



UNIVERSIDAD VERACRUZANA

INSTITUTO DE INVESTIGACIONES EN INTELIGENCIA ARTIFICIAL

SEGMENTACIÓN DE PLACENTAS EN IMÁGENES DE
ULTRASONIDO 2D EVOLUCIONANDO REDES NEURONALES
CONVOLUCIONALES

T E S I S

PARA OBTENER EL TÍTULO DE:

MAESTRO EN INTELIGENCIA ARTIFICIAL

PRESENTA:

JOSÉ ANTONIO FUENTES TOMÁS

DIRECTOR:

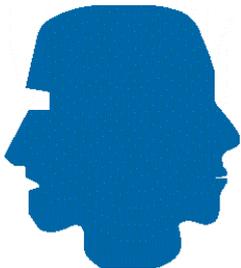
DR. HÉCTOR GABRIEL ACOSTA MESA

CODIRECTORES:

DR. EFRÉN MEZURA MONTES

DR. RODOLFO HERNÁNDEZ JIMÉNEZ

XALAPA, VERACRUZ, MÉXICO 2022



Universidad Veracruzana
Instituto de Investigaciones en Inteligencia Artificial
Calle Paseo No 112, Colonia Nueva Xalapa, C.P. 91097
(2022)
Xalapa, Veracruz, México

TÍTULO: Segmentación de placentas en imágenes de ultrasonido 2D evolucionando redes neuronales convolucionales
AUTOR: José Antonio Fuentes Tomás
DIRECTOR: Dr. Héctor Gabriel Acosta Mesa
CODIRECTORES: Dr. Efrén Mezura Montes Dr. Rodolfo Hernández Jiménez
NÚMERO DE PÁGINAS: x, 72

Resumen

En este trabajo de investigación se propone un algoritmo para el diseño automático de redes neuronales convolucionales que permitan segmentar el órgano de la placenta en imágenes de ultrasonido. El monitoreo de este órgano en la etapa de gestación es de vital importancia tanto para el feto como para la madre, favoreciendo la prevención de anomalías y la toma de decisiones por parte del médico. Características de la placenta como el tamaño, forma y ubicación permiten conocer el estado de este órgano y pueden ser monitoreadas mediante imágenes bidimensionales proporcionadas por un estudio de ultrasonido, sin embargo, este tipo de imágenes se caracterizan su alto nivel de ruido. Con el fin de conocer las características importantes de la placenta se requiere de un sistema de *segmentación* que identifique aquellos píxeles que corresponden al órgano de la placenta en una imagen digital. En la literatura se ha popularizado el uso de la arquitectura U-Net para tareas de segmentación de imágenes. Sin embargo, este tipo de red es de uso general, cuyo diseño requiere gran expertiz y que pudiera no aplicarse en distintos dominios. Es por ello que se recurre al paradigma de cómputo evolutivo de programación genética con el que se diseña la topología de una red neuronal convolucional tipo U-Net que permita obtener un desempeño de segmentación similar al de U-Net con una complejidad menor en términos del índice de Dice y el número de parámetros. Los resultados obtenidos muestran que el algoritmo propuesto permite obtener una reducción destacable en el número de parámetros de U-Net que oscila entre el 80 y 98 %. Además, el desempeño de segmentación alcanza un nivel competitivo en comparación con U-Net con una diferencia de 0.012 unidades en el índice de Dice al utilizar el conjunto de datos de placentas y de 0.043 unidades al considerar un segundo conjunto de datos (*Retinal Vessel Segmentation*).

Agradecimientos

En primer lugar quiero agradecer a mi director, codirectores: Dr. Héctor Gabriel Acosta Mesa, Dr. Efrén Mezura Montes y Dr. Rodolfo Hernández Jiménez, por su tiempo, consejos y guía a lo largo de este trabajo.

A mi familia, Yolanda, Antonio y Raziel, por su apoyo incondicional, comprensión y amor siempre presente en toda mi vida. s A mi querida Karlita, por acompañarme antes y durante el proceso de estancia en el posgrado, por su apoyo constante y por motivarme a mejorar personal y profesionalmente.

Los autores agradecen al Laboratorio Nacional de Supercómputo del Sureste de México perteneciente a la red de laboratorios nacionales CONACYT, por los recursos computacionales, el apoyo y la asistencia técnica.

Índice general

Resumen	III
Agradecimientos	IV
1. Introducción	1
1.1. Definición del problema	3
1.2. Justificación	4
1.3. Hipótesis	5
1.4. Objetivos	5
1.4.1. Objetivo General	5
1.4.2. Objetivos Específicos.	5
2. Marco Teórico	6
2.1. Segmentación de imágenes	6
2.1.1. Métricas de segmentación	6
2.2. Redes Neuronales Convolucionales	9
2.2.1. Capas de convolución	9
2.2.2. Convolución transpuesta	12
2.2.3. Convolución separable en profundidad	13
2.2.4. Capas de pooling	15
2.2.5. Capas de normalización	15
2.2.6. Arquitectura Res-Net	17
2.2.7. Arquitectura Dense-Net	18
2.2.8. Contracción y Excitación	19
2.3. Algoritmos Evolutivos	20
2.3.1. Neuroevolución	21
2.3.2. Programación Genética	22
3. Marco referencial	26
3.1. Segmentación de placentas	26
3.1.1. Gordon N. Stevenson, et al. (2015) [64]	26
3.1.2. Pádraig Looney, et al (2017) [47]	27
3.1.3. Ricky Hu, et al. (2019) [34-36]	27
3.1.4. Han, et al (2019) [24]	28
3.1.5. Zimmer, et al (2020) [85]	28
3.1.6. Comentarios	28

Índice general

3.2. Neuroevolución basada en PG	28
3.2.1. Suganuma, et al (2017) [66, 67]	28
3.2.2. Zhu, et al (2018) [84]	29
3.2.3. Harris, et al (2018) [41]	29
3.2.4. McGhie et al (2020) [48]	29
3.2.5. Comentarios	29
3.3. Segmentación de imágenes médicas utilizando neuroevolución	30
3.3.1. Mortazi y Bagei (2018) [53]	30
3.3.2. Weng, et al (2019) [81]	30
3.3.3. Baldeon Calisto y Lai Yuen (2020,2021 [2, 8])	31
3.3.4. Hassanzadeh, et al (2020, 2021) [25, 26]	31
3.3.5. Comentarios	31
4. Propuesta	33
4.1. Algoritmo	33
4.1.1. Estructura de operaciones	36
4.1.2. Conjunto de funciones	36
4.1.3. Conjunto de terminales	41
4.1.4. Parámetros y características de NASGP-Net	41
5. Experimentos y resultados	45
5.1. Primera Fase	46
5.2. Segunda fase	48
5.2.1. Segmentación de placentas	48
5.2.2. Segmentación de vasos en la retina	52
5.3. Comentarios	57
6. Conclusiones y trabajo futuro	60
A. Publicaciones del Proyecto	62
B. Carta de consentimiento informado	64
Bibliografía	67

Índice de figuras

1.1. Imagen de ultrasonido adquirida en clínica privada	3
1.2. Segmentación de la placenta en imágenes de ultrasonido	4
2.1. Interpretación de índice de Jaccard con conjuntos	7
2.2. Interpretación de índice de Dice con conjuntos	8
2.3. Interpretación gráfica de la distancia <i>Hausdorff</i>	9
2.4. Operación de convolución entre una imagen en escala de grises de 4×4 con un kernel de 3×3 . El resultado es asignado al píxel central marcado con rojo. w_i y b se denominan los pesos del filtro y sesgo, respectivamente	10
2.5. Capa de convolución aplicada a una imagen de 17×8 píxeles. Notar que las dimensiones de la imagen se reducen a la salida de la operación de convolución con dimensiones de 15×8	11
2.6. Convolución dilatada aplicada a una imagen de 17×8 píxeles con un filtro de 3×3 y una tasa de dilatación equivalente a 2. Notar las diferencias en los tamaños de salida al compararla con la figura 2.5b	12
2.7. Ejemplo de convolución transpuesta. Notar que las dimensiones de la imagen de salida son mayores a la de la imagen de entrada	12
2.8. Ejemplo de convolución regular para una imagen de entrada de 3 canales (RGB) con un filtro de 3×3	13
2.9. Ejemplo de convolución separable en profundidad para una imagen de entrada de 3 canales (RGB) con un filtro de 3×3	14
2.10. Ejemplo de una operación max-pooling	15
2.11. Tipos de normalización de acuerdo con el conjunto \mathcal{S}_i seleccionado	17
2.12. Bloque residual utilizado por Res-Net. Las líneas punteadas representan una capa de convolución con un tamaño de kernel de 1	18
2.13. Bloque denso utilizado por Dense-Net. Los mapas de características de entrada correspondientes a las capas anteriores se distinguen mediante los colores correspondientes. A la salida se aplica una convolución con un tamaño de kernel de 1×1 que reduce los canales de salida.	19
2.14. Bloque SE, por sus siglas para <i>Squeeze-and-Excitation</i> . La figura del lado derecho muestra a detalle la etapa de excitación	20
2.15. Ejemplo de programa representado como un árbol. La función considerada determina el máximo entre dos números, dado por $\max(x + x, x + 3 * y)$. En este ejemplo el conjunto de funciones es $\{\max, +, *\}$, mientras que el conjunto de terminales lo constituyen $\{X, Y, 3\}$	22

2.16. Ejemplo de cruza de un punto en PG. Notar que este tipo de cruza genera 2 hijos, aunque pudiera implementarse una versión que genere únicamente 1 hijo descartando alguno de ellos	24
2.17. Ejemplo de mutación de subárbol en PG	25
3.1. Arquitectura <i>U-Net</i>	27
4.1. Células de U-Net representadas como árboles. Se asume que el tamaño de la máscara aplicado a la convolución es cuadrada. (A)-(D) corresponden a las células de la etapa de contracción. (E) representa el cuello de botella, punto en el cual comienza la etapa de expansión. (F)-(I) corresponden a las células de la etapa de expansión	34
4.2. NASGP-Net: reglas de construcción y modularización. Los operadores de variación de PG afectan al árbol que representa la primera célula en la sección de contracción. La red convolucional se construye a partir de la primera célula siguiendo las reglas de la simbología, marcadas con flechas punteadas. El flujo de los tensores o mapas de características sucede sobre las flechas continuas. Las conexiones atajo o <i>skip connections</i> se efectúan mediante una operación de concatenación	35
4.3. Estructura de operaciones en NASGP-Net	37
4.4. Operaciones de extracción de características. Las operaciones añaden capas de convolución al contenedor original que a su vez generan capas de normalización y activación. Las capas añadidas se enmarcan con un rectángulo rojo. En el ejemplo se asume que la lista de construcción (terminal <i>módulo</i>) contiene una capa de convolución, con su respectiva capa de normalización y activación, y muestra las modificaciones que sufre al aplicar la función <i>conv</i> o <i>sconv</i>	39
4.5. Operaciones de conectividad. Las operaciones crean conexiones sobre la secuencia de bloques de entrada, marcadas con flechas rojas. En el ejemplo se asume que la lista de construcción (terminal <i>módulo</i>) se compone de tres elementos definidos como una secuencia convolución-convolución separable en profundidad-convolución. La operación <i>rCon</i> crea una conexión cada dos capas de convolución. Nótese que la conexión se realiza desde la entrada a la capa de convolución <i>depth-wise</i> a la salida de la secuencia. Además, crea una capa de convolución <i>point-wise</i> que ajusta el número de canales de entrada para que la función de adición sea aplicada correctamente. La función <i>dCon</i> añade conexiones densas a la secuencia y crea una capa de convolución <i>point-wise</i> que comprime el número de canales con base en un factor θ	40
4.6. Operación de recalibración, <i>se</i> . Esta operación puede ser aplicada a una secuencia de capas de convolución, un bloque residual o un bloque denso. En el ejemplo la operación <i>se</i> es aplicada al bloque residual obtenido en la figura 4.5. El factor de reducción, r , toma un valor de 8, dado que es el mínimo número de canales de salida posible para la primera célula. . .	41

Índice de figuras

5.1. Fase 1 de experimentación. Gráficas de convergencia, tamaño y profundidad promedio	49
5.2. Gráfica de convergencia, profundidad y tamaño promedio de la ejecución correspondiente a la mediana de acuerdo con la tabla 5.5	52
5.3. Árbol que representa la primera célula de la red convolucional correspondiente a la mediana de acuerdo con la tabla 5.5	52
5.4. Mejores 3 imágenes segmentadas por NASGP-Net	53
5.5. Mejores 3 imágenes segmentadas por U-Net	53
5.6. Peores 3 imágenes segmentadas por NASGP-Net	54
5.7. Peores 3 imágenes segmentadas por U-Net	54
5.8. Gráfica de convergencia, profundidad y tamaño promedio de la ejecución correspondiente a la mediana de acuerdo con la tabla 5.8	57
5.9. Árbol que representa la primera célula de la red convolucional neuroevolucionada para <i>Retinal Vessel Segmentation</i>	58
5.10. <i>Retinal Vessel Segmentation</i> : Mejores resultados de segmentación con <i>NASGP-Net</i>	58
5.11. <i>Retinal Vessel Segmentation</i> : Peores resultados de segmentación con <i>NASGP-Net</i>	58

Índice de tablas

2.1. Matriz de confusión para clasificación binaria	7
3.1. Hiperparámetros, función de costo y tipos de imágenes considerados en diferentes algoritmos basados en Neuroevolución para segmentación de imágenes	30
4.1. Conjunto de funciones	38
4.2. Conjunto de terminales	42
4.3. Características de NASGP-Net	43
5.1. Parámetros para los experimentos considerados en la fase 1 de experimentación	47
5.2. Resultados obtenidos en la fase 1 de experimentación. Se muestran en negritas los mejores resultados con base en las métricas de índice de Dice (Dice), Intersección sobre Unión (IoU) y <i>Hausdorff Distance (HD)</i> , ver sección 2.1.1	48
5.3. Parámetros de NASGP-Net para segmentación de placentas	48
5.4. 10 Ejecuciones de U-Net <i>vs</i> NASGP-Net, para la segmentación de placentas. Se marcan en negritas los mejores resultados por cada método del conjunto de ejecuciones	51
5.5. Estadísticas obtenidas con base en las 10 ejecuciones. Se realiza la prueba estadística <i>U de Mann Whitney</i> tomando en cuenta las métricas de Dice, IoU y Hausdorff Distance. Se marcan en negritas los mejores resultados entre ambos métodos.	51
5.6. Parámetros de NASGP-Net utilizando <i>Retinal Vessel Segmentation</i>	55
5.7. 5 Ejecuciones de U-Net <i>vs</i> NASGP-Net utilizando <i>Retinal Vessel Segmentation</i> . Se marcan en negritas los mejores resultados por cada método del conjunto de ejecuciones	56
5.8. Estadísticas obtenidas con base en las 5 ejecuciones. Se realiza la prueba estadística <i>U de Mann Whitney</i> tomando en cuenta las métricas de Dice, IoU y Hausdorff Distance. Se marcan en negritas los mejores resultados entre ambos métodos.	56

Capítulo 1

Introducción

El embarazo es una de las etapas más importantes en la vida de la mujer que requiere de muchos cuidados. Un embarazo normal toma alrededor de 40 semanas desde el primer día después del último periodo menstrual [65]. Enfermedades previas como sobrepeso, hipertensión, diabetes o tabaquismo representan un riesgo perinatal tanto para el feto como para la madre [49]. Solo a través de la placenta es posible comprender el desarrollo del ser humano. Desde su etapa de mayor cambio, mayor crecimiento y mayor especialización [17].

El peso del recién nacido es el resultado de efectos interactivos entre diferentes factores como el potencial genético, tiempo de gestación, adecuado ambiente intrauterino, nutrición, salud materna y función útero-placentaria [9].

La placenta es un órgano que se desarrolla casi al mismo tiempo que el feto. Actúa como los pulmones, intestino, riñones e hígado del feto [7]. Existen distintas anomalías relacionadas a la placenta, generalmente éstas pueden ser identificadas o dar algún indicio observando el tamaño, forma, ubicación, inserción del cordón umbilical y grado de madurez placentaria. A continuación, se detalla una breve descripción de estas características:

- *Tamaño*
Una de las reglas generales es que el grosor de la placenta en milímetros corresponde con la edad gestacional en semanas, sumando o restando 10mm. Otra característica está relacionada con el peso al nacer. Tanto una placenta gruesa en los primeros días o una placenta delgada en los últimos pueden estar relacionadas con un embarazo anormal[30].
- *Forma*
Una de las anomalías asociada es la forma de la placenta circunvalada, donde la placa coriónica (lado fetal de la placenta) no es tan grande como la basal (lado materno de la placenta).
- *Ubicación*
Se refiere a la posición del útero. Puede definir una cesárea y complicar el parto si el borde de la placenta está cerrada del cuello uterino alrededor de las 15-20 semanas de gestación [44].

- *Inserción del cordón umbilical*

Una inserción fuera del centro representa un riesgo porque los vasos umbilicales no tienen un soporte total por la placenta [30].

- *Grado de madurez*

Existe una relación entre la madurez de la placenta y su calcificación, la cual a su vez está vinculada a los riesgos en el resultado del embarazo, [49], aunque existen algunas opiniones encontradas sobre esta afirmación. Grannum propuso una clasificación basada en cuatro grados, de 0 a III [23].

Hay una observación especial para las placentas de grado III; algunos estudios relacionan un posible riesgo en el resultado perinatal si la aparición de las calcificaciones placentarias no coinciden con el tiempo de gestación [49]. De acuerdo con este estudio, pueden presentarse enfermedades como preclampsia, bajo peso del feto al nacer, sobrepeso y retraso del crecimiento intrauterino. El depósito de minerales a nivel de la microcirculación y vellosidades placentarias, cuando éstas se presentan en territorios extenso de la placenta, del 25-30 % del territorio placentario existen riesgos de que el producto sufra hipoxia y disminución en el aporte de los nutrientes, lo que traería consigo resultados perinatales adversos.

