posible llevar a cabo modificaciones que permitieran obtener una red más eficiente con respecto al número de parámetros empleados y el número de operaciones realizadas, existen muchas otras técnicas que se han empleado, que permiten mejorar el desempeño de la red, algunos de estos ejemplos son:

- La optimización de hiperparametros del modelo como el valor de la taza de aprendizaje inicial, el tamaño de lote de entrenamiento, el porcentaje de regularización, etc. Este proceso de optimización puede ser llevada a cabo mediante el uso de métodos de optimización bayesiana [5].
- El uso de variables latentes [21] para tener una mejor representación de las características de los datos que serán evaluados por el modelo.
- La determinación automática de propiedades de la red, como la profundidad (número de capas), factores de escalamiento de la imagen, etc [36].
- Uso de distintas funciones de error durante el proceso de entrenamiento, un ejemplo de esto es la entropía categórica cruzada con un factor de ponderación, el cual puede forzar al modelo a dar una mayor importancia a alguna categoría en particular, esto también puede ser empleado para corregir el posible desbalance de clases presente en los datos de entrenamiento [39].

En un futuro, seria ideal el poder usar estas técnicas en combinación, con el propósito de que el desempeño sea lo suficientemente bueno como para que el algoritmo sea incorporado en le área clínica.

Bibliografía

[1]

[2] Rafeef Abu-gharbieh, Ghassan Hamarneh, and Tomas Gustavsson. Review - active shape models - part ii: Image search and classification. In *In Proc. Swedish Symposium on Image Analysis*, pages 129–132, 1998.

[3] Sofia Antunes, Antonio Esposito, Anna Palmisano, Caterina Colantoni, Sergio Cerutti, and Giovanna Rizzo. Cardiac multi-detector CT segmentation based on multiscale directional edge detector and 3D level set. *Annals of Biomedical Engineering*, 44(5):1487–1501, 2016.

[4] Gu Y Kim J Wang H Liu Y Goldgof DB Hall LO Korn R Zhao B Schwartz LH Basu S Eschrich S Gatenby RA Gillies RJ Balagurunathan Y, Kumar V. Test-retest reproducibility analysis of lung ct image features. *Digit Imaging.*, 27(6):805–23., 2014.

[5] J. Bergstra, D. Yamins, and D. D. Cox. Making a science of model search, 2012.

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[7] François Chollet. Xception: Deep learning with depthwise separable convolutions, 2016.

[8] Torsten N. Wiesel David H. Hubel. *Brain and visual perception: the story of a 25-year collaboration.* Oxford University Press US., 1st edition, 2005.

[9] Instituto Nacional de Estadística y Geografía INEGI. Características de las defunciones registradas en méxico durante 2017, 2018. Accessed: 2019-09-02.

[10] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics.*, 36(4):193–202., 2013.

[11] Xavier Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research - Proceedings Track*, 9:249–256, 01 2010.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016. http://www.deeplearningbook.org.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

[14] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

[15] David H. Hubel and Torsten N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 184(3):574–91, 1959.

[16] David H. Hubel and Torsten N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, 195(1):215–243, 1968.

[17] Nabil Ibtehaz and M. Sohel Rahman. Multiresunet: Rethinking the u-net architecture for multimodal biomedical image segmentation. *Neural Networks*, 121:74–87, Feb 2019.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[19] Antonio Garcia-Uceda Juarez, Raghavendra Selvan, Zaigham Saghir, and Marleen de Bruijne. A joint 3d unet-graph neural network-based method for airway segmentation from chest cts, 2019.

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[21] Simon A. A. Kohl, Bernardino Romera-Paredes, Clemens Meyer, Jeffrey De Fauw, Joseph R. Ledsam, Klaus H. Maier-Hein, S. M. Ali Eslami, Danilo Jimenez Rezende, and Olaf Ronneberger. A probabilistic u-net for segmentation of ambiguous images, 2018.

[22] Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop*, pages 9–50, London, UK, UK, 1998. Springer-Verlag.

[23] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search, 2018.

[24] Andrew L. Maas. Rectifier nonlinearities improve neural network acoustic models. 2013.

[25] Vinod Nair and Geoffrey Hinton. Rectified linear units improve restricted boltzmann machines vinod nair. volume 27, pages 807–814, 06 2010.

[26] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention u-net: Learning where to look for the pancreas, 2018.

[27] Genevieve B. Orr and Klaus-Robert Müller, editors. *Neural Networks: Tricks of the Trade*, London, UK, UK, 1998. Springer-Verlag.

[28] Geva T. Powell A.J. Moghari M.H. Golland P. Pace D.F., Dalca A.V. Interactive whole-heart segmentation in congenital heart disease. In *Lecture Notes in Computer Science*, volume 9351, pages 80–88. Springer, Cham, 2015.

[29] Nikos Paragios, Marie-Pierre Jolly, Maxime Taron, and Rama Ramaraj. Active shape models and segmentation of the left ventricle in echocardiography. volume 3459, pages 131–142, 04 2005.

[30] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

[31] Brox T. Ronneberger O., Fischer P. U-net: Convolutional networks for biomedical image segmentation. IN: MICCAI 2015.

[32] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, 2013.

[33] Laurent Sifre. *Rigid-Motion Scattering For Image Classification*. PhD thesis, Ecole Polytechnique, CMAP, 2014.

[34] Shafiullah Soomro, Farhan Akram, Asad Munir, Chang Lee, and Kwang Choi. Segmentation of left and right ventricles in cardiac mri using active contours. *Computational and Mathematical Methods in Medicine*, 2017:1–16, 08 2017.

[35] Sara Raimondi Daniela Origgi Cristiana Fanciullo Alessio Giuseppe Morganti Massimo Bellomi Stefania Rizzo, Francesca Botta. Radiomics: the facts and the challenges of image analysis. *European Radiology Experimental*, 2(36), 2018.

[36] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2019.

[37] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christopher Bregler. Efficient object localization using convolutional networks, 2014.

[38] Jelmer M. Wolterink, Tim Leiner, Max A. Viergever, and Ivana Išgum. Dilated convolutional neural networks for cardiovascular mr segmentation in congenital heart disease. *Lecture Notes in Computer Science*, page 95–102, 2017.

[39] Xin Yang, Cheng Bian, † Lequan, Dong Ni, and Pheng-Ann Heng. Hybrid loss guided convolutional networks for whole heart parsing. 09 2017.

[40] M.D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q.V. Le, Phuongtrang Nguyen, Andrew Senior, V. Vanhoucke, J. Dean, and G.E. Hinton. On rectified linear units for speech processing. pages 3517–3521, 05 2013.

[41] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. *CoRR*, abs/1807.10165, 2018.

[42] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation, 2016.