Una mejora en la detección temprana de estas anomalías podría prevenir una situación de mortinato [59]. De acuerdo a la ONU, cada 16 segundos se produce un mortinato, esto representa alrededor de 1.9 millones de mortinatos en 2019, de los cuales el 84 % ocurren en países de ingresos bajos y medio-bajos con una mayor cantidad de ocurrencias en el continente africano [76]. En un escenario más particular, México registró 20,049 mortinatos en el mismo año [39].

Gracias a descubrimientos de fenómenos físicos han surgido instrumentos médicos con los que es posible entender la estructura, función y patologías del cuerpo humano, tal como los rayos-X, el ultrasonido, la radioactividad y resonancia magnética [3]. A través de las imágenes, como un medio de información, es posible intervenir en procesos de enfermedades y lesiones humanas al ser interpretadas y asimiladas [29].

Uno de los métodos no invasivos de imagenología médica que destaca entre los demás, dado su precio y accesibilidad, es el ultrasonido que permite evaluar al embarazo en su totalidad y en tiempo real. Sin embargo, tiene la desventaja de depender en gran medida de la experiencia del médico y que además contiene artefactos sonográficos (sombra acústica y reverberación) que pudieran confundir al médico. Funciona como una serie de ondas mecánicas originadas por la vibración de un cuerpo elástico (piezoeléctrico) y propagadas por un medio (tejido) con una frecuencia superior a la del sonido audible humano (20kHz). Parte de estas ondas son reflejadas y captadas por un transductor que genera un voltaje y posteriormente son mapeadas a una imagen (ver figura 1.1) de acuerdo a una profundidad e intensidad de la señal [34, 77].

En el presente trabajo se propone segmentar la placenta, es decir, delimitar la zona de píxeles correspondientes a la placenta en una imagen digital, para que eventualmente este ejercicio sirva para extraer información de ella y poder advertir sobre los riesgos al

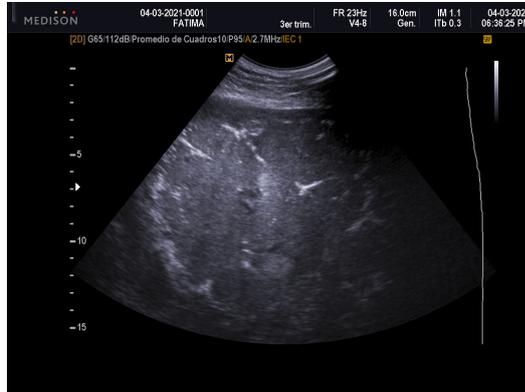


FIGURA 1.1: Imagen de ultrasonido adquirida en clínica privada

identificar sus características, por ejemplo, el tamaño, la forma o madurez, relacionado con el porcentaje de calcificación. Esta última fue una de las motivaciones iniciales del desarrollo de este estudio, el cual busca ser una base sólida para estudios posteriores de esta naturaleza.

1.1. Definición del problema

Un estudio de ultrasonido de la placenta es tan importante como examinar al feto. Una imagen en dos dimensiones provee información suficiente para detectar la ubicación y forma de la placenta [30]. Las técnicas de ultrasonido tienen algunas limitaciones como lo es la resolución de la imagen y artefactos, considerados como factores de confusión [77]. Es por ello que se requiere de un sistema de *segmentación* robusto, es decir, una delimitación de áreas de interés. Dado que la segmentación consiste en una clasificación de píxeles, a menudo es tratado como un problema de reconocimiento de patrones. Algunos enfoques para resolver este problema son la lógica difusa, redes neuronales y modelos deformables [3], particularmente en este estudio nos centraremos en las redes neuronales.

Las redes neuronales artificiales profundas como una técnica de aprendizaje automático se han vuelto indispensables para tareas de visión por computadora, reconocimiento de voz y procesamiento de lenguaje natural. Las redes que se especializan en el procesamiento visual de imágenes se denominan redes convolucionales (CNN), una herramienta bioinspirada en la corteza visual del cerebro. Este tipo de modelos han sido aplicados exitosamente en tareas de segmentación de imágenes médicas. En la Figura 1.2 se ilustra el concepto de segmentación en una imagen de ultrasonido, en la que ha intervenido un experto para delimitar el área de la placenta y se compara con la máscara obtenida por un modelo de aprendizaje profundo. Asumiendo un enfoque de aprendizaje supervisado, la discrepancia entre las imagen obtenida y de referencia define el desempeño de segmentación del modelo.

Una de las arquitecturas de red neuronal más popular utilizada en la segmentación

de imágenes médicas es U-Net [60] y que particularmente ha sido aplicada en la segmentación de la placenta en distintos tipos de imágenes [24, 36, 85]. U-Net es un claro ejemplo de red neuronal elaborada manualmente, donde un experto humano ha intervenido en su diseño y que pudiera no aplicarse en distintos dominios. Para superar dichas limitaciones surge la Neuroevolución (NE), un conjunto de métodos para determinar la topología de redes neuronales y/o su entrenamiento. Para la optimización de los pesos o entrenamiento de la red se ha popularizado emplear una función de costo continua que mide la discrepancia entre los datos reales y predichos por el modelo [46]. Por otro lado, la optimización de la topología es una tarea complicada dependiente del problema y cuyo impacto es mayor al de la optimización aislada de los pesos [74]. Metaheurísticas como los algoritmos evolutivos y de inteligencia colectiva han sido aplicados en el diseño automático de topologías de redes neuronales [78].

La programación genética (PG) [43] es una técnica del cómputo evolutivo para evolucionar programas con base en un conjunto de funciones aplicadas a un conjunto de terminales. Se caracteriza por utilizar una representación de árbol, a diferencia de los algoritmos genéticos (GA) que utilizan vectores de números binarios o reales. La diversidad en el espacio de búsqueda, expresividad y fácil decodificación son las principales ventajas de PG, paradigma de interés para este estudio.

En resumen, el problema consiste en diseñar una topología de red neuronal utilizando PG como mecanismo de búsqueda que sea capaz de segmentar placentas en imágenes de ultrasonido 2D y que toma inspiración en la arquitectura U-Net.



FIGURA 1.2: Segmentación de la placenta en imágenes de ultrasonido

1.2. Justificación

Al utilizar una representación de árbol se tiene mayor expresividad. A mayor expresividad, mayor flexibilidad y menor modularidad [78]. De acuerdo a la literatura, el uso de arquitecturas ‘encoder-decoder’, como lo es U-Net, han demostrado tener un buen desempeño en tareas segmentación de imágenes médicas [2, 53, 81]. Además, el potencial de PG en el campo del aprendizaje profundo no ha sido explorado extensamente [15], por lo que esto nos da un margen de contribución en esta área. Utilizando PG para diseñar una arquitectura de red neuronal, se obtendrá un modelo cuya tarea es exclusiva para la

segmentación de placentas en imágenes bidimensionales de ultrasonido. Además se buscará que esta arquitectura tenga menor cantidad de parámetros con lo que la latencia y tiempo de entrenamiento de la red neuronal disminuirán.

1.3. Hipótesis

Utilizando PG para crear una topología de red neuronal convolucional *ad hoc* para segmentar la región de la placenta en imágenes bidimensionales de ultrasonido, es posible obtener una topología más compacta que la arquitectura U-Net con un desempeño de segmentación similar, en términos del número de parámetros y del índice de Dice, donde los estudios mencionados [34, 36] han llegado a alcanzar un valor de 0.92.

1.4. Objetivos

1.4.1. Objetivo General

Diseñar mediante Neuroevolución basada en Programación Genética una arquitectura de red neuronal para la segmentación de placentas en imágenes bidimensionales de ultrasonido que obtenga un desempeño competitivo en comparación con U-Net en términos del índice de Dice y una complejidad más compacta.

1.4.2. Objetivos Específicos.

- Adquirir imágenes.
- Implementar una interfaz para el etiquetado de imágenes del conjunto de datos por parte de un experto humano.
- Establecer el conjunto de funciones y de terminales para que sean tomados en cuenta en el proceso evolutivo.
- Segmentar la placenta en la imagen de ultrasonido filtrada en el punto anterior utilizando una red convolucional con una topología dada por PG.
- Reportar y comparar resultados con U-Net.

La organización del documento es la siguiente. En el capítulo 2 se presenta el marco teórico donde se exponen conceptos fundamentales sobre segmentación de imágenes, redes neuronales convolucionales y algoritmos evolutivos. En el capítulo 3 se presenta una revisión de la literatura donde brevemente se describen estudios relacionados con la tarea particular de segmentación de la placenta en distintos tipos de imágenes, trabajos y propuestas de NE basada en PG y estudios donde métodos de NE han sido aplicados en tareas de segmentación. En el capítulo 4 se detalla el algoritmo propuesto de NE basado en PG para tareas de segmentación. El capítulo 5 describe los resultados obtenidos y la comparación con U-Net. Finalmente, se presenta una breve discusión y trabajo futuro.

Capítulo 2

Marco Teórico

2.1. Segmentación de imágenes

La *segmentación* consiste en separar las estructuras de interés del fondo y entre sí en una imagen; divide a una imagen en sus partes u objetos que la componen [3, 21]. A diferencia de la clasificación de imágenes donde cada imagen es una observación, en la segmentación de imágenes cada píxel es una observación a clasificar. En imagenología médica, la delimitación de distintos componentes sirve como herramienta para analizar estructuras anatómicas, tipos de tejidos y regiones patológicas [3].

Existen distintos enfoques para segmentar objetos o estructuras de interés en imágenes digitales de distintos dominios, entre ellos destacan los algoritmos basados en *umbralización*, que seleccionan un nivel de intensidad del histograma de la imagen como nivel de umbral. Otro enfoque se basa en la *detección de bordes* para delimitar una región de la imagen, como lo son los modelos deformables. La segmentación también se ha alcanzado mediante el método de *clustering*, basado en lógica difusa y funciones de pertenencia. Finalmente, las *redes neuronales*, enfoque de interés en este trabajo, han demostrado un buen desempeño en esta tarea tanto en la modalidad de aprendizaje supervisado como no supervisado.

2.1.1. Métricas de segmentación

Para responder a la pregunta, “¿qué tan bien se ha realizado la segmentación?” se utilizan métricas que cuantifican el desempeño del modelo al comparar la máscara de referencia con la máscara predicha. Generalmente las métricas se calculan a partir de la matriz de confusión, que despliega la información predicha (filas) y la real (columnas) de un sistema de clasificación. Particularmente para clasificación binaria, la matriz consta de 4 elementos con un tamaño de 2×2 , ver tabla 2.1.

Enseguida se describen los elementos que conforman la matriz de confusión para una clasificación binaria los cuales son primordiales para algunas de las métricas de desempeño que se describen posteriormente.

- *VP*(Verdadero Positivo). Cantidad de predicciones en el que el clasificador predice correctamente la clase positiva como tal.

TABLA 2.1: Matriz de confusión para clasificación binaria

		Valores Reales	
		Positivo	Negativo
Valores Predichos	Positivo	VP	FP
	Negativo	FN	VN

- *VN*(Verdadero Negativo). Cantidad de predicciones en el que el clasificador predice correctamente la clase negativa como tal.
- *FP*(Falso Positivo). Cantidad de predicciones en el que el clasificador predice incorrectamente la clase negativa como positiva.
- *FN*(Falso Negativo). Cantidad de predicciones en el que el clasificador predice incorrectamente la clase positiva como negativa.

Cuatro de las métricas más comunes para cuantificar el desempeño de segmentación son las siguientes:

Precisión de píxeles. Porcentaje de píxeles clasificados correctamente. Es posible calcularlo desde la matriz de confusión mediante la siguiente expresión:

$$\frac{VP + VN}{VP + VN + FP + FN} \quad (2.1)$$

Intersección sobre unión (IoU) o índice de Jaccard. Considerando cada imagen como un conjunto de píxeles, donde uno de ellos corresponde a la máscara segmentada por un experto y otro a la máscara predicha por el modelo, IoU se define como la intersección entre ambos conjuntos dividido entre la unión de los mismos, (ver figura 2.1). Sus valores se encuentran en el en el intervalo $[0, 1]$ donde un valor más alto indica un mejor desempeño. Dada la matriz de confusión, IoU toma la siguiente expresión:

$$\frac{VP}{VP + FP + FN} \quad (2.2)$$

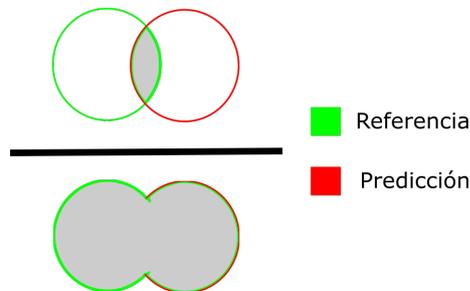


FIGURA 2.1: Interpretación de índice de Jaccard con conjuntos

Índice de Dice o puntaje F1. Partiendo desde teoría de conjuntos como en el punto anterior, la intersección refleja aquellos píxeles correctamente clasificados. Al duplicar esta intersección y dividirlos entre el total de píxeles de ambas imágenes se obtiene el índice de Dice. Sus valores se encuentran en el intervalo $[0, 1]$ donde un valor más alto indica un mejor desempeño. La figura 2.2 ilustra esta idea. El índice de Dice se calcula desde la matriz de confusión con la siguiente expresión:

$$\frac{2VP}{2VP + FP + FN} \quad (2.3)$$

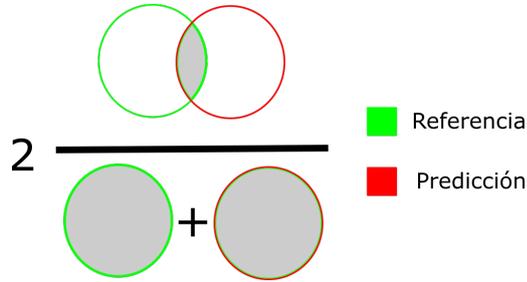


FIGURA 2.2: Interpretación de índice de Dice con conjuntos

Distancia Hausdorff. A diferencia de las métricas como Dice o IoU, que se basan en el traslape de imágenes, esta métrica basada en comparación de formas es una medida longitudinal de la discrepancia entre dos conjuntos. Un valor más bajo representa un mejor desempeño. Determina la distancia de cada punto de la máscara obtenida por un modelo con el la máscara de referencia y viceversa. La distancia *Hausdorff* es la distancia máxima desde un punto de un conjunto al punto más cercano del otro conjunto. Puede considerarse, por ejemplo, la distancia euclidiana. Se caracteriza por ser sensible a valores atípicos [83]. Sean A, B dos conjuntos de puntos, la distancia directa entre A y B o distancia *Hausdorff* unilateral está determinada por:

$$h(A, B) = \sup_{a \in A} \inf_{b \in B} \|a - b\|$$

Donde $\|a - b\|$ se refiere a la distancia entre los puntos a y b , *sup* e *inf* corresponden al supremo e ínfimo, respectivamente. Por tanto, la distancia directa de B a A se expresa como:

$$h(B, A) = \sup_{b \in B} \inf_{a \in A} \|b - a\|$$

Finalmente, la distancia Hausdorff se expresa como:

$$H(B, A) = \max(h(A, B), h(B, A)) \quad (2.4)$$

La figura 2.3 ilustra la idea del cálculo de esta medida.

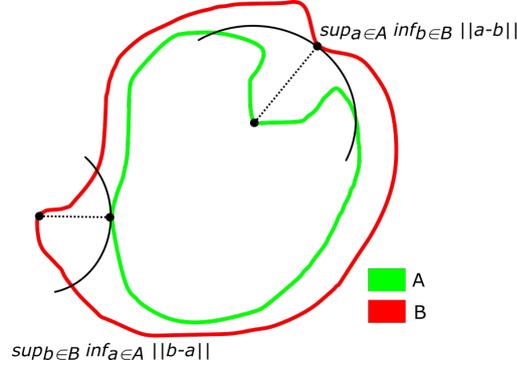


FIGURA 2.3: Interpretación gráfica de la distancia *Hausdorff*

2.2. Redes Neuronales Convolucionales

Las redes neuronales convolucionales, propuestas por Yann LeCun en 1989 [22], están inspiradas en la corteza visual del cerebro, que es la encargada de procesar imágenes con base en los experimentos de David H. Hubel y Torstenn Wilsen en 1968 [38] de donde se desprende el concepto de *campo receptivo* que describe la relación entre el campo visual y las neuronas que procesan esta información [62].

2.2.1. Capas de convolución

Una red convolucional es una red neuronal que tiene una o más capas convolucionales. Cada capa convolucional aplica la operación de convolución (*) sobre una imagen con un filtro dado [62], ver ecuación 2.5.

Una imagen puede definirse como una función de dos dimensiones, $I(x, y)$, que recibe las coordenadas espaciales (fila y columna, respectivamente) de un píxel y devuelve su valor de intensidad. En procesamiento digital de imágenes se requiere de un *muestreo* y una *cuantificación* que consiste en discretizar las coordenadas espaciales y los valores de intensidad, por lo que la imagen digital queda definida por un número *finito* de filas (M), columnas (N) y valores de intensidad (L) [21].

En imágenes en escala de grises se tiene que $L = \{l : l \in \mathbb{Z} \wedge 0 \leq l \leq 2^n - 1\}$, donde n se refiere al número de bits con los que se representa la imagen. De aquí que la función de mapeo que define la intensidad de la imagen es: $I : \mathbb{Z}^2 \rightarrow \mathbb{Z}$. Por otro lado, en las imágenes a color como RGB (por sus siglas en inglés para referirse a *Red, Green, Blue*), el conjunto de niveles de intensidad es $L = \{[l_i] : [l_i] \in \mathbb{Z}^{|C|} \wedge 0 \leq l_i \leq 2^n - 1 \wedge i = 1 \dots |C|\}$ donde C se refiere al conjunto de canales, rojo, verde y azul. De este modo, el mapeo de intensidad es $I : \mathbb{Z}^2 \rightarrow \mathbb{Z}^3$.

En la operación convolución se recorre un filtro o kernel de convolución sobre la dimensión espacial, $M \times N$ que efectúa el producto entre los pesos que contiene el kernel de convolución y el área que éste cubre sobre la imagen, área a la que se le denomina *campo receptivo*. El valor calculado es mapeado a un nuevo arreglo y colocado en la

misma posición del píxel central del campo receptivo, ver figura 2.4. El resultado de aplicar iterativamente uno o más filtros a lo largo y ancho de la imagen (ver figura 2.5) se conoce como *mapa de características*. Para realizar la convolución, el número de canales del filtro aplicado debe coincidir con el número de canales de la imagen de entrada.

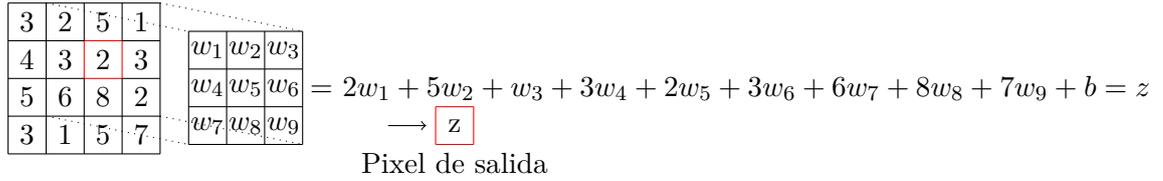


FIGURA 2.4: Operación de convolución entre una imagen en escala de grises de 4×4 con un kernel de 3×3 . El resultado es asignado al píxel central marcado con rojo. w_i y b se denominan los pesos del filtro y sesgo, respectivamente

La expresión matemática que define la operación de *convolución discreta* en dos dimensiones entre dos funciones, $f(x, y)$ y $h(x, y)$ con M filas y N columnas, se define en la ecuación 2.5.

$$(f * h)(x, y) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} f(i, j)h(x - i, y - j) \quad (2.5)$$

Es importante mencionar que una operación estrechamente relacionada con la convolución es la *correlación cruzada* (\star), que se caracteriza por aplicar una rotación de 180° al kernel de convolución, se expresa en la ecuación 2.6

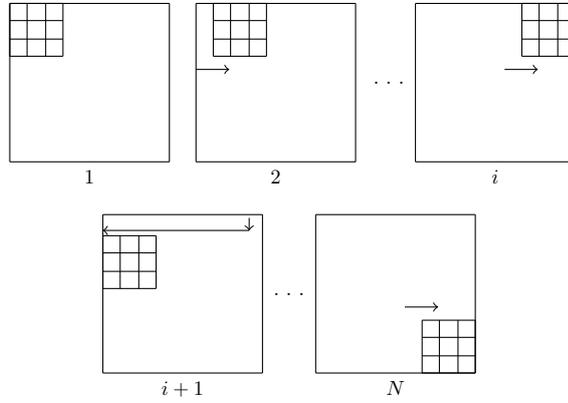
$$(f \star h)(x, y) = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} f(i, j)h(x + i, y + j) \quad (2.6)$$

En la operación de convolución intervienen otros parámetros, entre los que destacan:

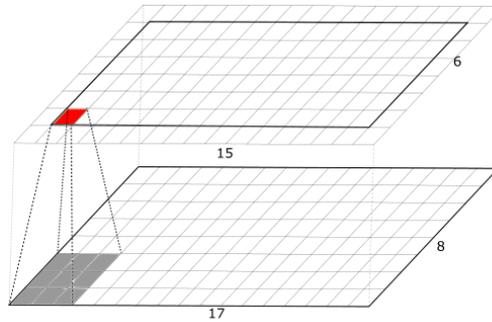
Padding. Consiste en añadir bordes a las imágenes de acuerdo a un criterio y así poder aplicar los filtros alrededor de los bordes originales de la imagen. Uno de los más utilizados es *same-padding*, que añade tantos bordes como sean necesarios para que al aplicar la convolución, las dimensiones de la imagen de entrada se mantengan a la salida.

Stride o tamaño de paso. Se refiere a cuántos píxeles se recorrerá el kernel sobre la imagen de forma iterativa.

Tasa de dilatación. Añade espacios entre los elementos de un filtro de modo que se cubra un mayor campo receptivo de forma más barata. Una tasa de dilatación equivalente a 1 se considera como una convolución regular. En la figura 2.6 se



(A) Recorrido del filtro 3x3 a lo largo de la imagen



(B) Capa de convolución resultante de aplicar un *kernel* 3x3 a una imagen de un solo canal

FIGURA 2.5: Capa de convolución aplicada a una imagen de 17 x 8 píxeles. Notar que las dimensiones de la imagen se reducen a la salida de la operación de convolución con dimensiones de 15 x 8

muestra un ejemplo al considerar una tasa de dilatación equivalente a 2 con un filtro de dimensión 3x3.

Las dimensiones de salida de una capa de convolución dependen tanto de los parámetros mencionados anteriormente como del tamaño del filtro seleccionado. Asumiendo una entrada y un filtro con dimensiones cuadradas, las dimensiones de salida están dadas por la ecuación 2.7

$$salida_{dim} = \left\lfloor \frac{M + 2P - D(K - 1) - 1}{S} + 1 \right\rfloor \quad (2.7)$$

donde M son el número de filas y columnas de la imagen de entrada, P es el número de bordes añadidos a la imagen con padding, D es la tasa de dilatación y S el stride.

En las redes neuronales las *funciones de activación* son de suma importancia ya que ayudan al aprendizaje y la extracción de características de mayor complejidad utilizando funciones no-lineales. La función ReLU (*Rectified Linear Unit* por sus siglas en inglés) se ha popularizado en el campo del aprendizaje profundo principalmente por su bajo costo

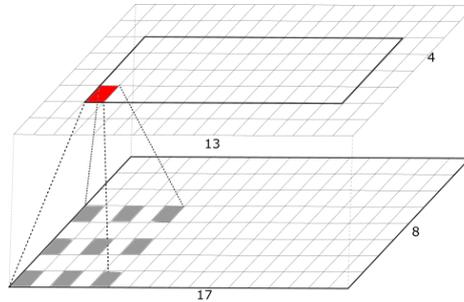


FIGURA 2.6: Convolución dilatada aplicada a una imagen de 17 x 8 píxeles con un filtro de 3 x 3 y una tasa de dilatación equivalente a 2. Notar las diferencias en los tamaños de salida al compararla con la figura 2.5b

computacional en comparación con las funciones logísticas y la ventaja de contender contra el desvanecimiento del gradiente al utilizar *retropropagación* como mecanismo de entrenamiento [20]. Se define como:

$$\text{ReLU}(x) = \max(x, 0) \quad (2.8)$$

2.2.2. Convolución transpuesta

Generalmente, la convolución transpuesta se utiliza por la necesidad de recuperar las dimensiones de la imagen o mapa de características de una operación de convolución [13]. Es aplicable en métodos donde se utilizan arquitecturas del tipo *encoder-decoder* como U-Net. Una propiedad interesante de la convolución transpuesta es la posibilidad de emularla con una convolución directa o regular. Sean k, s, p los valores correspondientes al tamaño del filtro, stride y padding de una convolución regular, respectivamente. Existe una convolución transpuesta relacionada cuyos parámetros toman los valores $k' = k, s' = 1$ y $p' = k - p - 1$.

Por ejemplo, tomando $i = 4, k = 3, s = 1, p = 0$ como las dimensiones de las entradas de una convolución, su equivalente convolución transpuesta sería $i' = 2, k' = k, s' = 1$ y $p' = 2$. En la figura 2.7 se ilustra la idea de la convolución transpuesta.

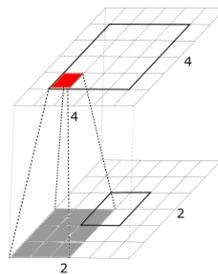


FIGURA 2.7: Ejemplo de convolución transpuesta. Notar que las dimensiones de la imagen de salida son mayores a la de la imagen de entrada

2.2.3. Convolución separable en profundidad

Con el fin de reducir los parámetros entrenables de una red neuronal, la convolución separable en profundidad ha sido aplicada en distintos modelos [11, 32]. Asumiendo que se tiene una imagen de 3 canales, rojo, verde y azul, la convolución regular se aplica sobre la dimensión de profundidad o dimensión de canales (ver figura 2.8). En contraste, la convolución separable en profundidad consiste en una convolución factorizada que se compone de dos etapas. La primera de ellas se denomina etapa de filtrado donde se aplica una convolución en profundidad (*depth wise convolution*), esto es, el mapa de características de entrada se divide en el número de canales, M . A cada canal, $m_i \in M$, se aplica el filtro correspondiente de forma aislada y los resultados se concatenan en la dimensión de canales (ver figura 2.9a). Posteriormente, en la etapa de combinación, una segunda convolución regular con un filtro de tamaño 1×1 se aplica sobre el mapa de características, denominada convolución por puntos o *pointwise convolution* (ver figura 2.9b). Utilizando este tipo de convolución el tiempo de entrenamiento y latencia se ven disminuidos. A continuación un breve análisis de la reducción de los parámetros.

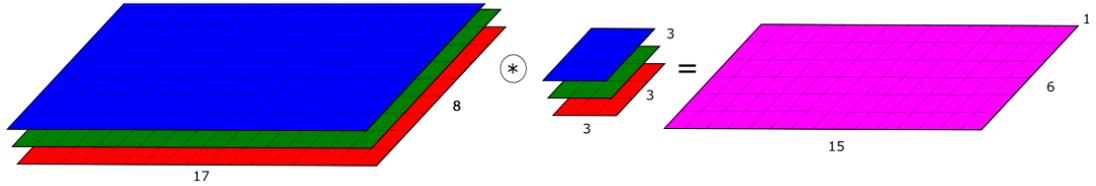


FIGURA 2.8: Ejemplo de convolución regular para una imagen de entrada de 3 canales (RGB) con un filtro de 3×3

Asumiendo N filtros de tamaño D_k , el número de parámetros en la operación de convolución regular se expresa como:

$$N \times M \times D_k^2 \quad (2.9)$$

Por otro lado, al utilizar convolución separable en profundidad, el número de parámetros está dado por la suma de los parámetros de cada una de las etapas:

$$M \times D_k^2 + N \times M = M(D_k^2 + N) \quad (2.10)$$

Al aplicar 32 filtros de tamaño 3×3 a una imagen RGB ($M = 3$), el número de parámetros es $32 \times 3 \times 9 = 864$. Utilizando convolución separable en profundidad, el número de parámetros se reduce a solo a $3(9 + 32) = 123$.

Finalmente, se determina la relación entre las expresiones anteriores (2.10 2.9) para determinar qué tanto decrecen los números de parámetros al utilizar la convolución separable en profundidad en términos de la cantidad de filtros aplicados y el tamaño del kernel.

$$\frac{M(D_K^2 + N)}{N \times M \times D_K^2} = \frac{1}{N} + \frac{1}{D_K^2} \quad (2.11)$$

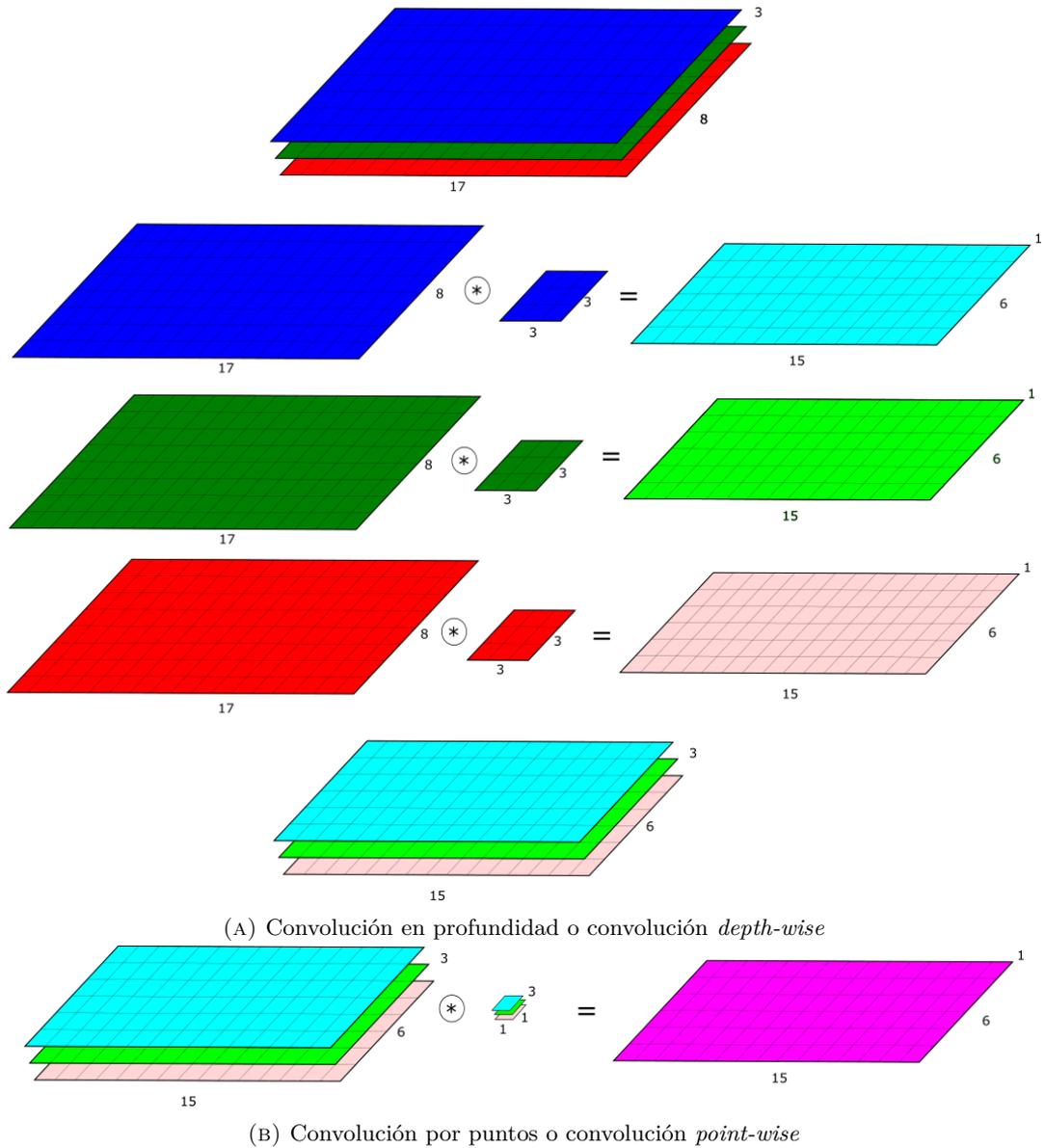


FIGURA 2.9: Ejemplo de convolución separable en profundidad para una imagen de entrada de 3 canales (RGB) con un filtro de 3 x 3

2.2.4. Capas de pooling

En las redes convolucionales existe otro tipo de capas, denominada capas de *pooling*, que realizan una operación de muestreo, reduciendo la resolución espacial de la imagen. Este tipo de operaciones permiten a la red reducir el número de parámetros entrenables y por tanto, su costo computacional. Se aplica recorriendo una máscara que define una vecindad sobre la entrada de la capa. Con los píxeles delimitados por el área de la máscara se determina una operación que identifica el máximo o el promedio del conjunto de píxeles, operación conocida como *max-pooling* y *avg-pooling*, respectivamente. Es importante señalar que esta operación no agrega parámetros entrenables a la red neuronal. En la figura 2.10 se muestra un ejemplo de operación *max-pooling*:

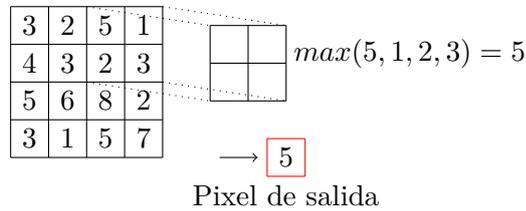


FIGURA 2.10: Ejemplo de una operación max-pooling

Usualmente una red convolucional está compuesta por una o más capas de convolución seguidas de una capa de pooling e intercaladas una tras otra [62].

2.2.5. Capas de normalización

Durante el entrenamiento de las redes neuronales, la distribución de los pesos se ve afectada y alenta el proceso de entrenamiento, fenómeno conocido como cambio de covariable interna (*internal covariate shift*) [40]. Una forma de mitigar este problema es mediante una normalización. La normalización consiste en modificar los datos del conjunto o subconjunto de entrada de modo que compartan una media de cero y una varianza de 1 mediante el siguiente cálculo:

$$\hat{x}_i = \frac{1}{\sigma_i}(x_i - \mu_i) \tag{2.12}$$

donde μ y σ en la ecuación 2.12 corresponden a la media y la desviación estándar del conjunto seleccionado, respectivamente, calculados como:

$$\mu_i = \frac{1}{m} \sum_{k \in \mathcal{S}_i} x_k \tag{2.13}$$

$$\sigma_i = \sqrt{\frac{1}{m} \sum_{k \in \mathcal{S}_i} (x_k - \mu_i)^2 + \epsilon} \tag{2.14}$$

donde m representa la cardinalidad del conjunto \mathcal{S}_i y ϵ es una constante añadida para fines de consistencia numérica evitando indeterminaciones en el cálculo de \hat{x}_i . Para compensar la posible pérdida de representación dos parámetros más se ven involucrados en la capa de normalización, γ y β , parámetros entrenables que ajustan la *escala y el desplazamiento* de los valores normalizados, de modo que la transformación aplicada tenga oportunidad de representar una función identidad [40]. Finalmente el valor asignado se representa como:

$$y_i = \hat{x}_i + \beta \quad (2.15)$$

De acuerdo al conjunto considerado, \mathcal{S}_i , para estimar las estadísticas, distintos tipos de normalización de características se han propuesto [82]. A continuación se describen los tipos de normalización de características mencionados en [82]. Considere B el tamaño del lote, C el número de canales y H, W la altura y ancho de cada mapa de características, respectivamente.

Normalización por lote [40].

La normalización por lote o BN (por sus siglas en inglés para *Batch Normalization*) es quizá la técnica de normalización más común en redes neuronales convolucionales. El conjunto sobre el cual son estimadas las estadísticas se define como $\mathcal{S}_i = \{k \mid k_C = i_C\}$, donde C es el eje sobre el que se iteran los cálculos, por lo que las estadísticas de la media y varianza se calculan a lo largo de los ejes (N,H,W). Ver figura del lado izquierdo en 2.11.

Normalización por capa [1].

La normalización por capa o LN (por sus siglas en inglés para *Layer Normalization*) considera el conjunto $\mathcal{S}_i = \{k \mid k_N = i_N\}$, donde N representa que los cálculos se realizan sobre cada elemento del lote, es decir, a lo largo de los ejes (C,H,W). Este tipo de normalización ha sido aplicada efectivamente en redes recurrentes o modelos secuenciales. Ver figura central izquierda en 2.11.

Normalización por instancia [75].

En la normalización por instancia o IN (por sus siglas en inglés para *Instance Normalization*) el conjunto $\mathcal{S}_i = \{k \mid k_N = i_N, k_C = i_C\}$. Es decir, las estadísticas se calculan a lo largo de los ejes (H,W), por cada canal y por cada elemento en el lote. Este tipo de normalización ha sido aplicada satisfactoriamente en redes generativas. Ver figura central derecha en 2.11

Normalización por grupo [82].

La normalización por grupo o GN (por sus siglas en inglés para *Group Normalization*) considera el conjunto $\mathcal{S}_i = \{k \mid k_N = i_N, \lfloor \frac{k_C}{C/G} \rfloor = \lfloor \frac{i_C}{C/G} \rfloor\}$. Esto es, las estadísticas se calculan sobre los ejes (H,W) y por cada grupo compuesto por $\frac{C}{G}$ canales, donde G representa el número de grupos considerado. Ver figura de la izquierda en 2.11. Este tipo de normalización se ha aplicado satisfactoriamente

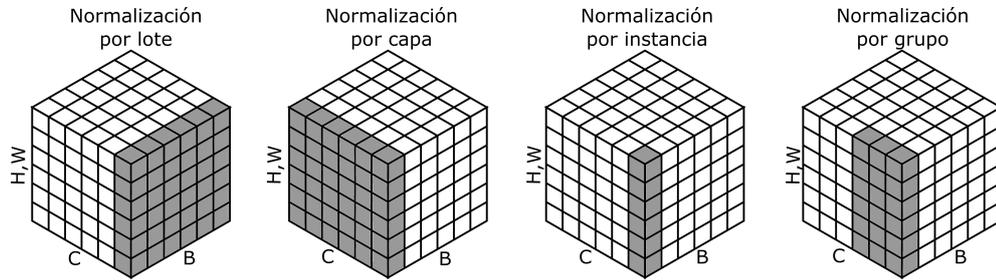


FIGURA 2.11: Tipos de normalización de acuerdo con el conjunto \mathcal{S}_i seleccionado

en tareas de segmentación, donde el tamaño de lote escogido regularmente toma valores de 1 o 2 [60] y ha tenido mejores resultados en comparación con BN .

2.2.6. Arquitectura Res-Net

Añadir capas de convolución al modelo aumenta la complejidad del modelo pero no necesariamente su desempeño. Se ha visto en la literatura que una conectividad no contigua permite obtener mejores resultados al aumentar la profundidad de la red.

La arquitectura Res-Net [27] introduce conexiones atajo o *skip connections* a las capas de convolución permitiendo afrontar el problema del desvanecimiento del gradiente, esto es, el atenuamiento del cambio de los pesos sinápticos en las capas intermedias al entrenar la red mediante el algoritmo de *retropropagación* [4, 19]. Además, permite obtener un mejor desempeño al aumentar la profundidad de las redes convolucionales en tareas de clasificación. Una característica de este tipo de red es la ausencia de capas de pooling y el uso de convoluciones con *stride* en su lugar.

En la figura 2.12 se muestra un bloque residual utilizado por Res-Net, que aplica una conexión atajo cada dos capas convolucionales. A través de la conexión atajo o *skip connection* se aplica una capa de convolución con un tamaño de kernel de 1, convolución *point-wise* que permite adaptar los canales de entrada a los de salida de ser necesario. La salida de la capa convolucional y de la conexión atajo se combinan mediante una operación de adición, previa a la función de activación ReLU.

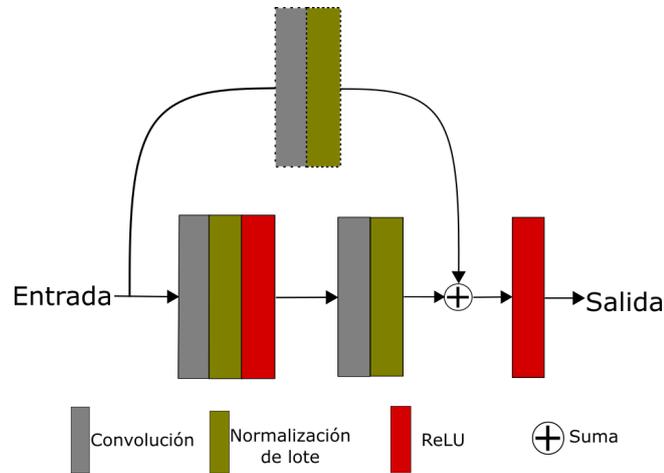


FIGURA 2.12: Bloque residual utilizado por Res-Net. Las líneas punteadas representan una capa de convolución con un tamaño de kernel de 1

2.2.7. Arquitectura Dense-Net

La arquitectura Dense-Net [37] extiende la idea de conexiones atajo o *skip connections* y omite el uso de la operación de adición. En su lugar introduce la operación de concatenación bajo un enfoque de *aprendizaje colectivo*. Entre sus ventajas, al igual que Res-Net, destaca la mitigación del desvanecimiento del gradiente, el fortalecimiento de la propagación y reuso de características y la reducción del número de parámetros.

La arquitectura Dense-Net se compone por bloques densos y bloques de transición. Los primeros de ellos llevan a cabo la conexión densa, donde para cada capa de convolución, los mapas de características de las capas de convolución anteriores fungen como entradas a todas las capas posteriores. Por otro lado, los bloques de transición permiten no solo reducir la resolución de los mapas de características obtenidos en el bloque denso (pooling) sino también reducir el incremento del número de canales, dada la operación de concatenación, con base en un *factor de compresión*, $\theta \in (0, 1]$, mediante una capa de convolución con un kernel de tamaño 1. Los bloques densos se caracterizan por sus pre-activaciones, aunque distintas variantes se han propuesto [28]. El parámetro k representa el número de *canales de salida por cada capa de convolución*.

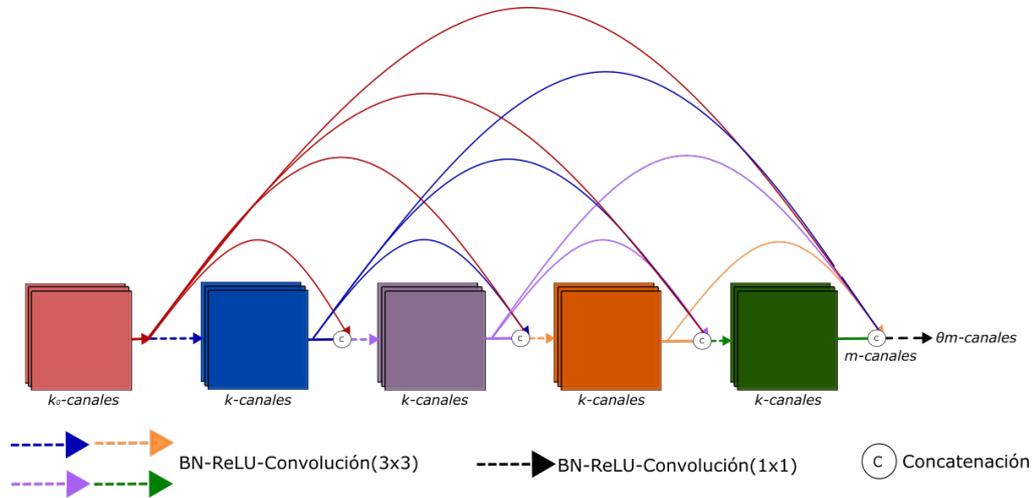


FIGURA 2.13: Bloque denso utilizado por Dense-Net. Los mapas de características de entrada correspondientes a las capas anteriores se distinguen mediante los colores correspondientes. A la salida se aplica una convolución con un tamaño de kernel de 1×1 que reduce los canales de salida.

2.2.8. Contracción y Excitación

La operación de contracción y excitación o *squeeze-and-excitation* [33] introduce pesos entrenables extra a las redes convolucionales, lo que permite enfatizar o suprimir determinados canales de los mapas de características utilizando un perceptrón multicapa interno, encargado de modelar la interdependencia entre canales. Entre sus ventajas destaca la escalabilidad para ser aplicado a distintos tipos de redes que mejoran el poder de representación del modelo. En la figura 2.14 se muestra un bloque de contracción y excitación o *squeeze-and-excitation*.

Los componentes de esta operación son: i) contracción, ii) excitación y iii) escalamiento. A continuación se describen cada uno de ellos.

Contracción.

Se extrae información global de cada canal del mapa de características de modo que pueda determinarse un descriptor por cada canal. Se utiliza un *pooling* promedio general.

Excitación.

Un perceptrón multicapa es utilizado. La capa oculta reduce las neuronas en un factor r . La capa de salida recupera las dimensiones del vector de entrada y una función sigmoideal es aplicada.

Escalamiento.

Es la multiplicación, elemento a elemento, de la salida de la función sigmoideal y el mapa de características de entrada.

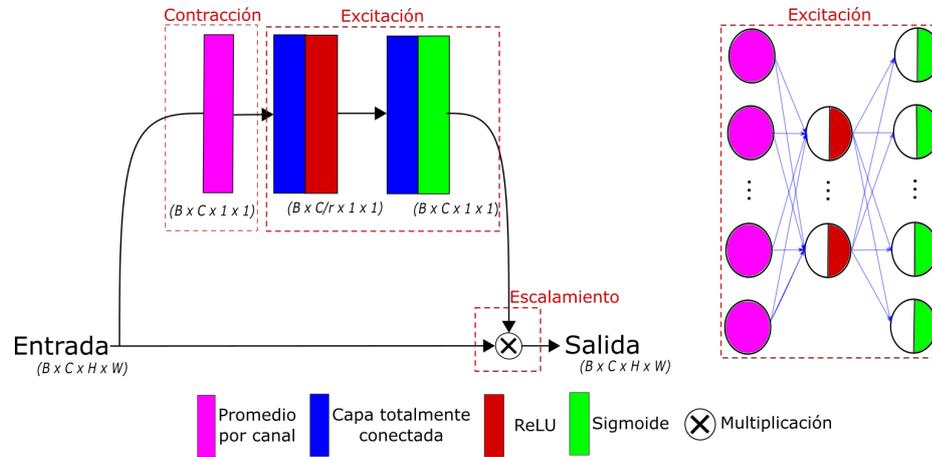


FIGURA 2.14: Bloque SE, por sus siglas para *Squeeze-and-Excitation*. La figura del lado derecho muestra a detalle la etapa de excitación

2.3. Algoritmos Evolutivos

Los Algoritmos Evolutivos (AE) son metaheurísticas estocásticas para resolver principalmente problemas de optimización [51]. Cada solución encontrada es evaluada en una función objetivo con el fin de minimizarla o maximizarla. Sin embargo, es posible quedarse en algún mínimo local en el espacio de búsqueda. Al tomar en cuenta restricciones, la función o funciones objetivo y restricciones se denomina paisaje de búsqueda, el cual está definido por la formulación del problema (número de parámetros, rango de parámetros, restricciones y objetivos) [51]. Se trata de un optimizador en un lazo de retroalimentación mediante el cual el sistema verifica las aptitudes de las salidas y realiza operaciones con ellas para ingresarlas nuevamente hasta cumplir con las condiciones de la función objetivo. Este proceso requiere de varias iteraciones.

Estos algoritmos consideran una población de soluciones de entre las cuales algunas son seleccionadas con base en aptitud para recombinarlas y mutarlas y fomentar así la exploración del espacio de búsqueda y evitar caer en mínimos locales. La idea principal de estos algoritmos es mejorar las soluciones iniciales tomando en cuenta la *exploración* y la *explotación* del espacio de búsqueda. Es decir, tomar una variedad de candidatos (*exploración*) los cuales se refinan con el paso de las iteraciones (*explotación*). Los algoritmos evolutivos son los siguientes:

- Algoritmos Genéticos
- Programación Genética
- Programación Evolutiva
- Estrategias Evolutivas
- Evolución Diferencial

En términos generales, los componentes principales en cualquier algoritmo evolutivo son los siguientes:

- *Representación.* Se considera uno de los pasos más importante en el diseño de un algoritmo evolutivo que depende en gran medida del dominio del problema que se quiere resolver. Tanto el desempeño como la complejidad del espacio de búsqueda dependen del esquema de representación.
- *Función de evaluación.* Consiste en una evaluación cuantitativa de la calidad de una solución conocida como aptitud del individuo.
- *Mecanismos de selección.* Este paso da pauta al sesgo hacia mejores soluciones con base en la aptitud de cada individuo. Aplican para la selección de padres (quiénes se reproducen) y para la selección de sobrevivientes quienes sobreviven para la siguiente generación.
- *Operadores de variación.* La cruza y mutación son los operadores que usualmente se emplean en los algoritmos evolutivos, los cuales permiten generar nuevas soluciones a partir de las ya existentes.
- *Criterio de parada.* Se refiere la condición para detener la ejecución del algoritmo. Entre los criterios de parada más utilizados se encuentran:
 - Número máximo de generaciones
 - Número máximo de evaluaciones
 - Baja diversidad en la población
 - No se observa una mejora de la calidad de las soluciones durante un cierto número de generaciones
 - Se encuentra una solución aceptable

2.3.1. Neuroevolución

La neuroevolución consiste en la aplicación de algoritmos evolutivos para determinar la topología de una red neuronal y/o su correspondiente fase de aprendizaje. Ambas características son de vital importancia para obtener resultados favorables. [18]. En la literatura es común que se clasifique a la *neuroevolución* en dos campos: el entrenamiento de la red y la búsqueda de una topología. Este trabajo se enfoca sobre el segundo punto ya que para el entrenamiento de la red se recurre al conocido algoritmo de retropropagación [10]. La evolución de topologías de redes convolucionales se ha alcanzado mediante dos vías: algoritmos evolutivos y aprendizaje por refuerzo. El primero de ellos ha sido utilizado ampliamente aunque cuenta con algunas desventajas, como lo es el alto costo computacional [14].

Los algoritmos genéticos (GA) para tareas de neuroevolución son los dominantes en la literatura. Como se mencionó al inicio de la sección 2.3 uno de los puntos primordiales en el desarrollo de un esquema, no solo de neuroevolución, sino de los algoritmos

evolutivos en general es la representación o codificación de las soluciones potenciales. En [78] se propone una taxonomía con base en las representaciones, que pueden clasificarse en codificación directa (nivel fenotipo), indirecta (nivel genotipo) e híbrida, como una combinación de las representaciones a distintos niveles para las cuales se desarrollan operadores de variación independientes. Para este trabajo nos centraremos en el paradigma de PG aplicado a neuroevolución el cual utiliza una representación directa (árbol), de acuerdo con la taxonomía mencionada.

2.3.2. Programación Genética

La Programación Genética [43] se caracteriza por utilizar una codificación de árbol para representar soluciones potenciales considerada como directa. Entre sus ventajas destaca la facilidad para construir o decodificar dicho árbol a su fenotipo correspondiente [78]. En la figura 2.15 se presenta un ejemplo de árbol que representa a un programa con funciones aritméticas. En PG destacan dos conceptos fundamentales para generar árboles o soluciones potenciales, cuya unión se conoce como conjunto de primitivas.

- *Conjunto de terminales.* Se constituye por las entradas del programa, como lo son variables y constantes. También se consideran como terminales aquellas funciones de aridad cero u operaciones nularias, es decir, operaciones que no requieren de argumentos para que puedan calcular un valor, por ejemplo, la generación de un número aleatorio. Debido a la aleatoriedad que representan las llamadas a rutinas de aridad 0, las *constantes efímeras aleatorias* son terminales que se crean solo una ocasión y se mantienen durante todo el proceso evolutivo.
- *Conjunto de funciones.* Las funciones consideradas en este conjunto dependen en gran medida de la naturaleza del problema, por lo que cada función se especializa en el dominio particular y éstas a su vez, definen parte del conjunto de terminales [55]. Los argumentos de cada función están dadas por el conjunto de terminales o por la salida de alguna otra función.

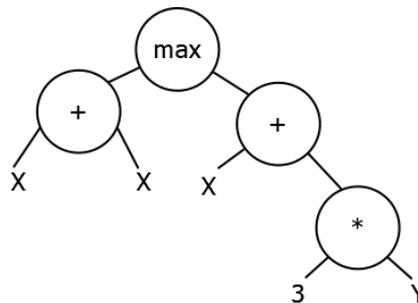


FIGURA 2.15: Ejemplo de programa representado como un árbol. La función considerada determina el máximo entre dos números, dado por $max(x + x, x + 3 * y)$. En este ejemplo el conjunto de funciones es $\{max, +, *\}$, mientras que el conjunto de terminales lo constituyen $\{X, Y, 3\}$

Con el fin de obtener árboles que representen una solución válida, Montana [52] presenta la Programación Genética Fuertemente Tipificada (STGP por sus singlas en inglés) en el que se declara un tipo de dato para cada función y terminal considerada. De este modo se asegura una consistencia de tipo y evita generar árboles que no son legales dentro del dominio del problema.

En PG han surgido variantes que no utilizan una representación de árbol, sino de un grafo acíclico dirigido, como es el caso de la Programación Genética Cartesiana (PGC) [50]. Se caracteriza por codificar las funciones y sus argumentos en un vector. Una de sus principales ventajas es que carecen del efecto bloat (ver subsección 2.3.2), sin embargo, dificulta la aplicación de operadores como la cruza, presentan redundancia y, dada su representación, la complejidad de la codificación aumenta en comparación con la PG basada en árboles.

Población inicial

En PG la población inicial consiste en una colección de individuos generados aleatoriamente con base en el conjunto de funciones y conjunto de terminales. En la literatura se encuentran 3 criterios para generar los árboles de manera aleatoria: método *full*, donde todas las hojas del árbol comparten la misma profundidad; método *grow*, que crea árboles de distintas formas y tamaños; método *ramped half and half*, que crea una población construyendo la mitad de la población por el método *full* y la otra mitad con el método *grow*, tomando en cuenta una profundidad mínima y máxima definida por el usuario.

Función de evaluación

Los árboles generados cubren un amplio espacio de búsqueda de acuerdo al conjunto de primitivas. Para determinar cuáles son las mejores soluciones se establece una medida de aptitud que dicta a la PG los requisitos del problema a resolver. Esta tarea suele ser la de mayor costo computacional y por ende, de mayor tiempo de ejecución.

Mecanismos de selección

El mecanismo de selección más popular en PG es la selección mediante torneo. Se seleccionan un número de individuos y se comparan entre ellos. Aquel individuo que representa la mejor solución en este torneo es seleccionado como padre de la siguiente generación. El número de individuos que participan en el torneo es un parámetro del proceso evolutivo, dado por el usuario y que está ligado fuertemente al concepto de presión de selección, es decir, la dureza o fortaleza del proceso de selección.

Operadores de variación

En PG los operadores de variación son muy distintos a los utilizados en otros paradigmas evolutivos. Usualmente, la operación de cruza más utilizada es la cruza de un punto. La cruza, como una operación sexual, toma un par de padres y selecciona un nodo aleatoriamente en cada uno de ellos el cual toma el rol del punto de cruza. Después,

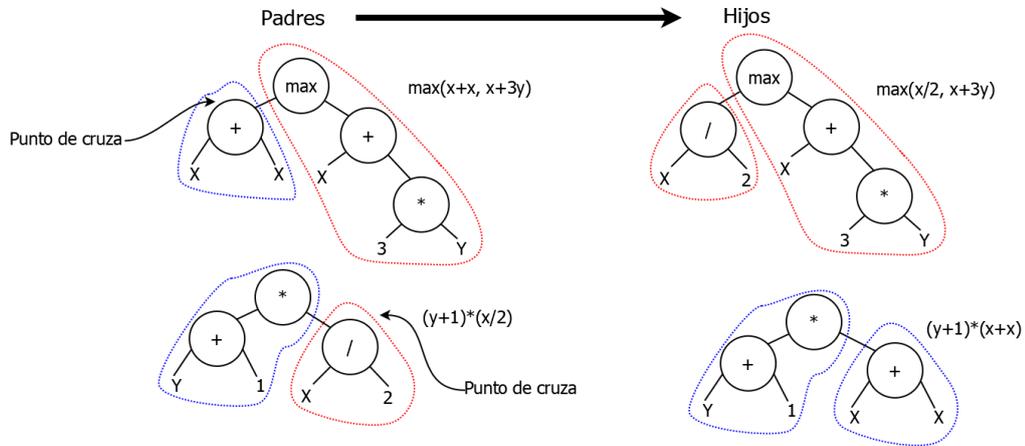


FIGURA 2.16: Ejemplo de cruce de un punto en PG. Notar que este tipo de cruce genera 2 hijos, aunque pudiera implementarse una versión que genere únicamente 1 hijo descartando alguno de ellos

los subárboles generados son intercambiados para generar así la descendencia. La figura 2.16 ilustra la idea de esta operación.

La operación de mutación no fue considerada originalmente por [43], sin embargo, estudios más recientes utilizan este operador con una probabilidad baja en comparación a la probabilidad de cruce [55]. Se trata de una operación asexual que realiza un cambio aleatorio en el individuo. Distintos tipos de mutación han sido propuestos, donde destaca la mutación de sub-árbol. Esta consiste en seleccionar un punto aleatorio en el árbol y reemplazar el sub-árbol correspondiente con uno nuevo generado aleatoriamente. La figura 2.17 ilustra la idea de este operador.

Es importante destacar que en PG, los operadores de variación son aplicados exclusivamente a diferencia de los GA, que usualmente se aplican como una composición de éstos. La probabilidad de cruce y mutación, parámetros del algoritmo evolutivo, deben cumplir que la suma de éstos sea menor o igual a 1. En el caso de ser menor se efectúa una tercera operación denominada *reproducción* que selecciona aleatoriamente un individuo de la población con base en su aptitud y lo copia a la siguiente generación. La probabilidad corresponde a la resta entre 1 y la suma de probabilidades de cruce y mutación.

Efecto *bloat*

Dependiendo de los operadores de cruce y mutación utilizados, los individuos pueden caer en un efecto negativo de PG denominado *bloat*, que toma lugar cuando la profundidad de los árboles aumenta sin reflejar una mejora en la aptitud. Distintos mecanismos para evitar esto se proponen en PG, como establecer un límite de profundidad de modo que, si algún individuo producto del operador de variación supera este límite, uno de los padres es tomado aleatoriamente en su lugar. Otros mecanismos incluyen operadores de

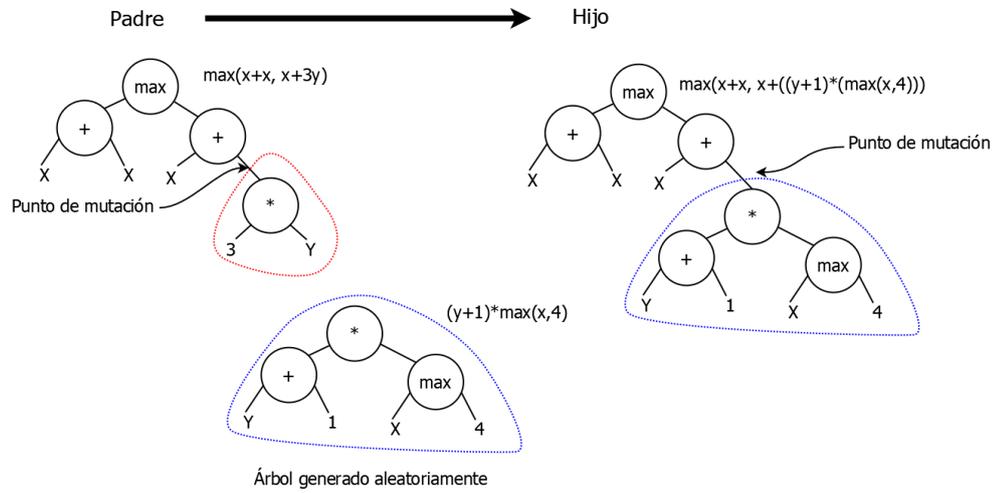


FIGURA 2.17: Ejemplo de mutación de subárbol en PG

mutación que contienen contra este efecto, como lo es *shrink mutation* o mutación de contracción. Se aplica de manera similar a la mutación de sub-árbol con la restricción que el árbol generado aleatoriamente será una terminal o nodo hoja.

Capítulo 3

Marco referencial

En los próximos párrafos se abordan estudios y trabajos cuyo dominio y técnicas son similares a nuestro enfoque. Los trabajos abordados cubren 3 áreas distintas. Primeramente se toman en cuenta los métodos utilizados para segmentar el órgano de la placenta que se trata de nuestra tarea principal. En segundo lugar, se abordan algunos estudios que muestran cómo es que PG ha sido aplicada en el área de neuroevolución y finalmente, se abordan los trabajos donde la neuroevolución ha sido utilizada para tareas de segmentación.

3.1. Segmentación de placentas

Una de las técnicas tradicionales para monitorear no solo la placenta sino también el proceso de gestación durante el embarazo es el ultrasonido, estudio generalmente realizado en un consultorio de ginecología y/o obstetricia. Tiene sus orígenes entre los años de 1930 y 1940. En el ámbito médico el ultrasonido surge como tratamiento de cáncer y de terapia física. Theodore Dussik y su hermano trataron de utilizarlo para diagnosticar tumores en el cerebro [29, 54].

Imágenes bidimensionales de ultrasonido pueden otorgar suficiente información para determinar características de la placenta como su ubicación, tamaño, forma y madurez [30]. Particularmente, es de gran interés identificar la placenta para poder asociar anomalías y así dar indicios y advertir al médico sobre posibles riesgos que atenten a una gestación saludable tanto para el feto como la madre. Una herramienta de segmentación adicional que apoye el diagnóstico del médico requiere robustez ante el ruido inherente en las imágenes de ultrasonido.

En los siguientes párrafos se describen brevemente algunos estudios de la literatura especializada donde se documenta la segmentación de placentas.

3.1.1. Gordon N. Stevenson, et al. (2015) [64]

En este estudio se utiliza el algoritmo *random walker* para segmentar la placenta en imágenes de ultrasonido 3D con el fin de evaluar su volumen y morfología. Buscan

superar el método usado por *Virtual Organ Computer aided analysis (VOCAL)*, perteneciente a la empresa General Electric especializados en la fabricación de equipo médico. Tomando en cuenta pocos píxeles etiquetados, teoría de circuitos eléctricos, de grafos y probabilísticos e implementado en C++, obtienen un índice de Dice de 0.87.

3.1.2. Pádraig Looney, et al (2017) [47]

De este estudio se destaca el uso de una red convolucional como herramienta de segmentación, una arquitectura denominada *Deep Medic* con 11 capas de convolución [42] y entrenada durante 35 épocas, alcanzando un tiempo de ejecución de 26 horas. Al igual que el estudio anterior también está enfocado en imágenes tridimensionales de ultrasonido y además utilizan *random walker algorithm* para etiquetar las imágenes. Se tomaron en cuenta 300 imágenes de las cuales 240 se utilizaron para entrenar, 40 para validar y 20 para probar el desempeño de segmentación. El índice de Dice que corresponde a la mediana del conjunto de prueba fue de 0.73.

3.1.3. Ricky Hu, et al. (2019) [34-36]

Los trabajos realizados por el autor, una tesis y un par de artículos, son de gran interés para este trabajo ya que se tiene en común el dominio en el que se desarrolla, imágenes de ultrasonido 2D. Toma en cuenta el filtrado de sombras acústicas como preprocesamiento y alcanza el índice de Dice más alto encontrado, 0.92. Propone una arquitectura denominada *U-Net* [60]. Se trata de una red convolucional caracterizada por utilizar convolución transpuesta en lugar de capas completamente conectadas. Consiste en un camino de contracción y otro de expansión con conexiones puente entre ellos. Para este experimento utilizan 1364 imágenes de las cuales el 15 % son para validación, 15 % para prueba y el resto para entrenamiento. La arquitectura utilizada se muestra en la figura 3.1

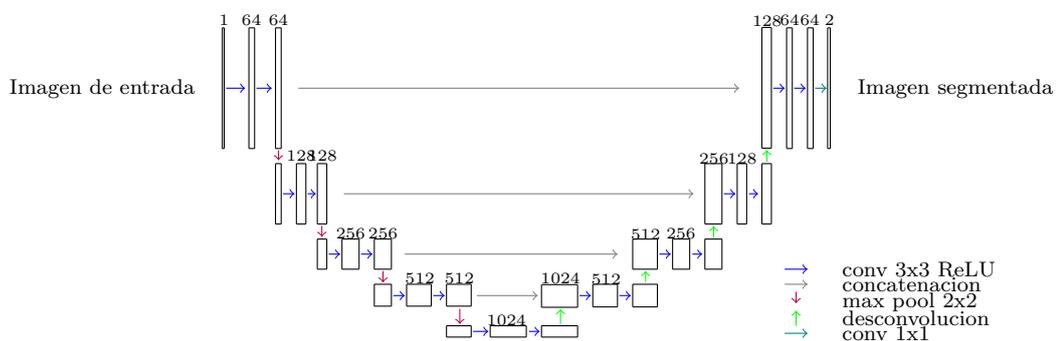


FIGURA 3.1: Arquitectura *U-Net*

3.1.4. Han, et al (2019) [24]

La trascendencia de este artículo [24] en esta comparación con los anteriores no reside en el uso de imágenes de ultrasonido, sino en el uso de la arquitectura *U-Net* para segmentar la placenta en imágenes de resonancia magnética (MRI). Además proponen ligeras modificaciones a esta arquitectura con el fin de reducir el tamaño del modelo, como lo es el uso de la convolución separable en profundidad. Pese a que no señalan explícitamente el índice de Dice obtenido, éste es graficado para la arquitectura U-Net utilizadas alcanzando un valor alrededor de 0.7.

3.1.5. Zimmer, et al (2020) [85]

Utilizando una red neuronal convolucional y bajo un enfoque multitarea aumentan el conjunto de datos de entrenamiento e incluyen información acerca de la variabilidad de los datos haciendo uso de una tarea auxiliar, la cual consiste en determinar la posición de la placenta utilizando aprendizaje por transferencia y, de nuevo, la arquitectura *U-Net*. El entrenamiento se realiza sobre imágenes de ultrasonido 2D extrayéndolas desde las imágenes 3D. Para la implementación utilizan Python y la librería de PyTorch, homogeneizando el tamaño a imágenes de 128x128 píxeles. El índice de Dice alcanzado fue de 0.87 [85].

3.1.6. Comentarios

Los estudios revisados muestran una tendencia a utilizar la arquitectura U-Net o algunas de sus variantes para segmentar la placenta, no solo en imágenes de bidimensionales de ultrasonido, sino también en imágenes tridimensionales y de resonancia magnética. Es importante destacar el estudio realizado por Ricky Hu [36] con el que se comparte el dominio de imágenes, es decir, imágenes bidimensionales de ultrasonido. Cabe mencionar que las imágenes utilizadas en cada estudio son propias y no se encuentran disponibles públicamente.

3.2. Neuroevolución basada en PG

LA PG y algunas de sus variantes como PGC han sido aplicada en el diseño de arquitecturas de redes neuronales, generalmente en tareas de clasificación de imágenes. La trascendencia de los artículos que se muestran a continuación radica en el uso de operadores, funciones y terminales considerados por PG para neuroevolución.

3.2.1. Suganuma, et al (2017) [66, 67]

En [67] se propone el método CGP-CNN (*Cartesian Genetic Programming and Convolutional Neural Networks*), consiste en una codificación basada en bloques que utiliza PGC para clasificar imágenes. Toma en cuenta bloques residuales y considera la suma y concatenación como parte del conjunto de funciones. Una de las desventajas observadas

fue el alto costo computacional. Como conjunto de datos de prueba, utilizan CIFAR-10. En un trabajo más reciente [66] diseña algunas técnicas para una convergencia más rápida como lo es un criterio de terminación de entrenamiento.

3.2.2. Zhu, et al (2018) [84]

Zhu propone el método denominado GP-CNAS (*Genetic Programming for Convolutional Neural Architecture Search*), un método semi-automático basado en PG con representación de árbol que considera como parte del conjunto de terminales bloques residuales predefinidos. Por otro lado, las funciones consideradas alteran el número de filtros, el tamaño de éstos, *stride* y la conectividad entre ellos. Para probar el desempeño del algoritmo se recurre a la conocida base de datos CIFAR-10.

3.2.3. Harris, et al (2018) [41]

Harris hace uso de PGC para diseñar una arquitectura de red neuronal. Entre las funciones consideradas resalta la convolución y el número de filtros considerado. Al igual que [67], utilizan la suma y concatenación como operadores aplicados al conjunto de terminales. La operación de pooling es de aridad 0 y se aplica utilizando un filtro de 2×2 indiscriminadamente. Si se exceden las operaciones de pooling, que puede provocar la anulación de las dimensiones de la imagen, cada operación de pooling es remplazada por alguna otra operación seleccionada aleatoriamente. Al igual que [84], para evaluar el desempeño de la red se acude a la base de datos CIFAR-10.

3.2.4. McGhie et al (2020) [48]

McGhie utiliza PG basada en árboles para evolucionar arquitecturas de redes neuronales convolucionales, método que denomina GPCNN a (*Genetic Programming - Convolutional Neural Networks*). Toma inspiración de distintos tipos de arquitecturas para generar funciones que alteran la conectividad, como lo es ResNet, DenseNet, GoogleNet [71]. Además, se propone un nuevo mecanismo de cruce, denominado *cruza parcial de su árbol*. Es importante señalar que en este trabajo no se incorporan funciones aritméticas. Además, el conjunto de datos que se considera para probar el desempeño de las redes es CIFAR-10, al igual que en las referencias anteriores.

3.2.5. Comentarios

Como puede observarse, la mayor parte de la literatura revisada se enfoca en aplicar PG para diseñar redes neuronales que realicen tareas específicas de clasificación de imágenes. De estos estudios trasciende el uso de PG y las funciones u operaciones consideradas. Los operadores aritméticos fungen como constructores de características en el diseño de algunos de los métodos descritos. Por otra parte, operaciones que alteran la conectividad de capas de convolución basándose en las arquitecturas Res-Net y Dense-Net también han sido consideradas por PG. La operación de pooling debe aplicarse restringidamente para no anular las dimensiones de la imagen de entrada. A continuación, se

realiza una revisión de trabajos que han utilizado el enfoque de Neuroevolución para tareas de segmentación de imágenes médicas.

3.3. Segmentación de imágenes médicas utilizando neuroevolución

Para la tarea de segmentación de imágenes médicas, el uso de arquitecturas del tipo *encoder-decoder* como base ha sido ampliamente utilizada [72] En la tabla 3.1 se resumen los hiperparámetros considerados, funciones de costo y tipos de imágenes que se han utilizado en los trabajos que se describen a continuación, que incluyen imágenes de resonancia magnética (MRI), ultrasonido (US) y tomografía computarizada (CT). Es destacable que para el entrenamiento de cada una de las redes generadas se utiliza el índice de Dice (ver ecuación 2.3) como una función de costo, expresado como $1 - DSC$, donde DSC representa esta métrica de desempeño. Los hiperparámetros utilizados en los estudios que se mencionan a continuación son de interés para el diseño del conjunto de funciones y terminales considerado por PG.

TABLA 3.1: Hiperparámetros, función de costo y tipos de imágenes considerados en diferentes algoritmos basados en Neuroevolución para segmentación de imágenes

Publicación	Hiperparámetros	Función de costo	Imágenes
[53]	Número de filtros, tamaño del filtro, pooling	-	MRI
[81]	Pooling, convolución	Dice	MRI, CT US
[2]	Tasa de aprendizaje, probabilidad disolución o <i>dropout</i> , función de activación, número de filtros, tamaño de filtros	Dice	MRI
[8]	Número de filtros, tamaño de los filtro, número de células	Dice	MRI
[26] [25]	Función de activación, número de filtros tamaño del filtro, pooling, conexiones atajo, inicialización de pesos, tamaño de datos aumentados.	Dice	MRI

3.3.1. Mortazi y Bagei (2018) [53]

Bajo un enfoque de aprendizaje por refuerzo, Mortazi y Bagei utilizan el algoritmo *Policy Gradient* donde el índice Dice toma la función de recompensa. En este estudio destaca el uso de una arquitectura *encoder-decoder* para segmentar áreas de interés del corazón en imágenes MRI. Además, utilizan bloques que contienen conexiones residuales, densas y operaciones de concatenación. Se reemplaza la función de activación ReLU por la función *Swish* [57].

3.3.2. Weng, et al (2019) [81]

Weng propone el método NAS-Unet. Es el primer trabajo del que se tiene conocimiento en utilizar U-Net como estructura base. La evolución se realiza modularmente con células que posteriormente son colocadas sobre esta estructura siguiendo algunas reglas, como duplicar el número de canales por cada nivel de profundidad. Cada célula

se representa con un grafo acíclico dirigido, donde cada nodo representa una imagen o mapa de características, mientras que las aristas representan el flujo de las operaciones. Además, hacen uso de la convolución separable en profundidad y convolución dilatada para reducir los parámetros entrenables. Segmentan el órgano próstata, hígado y nervios en imágenes de MRI, CT y US, respectivamente. Para buscar la arquitectura de la red, se recurre a la *estrategia de búsqueda de arquitectura diferencial (DARTS)* [45], caracterizado por utilizar un espacio de búsqueda continuo, donde la arquitectura U-Net es optimizada con respecto a su desempeño en el conjunto de validación mediante el descenso del gradiente.

3.3.3. Baldeon Calisto y Lai Yuen (2020,2021 [2, 8])

Este trabajo es uno de los referentes en neuroevolución para tareas de segmentación. Baldeon y Lai proponen el método AdaresU-Net [2]. Utilizan un enfoque multiobjetivo que considera el índice de Dice y la complejidad del modelo. Se basa en una codificación basado en células compuesta por 3 capas de convolución y conexiones residuales siguiendo una representación de vector de número reales. En su trabajo más reciente [8], se propone el método SaMEA, se utiliza un grafo acíclico dirigido con un número fijo de nodos bajo una codificación basada en células y un sistema subrogado para aminorar el costo computacional del proceso evolutivo. Cada nodo de la célula representa una operación, mientras que las aristas señalan el flujo de las operaciones.

3.3.4. Hassanzadeh, et al (2020, 2021) [25, 26]

Hassanzadeh presenta el método EvoU-Net. En este trabajo se utiliza una codificación basada en bloques utilizando una representación de cadena de números reales manipulado por un algoritmo genético. Como función de evaluación se utiliza la métrica Dice. Para acelerar el proceso de entrenamiento, cada red es entrenada durante 5 épocas y, si el modelo no mejora el desempeño durante 3 veces seguidas, el entrenamiento se detiene. En su trabajo más reciente [25], se incluyen 6 modelos de atención que consisten en bloques definidos y que funcionan como una alternativa a las conexiones puente consideradas por U-Net.

3.3.5. Comentarios

La arquitectura U-Net ha sido utilizada ampliamente para problemas de segmentación y diseño automático de topología de redes neuronales, como se muestra en la sección 3.3, donde distintos tipos de paradigmas evolutivos han sido aplicados como algoritmos genéticos (AG) y DARTS [45]. Los estudios revisados muestran que las técnicas de neuroevolución basadas en PG se han limitado a resolver problemas de clasificación de imágenes mas no de segmentación.

De acuerdo a la revisión de la literatura, la segmentación de imágenes de US no ha sido suficientemente explorada. Únicamente [81] utiliza este dominio para segmentar nervios bajo un enfoque de neuroevolución. Por otro lado, [24] utiliza la arquitectura U-Net

sustituyendo cada operación de convolución con convoluciones separables en profundidad o *dept-wise separable convolution* para segmentar la placenta en imágenes MRI, disminuyendo así la complejidad de U-Net. El estudio realizado por Ricky Hu [34, 36] también utiliza U-Net, que se caracteriza por el preprocesamiento aplicado y es el trabajo con quien se comparte el dominio de imágenes (imágenes bidimensionales de ultrasonido) y el órgano de interés (placenta). Es importante destacar que ninguno de los métodos de segmentación de placentas revisados utilizan neuroevolución.

En este trabajo se propone un algoritmo de neuroevolución basado en PG para segmentar placentas en imágenes de US que busca obtener resultados competitivos tanto en desempeño como en complejidad con respecto a U-Net.

Capítulo 4

Propuesta

Como se mencionó en la sección 3.1, el estudio de la placenta permite identificar y prevenir patologías que atenten contra el feto y/o la madre. Las imágenes de ultrasonido son difíciles de interpretar si el médico no cuenta con un buen entrenamiento. El uso de redes neuronales convolucionales como U-Net han logrado segmentar regiones de interés en imágenes médicas, particularmente en [36] se ha empleado en el dominio de nuestro interés, imágenes de ultrasonido 2D. Motivados por este tipo de arquitectura y utilizando PG, en esta sección se describe una propuesta para diseñar una arquitectura de red neuronal convolucional que sea exclusiva para la tarea de segmentación con un nivel competitivo al alcanzado por U-Net.

4.1. Algoritmo

El algoritmo propuesto se denomina NASGP-Net (Neural Architecture Search / Genetic Programming / U-Net). Utiliza una codificación basada en células que se comparte por la red entera sobre una estructura U-Net, es decir, la forma simétrica, las conexiones puente y el uso de convoluciones transpuestas características de esta arquitectura se mantienen.

En la figura 3.1 se pueden identificar cuatro niveles en la sección de contracción y en la sección de expansión con un elemento denominado ‘cuello de botella’ entre ambos. Por cada nivel en la sección de contracción se tienen dos capas de convolución y una capa de max-pooling. Nótese que por cada nivel subsecuente se duplica el número de filtros de cada operación de convolución. Por otro lado, en la sección de expansión se aplica en forma de espejo la sección de contracción reemplazando cada operación de max-pooling con una convolución transpuesta excepto en el último nivel, donde se aplica una operación de convolución a puntos o convolución *pointwise* con dos canales de salida donde se obtiene la máscara predicha. Con base en esta idea es posible representar cada nivel como un árbol. Asumiendo que el conjunto de funciones está compuesto por la operación de convolución (*conv*), convolución transpuesta (*convT*) y max-pooling (*mpool*). Así mismo, el conjunto de terminales se compone del número de filtros, tamaño del kernel de convolución y tamaño de kernel de pooling, la figura 4.1 representa cada una de las células que conforma U-Net.

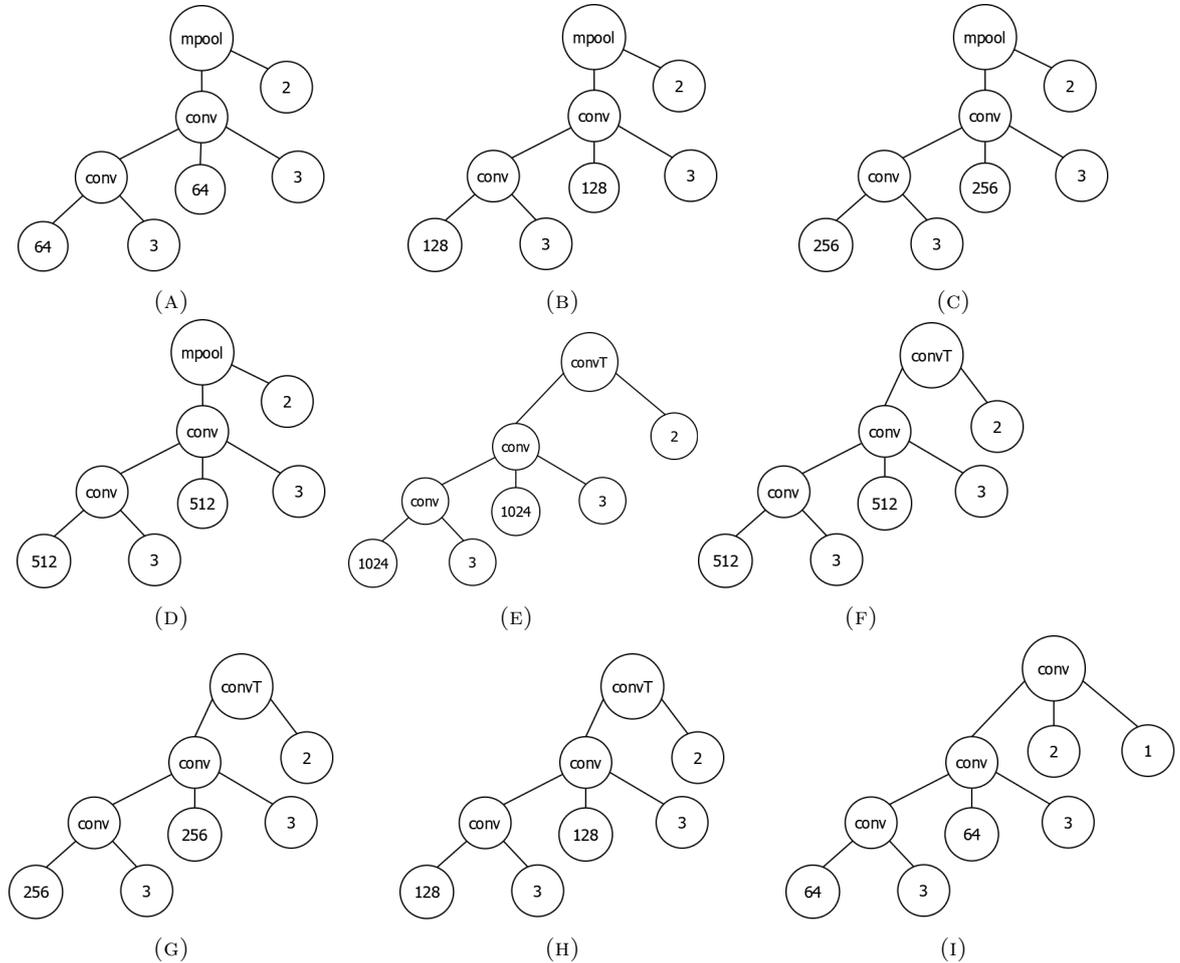


FIGURA 4.1: Células de U-Net representadas como árboles. Se asume que el tamaño de la máscara aplicado a la convolución es cuadrada. (A)-(D) corresponden a las células de la etapa de contracción. (E) representa el cuello de botella, punto en el cual comienza la etapa de expansión. (F)-(I) corresponden a las células de la etapa de expansión

Basándose en esta idea es posible representar cada nivel con un árbol donde la raíz es una operación de pooling y el resto contiene operaciones de convolución y funciones declaradas. Dado que se utiliza una codificación basada en células, la red convolucional se puede inferir a partir de la primera célula aplicando la heurística de duplicar el número de canales de cada capa operación de convolución en los niveles subsecuentes en la etapa de contracción y sustituyendo la operación de pooling por convolución transpuesta en la etapa de expansión, donde el número de canales se reduce a la mitad y las dimensiones espaciales se duplican, a excepción de la última célula, donde la operación pooling es remplazada por una operación de convolución *point-wise* con dos canales de salida. En la figura 4.2 se ilustra esta idea.

El algoritmo 1 muestra el proceso general del método NASGP-Net. Primeramente se genera una población de árboles con base en el conjunto de funciones y terminales considerado. Posteriormente, en la etapa de evaluación, se construye el modelo correspondiente a cada árbol que representa la primera célula y se obtiene una aptitud considerando el índice de Dice y la complejidad del modelo. Enseguida, se actualiza la población utilizando los operadores de cruce y mutación. A continuación, se evalúa la nueva población. El proceso continúa iterativamente hasta llegar a un máximo de generaciones definido por el usuario. Finalmente, NASGP-Net devuelve el mejor individuo de la población, es decir, aquella célula con la que se construye el modelo de segmentación con la mejor aptitud.

Algoritmo 1: NASGP-Net

Entrada: Conjunto de imágenes de entrenamiento, validación y prueba

Salida: Mejor individuo de la población

Inicialización de población;

Evaluación de la población;

mientras $Generaciones < Max\ Generaciones$ **hacer**

 Selección de padres con base su aptitud;

 Actualización de la población utilizando los operadores de variación;

 Evaluación de los nuevos individuos;

 Selección de individuos sobrevivientes para la siguiente generación;

fin

Devuelve el mejor individuo de la población

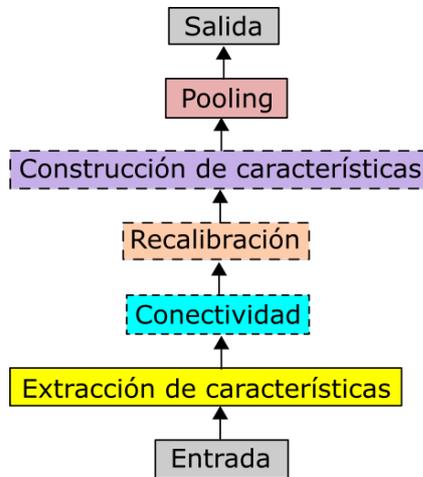
4.1.1. Estructura de operaciones

Inspirados por [5] y gracias a las propiedades de STGP [52], las operaciones o funciones consideradas por *NASGP-Net* se rigen bajo una estructura jerárquica por capas (ver figura 4.3a) que incluye operaciones de extracción de características, de conectividad de capas de convolución, recalibración y construcción características. En la raíz de cada árbol se encuentra la operación de pooling. Las operaciones de conectividad, recalibración y construcción de características son opcionales y pueden o no ser aplicadas en cada célula.

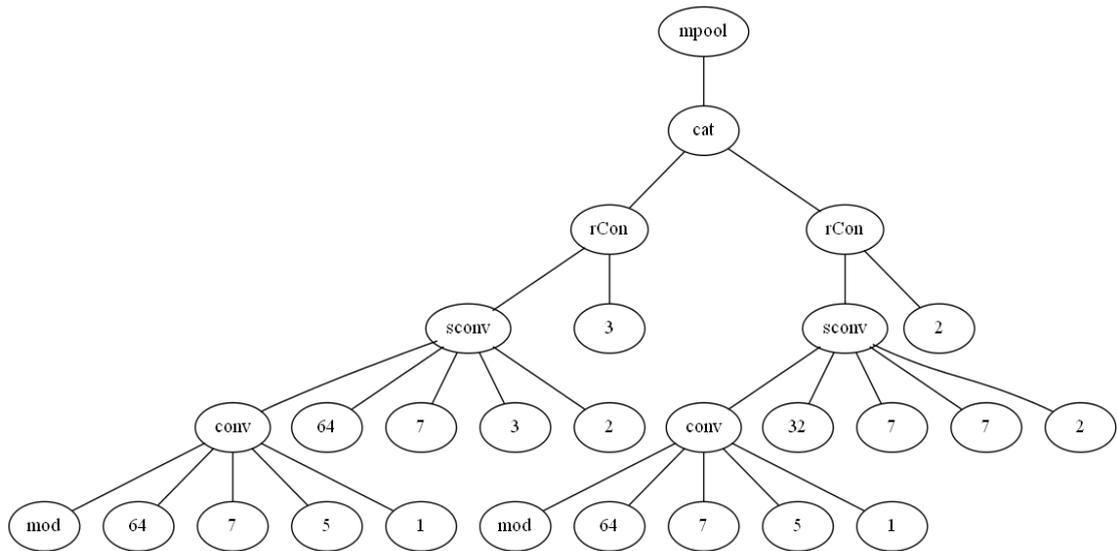
4.1.2. Conjunto de funciones

En la tabla 4.1 se describe el conjunto de funciones empleados, el tipo de dato entrada y de salida de cada función y la capa de operación correspondiente de acuerdo con la figura 4.3a.

Como parte de la extracción de características, se considera la operación de convolución (*conv*) y convolución separable en profundidad (*sconv*), que reduce considerablemente el número de parámetros del modelo como se menciona en la sección 2.2.3, al aplicar convoluciones *depth-wise* y *point-wise*. El tamaño de los filtros de convolución aplicados no son necesariamente cuadrados (se considera como una 2-tupla). Por cada



(A) Jerarquía de operaciones utilizada por NASGP-Net. Los bloques marcados con líneas punteadas corresponden a capas de operaciones opcionales



(B) Ejemplo de árbol generado por NASGP-Net. Nótese que la capa de recalibración se encuentra desactivada al no contener alguna función *se*

FIGURA 4.3: Estructura de operaciones en NASGP-Net

capa de convolución empleada por estas funciones se agrega una *capa de normalización* de grupo o *GN* (ver sección 2.2.5) seguido de una *capa de activación ReLU*. Se consideran también operaciones que alteran la conectividad de una secuencia de capas convolucionales, permitiendo realizar conexiones del tipo residual o denso con las operaciones *rCon* y *dCon*, respectivamente, ver secciones 2.2.6 y 2.2.7. Dado que la operación *squeeze and excitation* (*se*) requiere un incremento en los parámetros entrenables del modelo, se limita a aplicarse una sola vez por cada secuencia de capas de convolución, bloque denso

TABLA 4.1: Conjunto de funciones

Capa	Función	Entrada	Salida	Descripción
Pooling	mpool	Módulo	Módulo	Crea un módulo de max-pooling con un tamaño de kernel = 2 y stride = 2
	apool	Módulo	Módulo	Crea un módulo average-pooling con un tamaño de kernel = 2 y stride = 2
Construcción de características	add	Módulo 1, Módulo 2, ω_1, ω_2	Módulo	Realiza una suma ponderada entre la salida de dos módulos con tamaños distintos. Agrega canales de ceros de ser necesario
	sub	Módulo 1, Módulo 2, ω_1, ω_2	Módulo	Realiza una resta ponderada entre la salida de dos módulos con tamaños distintos. Agrega canales de ceros de ser necesario
	cat	Módulo 1, Módulo 2	Módulo	Concatena la salida de dos módulos en la dimensión de canales
Recalibración	se	Módulo	Módulo	Aplica la operación <i>squeeze-and-excitation</i> con un factor de reducción $r = 8$
Conexión	dCon	Módulo, θ	Módulo	Reconecta una secuencia de capas de convolución con conexiones densas. Los canales de salida se comprimen con un factor θ
	rCon	Módulo	Módulo	Reconecta una secuencia de capas de convolución con conexiones residuales. La conectividad se realiza cada dos capas de convolución si el número de capas de convolución en la secuencia es par.
Extracción de características	conv	Módulo, no. de filtros, tamaño de kernel (2-tupla), tasa de dilatación.	Módulo	Crea un módulo secuencial convolución-GN-ReLU. El tamaño del kernel puede ser rectangular. Utiliza <i>same padding</i> para preservar las dimensiones espaciales.
	sconv	Módulo, no. de filtros, tamaño de kernel (2-tupla), tasa de dilatación	Módulo	Crea un módulo secuencial de convolución separable en profundidad-GN-ReLU. El tamaño del kernel puede ser rectangular. Utiliza <i>same padding</i> para preservar las dimensiones espaciales.

o bloque residual. Para poder aplicar las operaciones aritméticas es necesario que las entradas tengan las mismas dimensiones. Si las dimensiones espaciales de los mapas de características de las funciones de suma y resta (*add* y *sub*) no son las mismas, se realiza un escalamiento a la segunda entrada de modo que coincida con las dimensiones de la primera entrada; si el número de canales no coincide, se añaden canales de ceros a aquella entrada que los requiera, similar a [68]. El tamaño del kernel y stride utilizado en las operaciones de pooling, *mpool* y *apool* se mantiene constante con un valor de 2.

En las figuras 4.4, 4.5 se ejemplifican gráficamente las operaciones de extracción de características, conexión y recalibración.

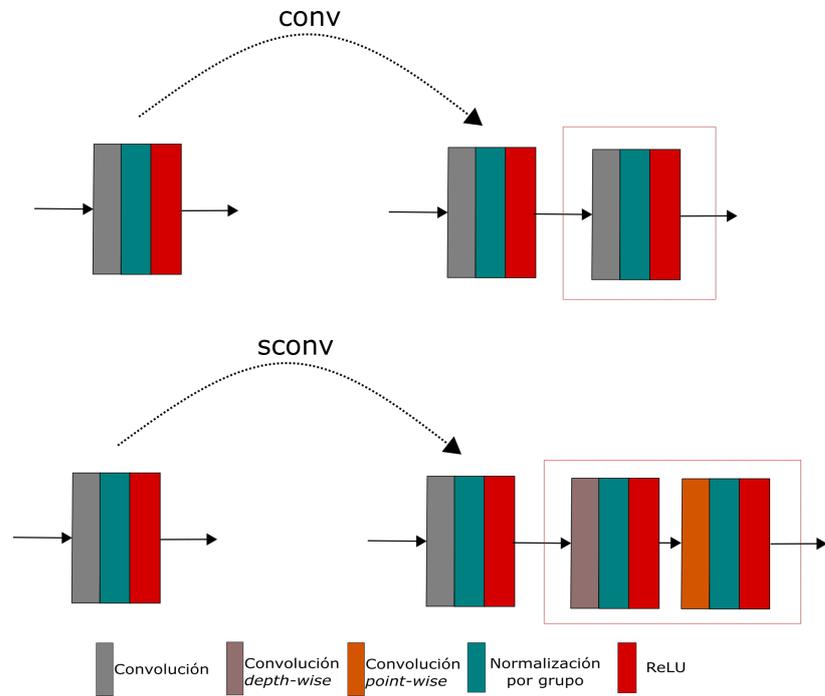


FIGURA 4.4: Operaciones de extracción de características. Las operaciones añaden capas de convolución al contenedor original que a su vez generan capas de normalización y activación. Las capas añadidas se enmarcan con un rectángulo rojo. En el ejemplo se asume que la lista de construcción (terminal *módulo*) contiene una capa de convolución, con su respectiva capa de normalización y activación, y muestra las modificaciones que sufre al aplicar la función *conv* o *sconv*

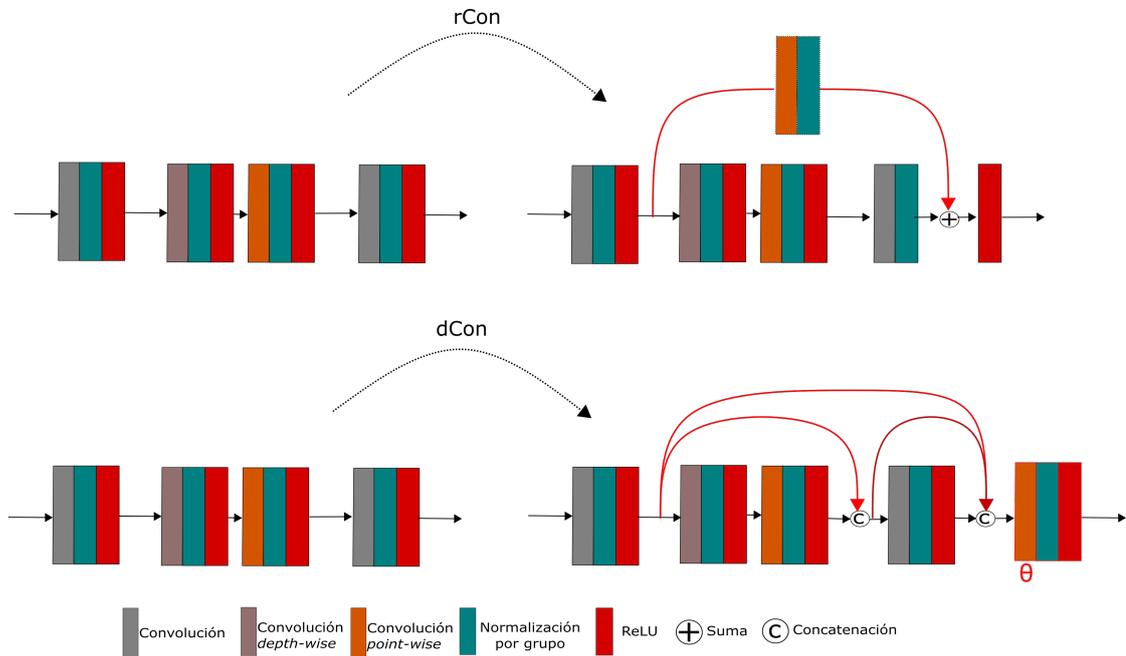


FIGURA 4.5: Operaciones de conectividad. Las operaciones crean conexiones sobre la secuencia de bloques de entrada, marcadas con flechas rojas. En el ejemplo se asume que la lista de construcción (terminal *módulo*) se compone de tres elementos definidos como una secuencia convolución-convolución separable en profundidad-convolución. La operación *rCon* crea una conexión cada dos capas de convolución. Nótese que la conexión se realiza desde la entrada a la capa de convolución *depth-wise* a la salida de la secuencia. Además, crea una capa de convolución *point-wise* que ajusta el número de canales de entrada para que la función de adición sea aplicada correctamente. La función *dCon* añade conexiones densas a la secuencia y crea una capa de convolución *point-wise* que comprime el número de canales con base en un factor θ .

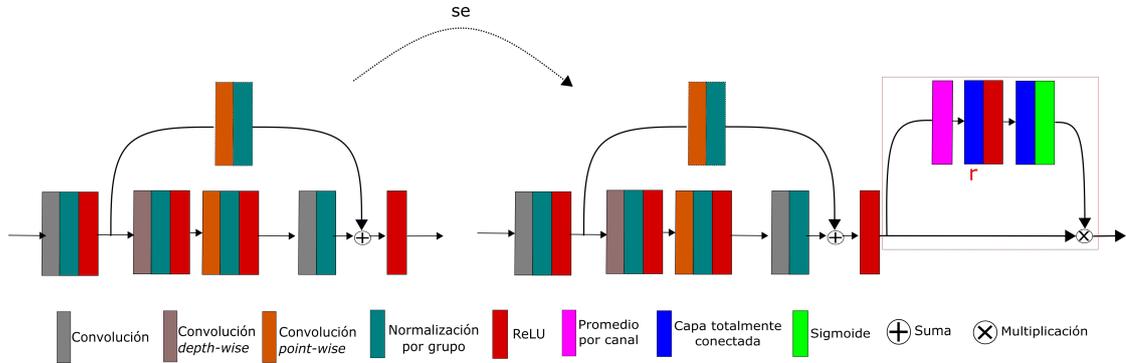


FIGURA 4.6: Operación de recalibración, *se*. Esta operación puede ser aplicada a una secuencia de capas de convolución, un bloque residual o un bloque denso. En el ejemplo la operación *se* es aplicada al bloque residual obtenido en la figura 4.5. El factor de reducción, r , toma un valor de 8, dado que es el mínimo número de canales de salida posible para la primera célula.

4.1.3. Conjunto de terminales

En la tabla 4.2 se muestra el conjunto de terminales considerado, los valores posibles que pueden tomar y una breve descripción de cada uno. La terminal *Módulo* es un contenedor que concatena las capas de convolución creadas en la capa de extracción de características. Dado que se utiliza una codificación basada en células, un paso importante previo a la evaluación del individuo es la *construcción* de la red convolucional (ver figura 4.2). Una vez que el modelo se crea a partir de la primera célula, las entradas pueden considerarse como tensores o mapas de características. Los parámetros w_1 y w_2 son números aleatorios que ponderan las funciones de *suma* y *resta* en la construcción de características (*add* y *sub*). El parámetro θ corresponde al factor de compresión utilizado en la función *dCon*, que reduce el número de canales de la salida del bloque denso aplicando una convolución tipo *point-wise*, ver subsección 2.2.7. Las terminales de las operaciones de extracción (*conv* y *sconv*) incluyen el número de filtros, tamaño del kernel y tasa de dilatación (ver figura 2.6). El número de filtros se ha limitado a tomar valores de hasta 32 para asegurar tener una menor cantidad de filtros por convolución en comparación a U-Net que toma 64 filtros. El tamaño de kernel se mantiene al sustituir la función de *pooling* por *convolución transpuesta* al construir la red convolucional (ver figura 4.2).

4.1.4. Parámetros y características de NASGP-Net

Distintos parámetros intervienen en la ejecución del método NASGP-Net: tamaño de población, número de generaciones, probabilidad para cada uno de los operadores de variación, parámetros relacionados con el mecanismo de selección, función de costo, número de épocas de entrenamiento para cada red, tasa de aprendizaje y peso de complejidad.

TABLA 4.2: Conjunto de terminales

Capa	Terminal	Rango/Tipo	Descripción
Construcción de características	ω_1, ω_2	[0,1,1.0]	Valores aleatorios. Son terminales hijos de la operación sub y add.
Conexión	θ	[0,3,0,8]	Tasa de compresión para la operación <i>dConv</i> .
Extracción de características	Número de filtros	{8,16,32}	Número de filtros aplicados a cada operación conv o sconv.
	Tamaño de kernel	{3,5,7}	Tamaño del kernel para referirse al alto y ancho del kernel de convolución en las operaciones conv y sconv.
	Tasa de dilatación	{1,2}	Tasa de dilatación que agrega espacios entre los elementos del kernel de convolución. Para una tasa de dilatación = 1 es una operación regular.
Entrada	Modulo	Lista	Contenedor que almacena y concatena las capas de convolución durante la etapa de extracción de características.

La elección del máximo número de parámetros está basado en la cantidad de parámetros que componen U-Net. Como se menciona en la sección 2.3.2, la suma de la probabilidad de cruce y de mutación debe ser menor o igual que 1. En este enfoque se ha remplazado la operación de *reproducción* por la *mutación de terminales*, cuya probabilidad de aplicación corresponde a $1 - (p_c + p_m)$, donde p_c y p_m representan la probabilidad de cruce y mutación, respectivamente.

Debido al enfoque poblacional con múltiples generaciones que consideran PG, cada red neuronal convolucional se entrena durante un número bajo de épocas (entre 6 y 10 épocas), pues ello estima el desempeño del modelo al entrenarse durante más épocas con un costo computacional razonable [48, 70]. Una vez que NASGP-Net devuelve el mejor individuo, la red construida a partir de éste se entrena por 100 épocas. El tamaño del lote se establece con un valor de 1, tal como se maneja en [60]. Además, cada modelo utiliza como función de costo *combo loss* [73], denotado como \mathcal{L} , que aprovecha la flexibilidad del *índice de Dice* para clases desbalanceadas y la suavidad de la curva de *entropía cruzada* [72]. Para N muestras y K clases, la función de costo *combo* se expresa como:

$$\mathcal{L} = \alpha \left(-\frac{1}{N} \sum_{i=1}^N \beta (t_i - \ln p_i) + (1 - \beta) [(1 - t_i) \ln (1 - p_i)] \right) - (1 - \alpha) \sum_{i=1}^K \left(\frac{2 \sum_{i=1}^N p_i t_i + s}{\sum_{i=1}^N p_i + \sum_{i=1}^N t_i + s} \right) \quad (4.1)$$

donde t_i representa el valor del píxel de la máscara de referencia y p_i representa la probabilidad del píxel predicho al evaluarse en una función sigmoide. α corresponde a la contribución de cada función de costo, s es un término de suavizado para tener consistencia numérica. El parámetro $\beta \in [0, 1]$ controla el nivel de penalización de los

falsos positivos y negativos. Un valor de $\beta < 0.5$ penaliza más los falsos positivos mientras que $\beta > 0.5$, penaliza más los falsos negativos. En las imágenes de ultrasonido donde los bordes de la placenta no son muy claros, algunas zonas de la placenta podrían no etiquetarse correctamente, por lo que se requiere penalizar más a los falsos negativos (píxeles de la región de interés etiquetados como parte del fondo) [73]. Los valores de α y β se han establecido con un valor de 0.5 y 0.7, respectivamente, de acuerdo a lo reportado en [73] para imágenes de ultrasonido.

El algoritmo *NASGP-Net* cuenta con ciertas características inmutables para el usuario, relacionadas al proceso de PG y entrenamiento de las redes, ver tabla 4.3.

TABLA 4.3: Características de NASGP-Net

Característica	Valor	Característica	Valor
Generación de población	Ramped half-and-half	Tipo de mutación	Mutación de subárbol, de contracción, de remplazo y de terminales
Profundidad de población inicial	[2,6]	Tasa de aprendizaje	0.0001
Profundidad máxima descendencia	10	Optimizador	ADAM
Tipo de Cruza	Cruza de subárbol		

La generación de la población se realiza mediante el método *ramped half-and-half* (ver subsección 2.3.2) que establece una cota mínima y máxima de profundidad a cada individuo de la población inicial.

La *cruza de un punto o cruza uniforme* es el operador de variación seleccionado (ver sección 2.3.2). Se han seleccionado cuatro tipos de mutación: *mutación de subárbol*, *mutación de contracción* o *shrink mutation* (ver subsección 2.3.2), *mutación de terminales*, que altera todas las hojas del árbol, y *mutación de remplazo*, que sustituye una función con cualquier otra de la misma aridad. Es importante destacar que, a excepción de la *mutación de subárbol*, los operadores de mutación seleccionados no aumentan la profundidad del árbol, aminorando así el efecto *bloat*.

Otro mecanismo utilizado para controlar el efecto *bloat* es establecer una *cota superior de altura* de modo que, si la descendencia presenta una altura mayor a la cota establecida, uno de los padres es seleccionado aleatoriamente en su lugar.

El algoritmo incluye un *mecanismo de elitismo*, que preserva la mejor solución a lo largo de las generaciones. La *función de aptitud* es una combinación lineal que considera el desempeño de segmentación en términos del índice de Dice y la complejidad del modelo al cuantificar la cantidad de parámetros, como se expresa en la ecuación 4.2. Cabe señalar que la máxima cantidad de parámetros y el peso de la complejidad del modelo, w , forman parte de los parámetros de NASGP-Net.

$$\text{aptitud} = (1 - w) * \text{Dice} + w * \frac{MP - NP}{MP} \quad (4.2)$$

donde MP se refiere al máximo número de parámetros permitido y NP es el número de parámetros del modelo.

Capítulo 5

Experimentos y resultados

La experimentación se divide en dos fases. En la fase 1 se exponen los cambios preliminares antes de consolidar el algoritmo propuesto. Se muestran los resultados de una ejecución de NASGP-Net y dos experimentos previos utilizando el conjunto de imágenes de placentas. Los cambios se realizan en parámetros relacionados a los operadores de variación, conjunto de funciones, función de aptitud y profundidad de los árboles. En la segunda fase de experimentación se realizan pruebas estadísticas considerando, además de la base de datos de placentas, la base de datos *Retinal Vessel Segmentation* [58].

La base de datos de placentas se conformó paulatinamente durante el desarrollo del proyecto. El consentimiento informado se realiza de acuerdo al comité de investigación del Instituto de Investigaciones Biológicas de la Universidad Veracruzana (ver B). Se compone de 100 imágenes de ultrasonido 2D correspondiente a 28 pacientes embarazadas adquiridas durante el periodo del 4 de marzo de 2021 al 17 de junio del 2021. La edad de las pacientes se encuentra entre 18 a 42 años. Las imágenes son adquiridas utilizando una máquina de ultrasonido Accuvix 2016. La resolución de las imágenes es de 768×1024 píxeles y redimensionadas a 288×480 para efectos de experimentación. La máscara de segmentación de referencia o *groundtruth* fue realizada manualmente por el experto en cada una de las imágenes mediante una interfaz implementada en MATLAB.

La base de datos para la segmentación de vasos de la retina se compone de 20 imágenes con sus respectivas máscaras de segmentación o *groundtruth*. La resolución de las imágenes originalmente es de 565×584 y son redimensionadas a 256×256 para efectos de experimentación. En ambos conjuntos se considera el 70% para entrenamiento, 15% para validación y el resto para prueba. Éste último subconjunto es con el cual el desempeño del modelo de segmentación es evaluado.

Para un mejor aprendizaje de las redes generadas se aplica la técnica de *aumento de datos* o *data augmentation*. Se consideran giros horizontales en la imagen, transformaciones elásticas [61] con un coeficiente de elasticidad, $\sigma = 10$ y distorsiones de malla de tamaño 5×5 . Los pesos iniciales de cada capa de convolución siguen una distribución Gaussiana cuya desviación estándar depende del número de filtros, N , y su tamaño, H, W . Se expresa mediante la siguiente expresión: $\sqrt{2/NHW}$. Capas de dilución (o *dropout*) se añaden al final de la etapa de contracción del modelo para evitar sobreajuste.

Para la implementación de las redes neuronales se utiliza la librería *Pytorch* versión 1.5.0 [56]. El entrenamiento de las redes neuronales se realiza utilizando 4 tarjetas GPU Volta 100 en el clúster de supercómputo de la Benemérita Universidad Autónoma de Puebla. El proceso evolutivo fue implementado utilizando la librería DEAP (*Distributed Evolutionary Algorithm Package*) [16] en su versión 1.0.2.

Dado el costo computacional del algoritmo, se han implementado tres estrategias para mayor rapidez y seguridad durante su ejecución:

1. *Checkpoint*. Por cada evaluación se guarda la población y la aptitud de los individuos ya evaluados de modo que, ante cualquier evento externo que detenga la ejecución, sea posible restaurar la ejecución desde el punto donde se quedó.
2. *Multiprocesamiento*. Para aprovechar las cuatro tarjetas gráficas disponibles en el servidor, los individuos son evaluados asincrónicamente en grupos de 4 asignando una tarjeta a cada uno de ellos.
3. *Memoria Caché*. Inspirados por [69] y con el fin de evitar evaluar individuos que ya han sido evaluados a lo largo de las generaciones, se utiliza una memoria caché haciendo uso de una función *hash-224* [31].

A continuación se detalla cada una de las fases de experimentación.

5.1. Primera Fase

Durante la primera fase de experimentos se determinaron parámetros correspondientes a la probabilidad de los operadores de variación, conjunto de funciones y terminales, profundidad de los árboles y la función de aptitud. Los cambios graduales son reportados hasta establecer el algoritmo NASGP-Net de modo que el desempeño en términos del índice de Dice alcance, al menos, un nivel competitivo o cercano al obtenido por U-Net (0.934), y que a su vez reduzca el número de parámetros entrenables (31M).

En la tabla 5.1 se muestra el penúltimo y último experimento antes de conformar el algoritmo NASGP-Net, señalados como experimento 1 y experimento 2, respectivamente. La principal diferencia del experimento 2 con respecto al experimento 1 radica en las probabilidades de cruce y mutación. Se aumentó la probabilidad de mutación de modo que los distintos tipos de mutación utilizados tengan un mayor efecto durante el proceso evolutivo. El experimento 2 fue el primero en utilizar la jerarquía de operaciones como en la figura 4.3a con la excepción de no considerar la operación *se* o *squeeze-and-excitation*, por lo que la capa de recalibración no es utilizada. El número de filtros máximo considerado se ha reducido a 32, en comparación a los 64 considerados en el experimento 1 y el tamaño máximo del kernel de convolución se reduce a 5. Finalmente, en NASGP-Net la función de aptitud, además del índice de Dice incluye la complejidad del modelo en términos del número de parámetros (ver 4.2). Por lo anterior el máximo tamaño de los filtros de convolución vuelve a tomar un valor de 7.

TABLA 5.1: Parámetros para los experimentos considerados en la fase 1 de experimentación

Param	Exp. 1	Exp. 2	NASGP-Net
Población	100	100	100
Cruza	Cruza de un punto (0.9)	Cruza de un punto (0.8)	Cruza de un punto (0.8)
Mutación	Efímera, shrink, remplazo (0.1)	Efímera, shrink, remplazo, uniforme (0.19)	Efímera, shrink, remplazo, uniforme(0.19)
Selección	Torneo (3)	Torneo (3)	Torneo (3)
Generaciones	25	25	25
Profundidad	[2,8]	[2,8]	[2,10]
Función de costo	Combo Loss	Combo Loss	Combo Loss
Función de aptitud	Dice	Dice	Dice y no. param ($w = 0.3$)
Dataset	100 (70%-15%-15%)	100 (70%-15%-15%)	100 (70%-15%-15%)
Conjunto de funciones	conv, sconv, dCon, rCon, add, sub, cat, mpool, apool	conv, sconv, dCon, rCon, add, sub, cat, mpool, apool	conv, sconv, dCon, rCon, se, add, sub, cat, mpool, apool
Conjunto de terminales	Módulo, No. filtros (64), tamaño de filtro, tasa de dilatación, $\theta, \omega_1, \omega_2$	Módulo, No. filtros, tamaño de filtro, tasa de dilatación, $\theta, \omega_1, \omega_2$	Módulo, No. filtros, tamaño de filtro, tasa de dilatación, $\theta, \omega_1, \omega_2$

En la tabla 5.2 se muestran los resultados obtenidos durante esta fase de experimentación y su comparación con el modelo U-Net. Nótese cómo es que el número de parámetros incrementa considerablemente al utilizar 64 filtros como cota máxima en el experimento 1, cuatriplicando el número de parámetros de U-Net. Durante el experimento 2 se limitó el número de filtros a 32 y con un máximo tamaño de 5, obteniendo resultados favorables. Como se puede observar, se alcanza prácticamente el mismo índice de Dice que U-Net y se reducen los parámetros a la mitad. Al considerar la complejidad del modelo en la función de aptitud en NASGP-Net, fue posible disminuir aún más el número de parámetros en casi un 93 %. Sin embargo, esto trajo consigo un ligero decremento en el índice de Dice.

En la figura 5.1 se muestran las gráficas de convergencia para estos tres experimentos y el comportamiento de la profundidad y tamaño (número de nodos) promedio de la población a lo largo de las generaciones.

Con los resultados de la fase 1 de experimentos fue posible establecer los parámetros y características del algoritmo NASGP-Net para la segmentación de placentas. Es importante mencionar que la elección del valor w en la función de aptitud (ver ecuación 4.2) no es una tarea trivial dado el costo computacional del algoritmo. Además, dicho peso afecta de forma directa el desempeño de segmentación. El valor $w = 0.3$ permite una reducción considerable en el número de parámetros del modelo con una diferencia de desempeño en el índice de Dice de 0.009 en comparación con el mejor resultado (experimento 1) y de 0.004 en comparación con U-Net, por lo que podemos afirmar que se tiene un desempeño de segmentación competitivo con tan solo el 7 % de parámetros considerados por U-Net.

TABLA 5.2: Resultados obtenidos en la fase 1 de experimentación. Se muestran en negritas los mejores resultados con base en las métricas de índice de Dice (Dice), Intersección sobre Unión (IoU) y *Hausdorff Distance* (*HD*), ver sección 2.1.1

Estadística	U-Net	Exp.1	Exp.2	NASGP-Net
Dice Prom. \pm Std	0.934 \pm 0.047	0.937 \pm 0.045	0.934 \pm 0.050	0.928 \pm 0.052
Dice Min	0.793	0.796	0.784	0.773
Dice Max	0.967	0.975	0.984	0.977
IoU Prom \pm Std	0.879 \pm 0.076	0.885 \pm 0.074	0.881 \pm 0.081	0.869 \pm 0.085
IoU Min	0.657	0.662	0.645	0.631
IoU Max	0.941	0.951	0.969	0.956
HD Prom. \pm Std	8.398 \pm 1.570	8.389 \pm 1.720	8.589 \pm 1.835	8.813 \pm 2.097
HD Min	6.00	5.290	5.196	5.099
HD Max	11.314	11.136	11.790	13.928
No. Parámetros	31,037,698	129,435,202	15,943,475	2,165,483

5.2. Segunda fase

5.2.1. Segmentación de placentas

Durante la segunda fase se realizó una serie de 10 experimentos utilizando el algoritmo NASGP-Net y se comparó con el modelo U-Net. Además, se realizó una prueba estadística utilizando la prueba no paramétrica *U de Mann Whitney* con un nivel de confianza al 95% para establecer si existe o no una diferencia significativa en las métricas de Dice, IoU y *Hausdorff Distance*.

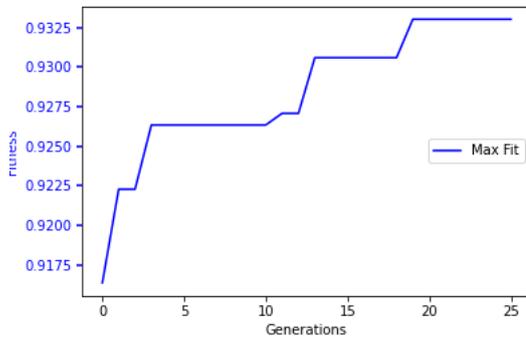
Los parámetros utilizados para las 10 ejecuciones se resumen en la tabla 5.3.

TABLA 5.3: Parámetros de NASGP-Net para segmentación de placentas

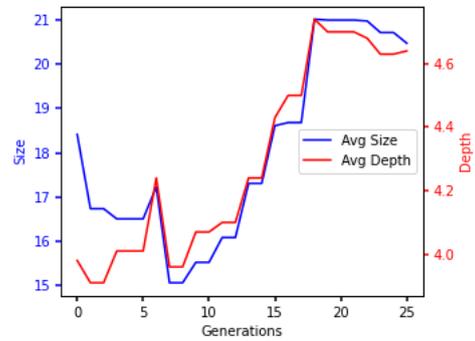
Parámetro	Valor	Parámetro	Valor
Tamaño de población	100	Peso de complejidad	0.3
Número de generaciones	25	Máximo no. de parámetros	31 038 000
Probabilidad de cruza	0.8	Función de costo	combo loss
Probabilidad de mutación	0.19	Número de épocas	6
Tamaño de torneo	3	Tasa de aprendizaje	0.0001

Cada una de las ejecuciones y métricas obtenidas se muestran en la tabla 5.4. Además, se muestra el árbol sintáctico correspondiente a la primer célula de la red neuronal convolucional neuroevolucionada. Las estadísticas de estas observaciones se muestran en la tabla 5.5

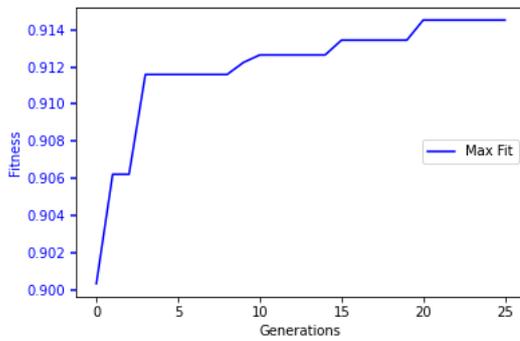
De acuerdo a las estadísticas obtenidas y el valor p , es evidente la superioridad de U-Net ante NASGP-Net en términos de las métricas basadas en solapamiento. Particularmente, el índice de Dice promedio alcanzado por U-Net es de 0.934 mientras que



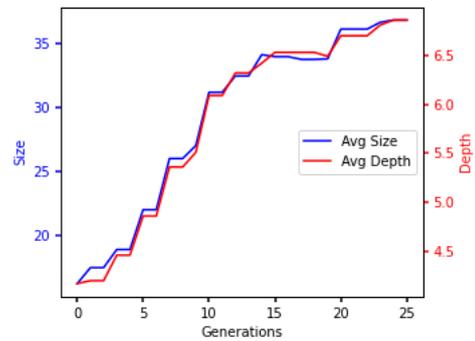
(A) Experimento 1. Gráfica de convergencia



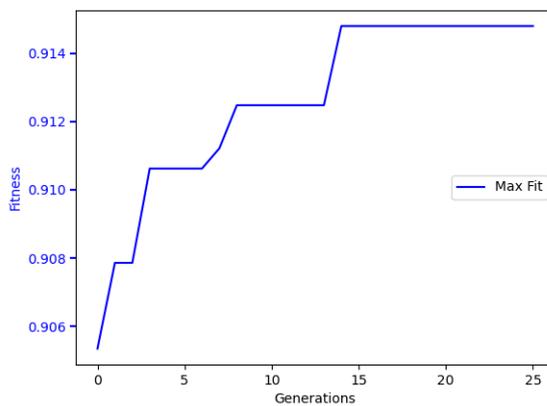
(B) Experimento 1. Profundidad y tamaño



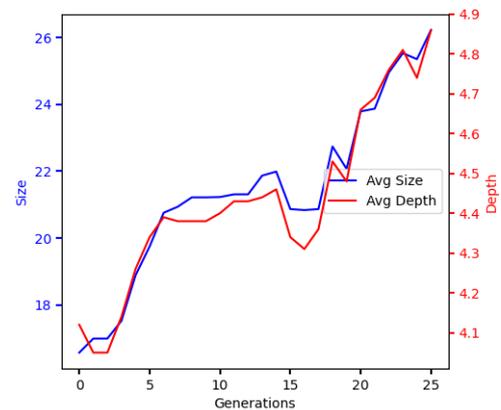
(C) Experimento 2. Gráfica de convergencia



(D) Experimento 2. Profundidad y tamaño



(E) NASGP-Net. Gráfica de convergencia



(F) NASGP-Net. Profundidad y tamaño

FIGURA 5.1: Fase 1 de experimentación. Gráficas de convergencia, tamaño y profundidad promedio

nuestro enfoque alcanza un índice de Dice promedio de 0.921. Sin embargo, podemos atribuir una mayor precisión a nuestro enfoque al comparar la desviación estándar obtenida.

La diferencia entre el promedio de las métricas basadas en solapamiento toman valores bajos, siendo U-Net superior a NASGP-Net con una diferencia de 0.012 en la métrica Dice y 0.018 en la métrica IoU. El valor p obtenido al comparar la métrica Hausdorff Distance nos muestra que la hipótesis nula se acepta, por lo que estadísticamente no hay diferencias significativas entre los métodos U-Net y NASGP-Net.

En cuanto a la complejidad del modelo, los resultados muestran una reducción entre el 92-98 % con respecto a los 31M de parámetros de U-Net.

En la figura 5.2 se muestra la gráfica de convergencia, la profundidad y el tamaño de la población a lo largo de las generaciones de la ejecución que corresponde a la mediana del índice de Dice (ejecución 1), de acuerdo con la tabla 5.5. El árbol que representa la primera célula de la red neuronal convolucional se muestra en la figura 5.3. Las tres imágenes mejor y peor segmentadas por NASGP-Net y U-Net se muestran en las figuras 5.4, 5.5, 5.6 y 5.7, respectivamente.

Es importante mencionar que además de las métricas de desempeño la opinión del experto es de suma importancia. Los resultados fueron mostrados al ginecólogo para evaluar el desempeño de segmentación. La mayoría de las imágenes se consideraron con una buena segmentación, con ciertos casos atípicos donde un desempeño menor se relacionó con zonas oscuras o cuya intensidad es similar a la región de referencia.

TABLA 5.4: 10 Ejecuciones de U-Net vs NASGP-Net, para la segmentación de placentas. Se marcan en negritas los mejores resultados por cada método del conjunto de ejecuciones

No.	U-Net			NASGP-Net					
	Dice Prom.	IoU Prom.	H.D. Prom.	No. Param	Árbol Sintáctico	Dice Prom.	IoU Prom.	H.D. Prom.	No. Param
1	0.934±0.047	0.879±0.076	8.399±1.570	31,037,698	$mpool(sub(rCon(sconi(mod, 16, 5, 5, 2)), 0.71, rCon(sconi(mod, 16, 5, 5, 2)), 0.81))$	0.920±0.048	0.856±0.077	9.007±1.754	567,636
2	0.934±0.057	0.880±0.090	8.601±1.536	31,037,698	$apool(rCon(sconi(mod, 32, 5, 5, 1)))$	0.929±0.054	0.871±0.087	8.558±1.890	1,438,227
3	0.931±0.046	0.875±0.075	8.869±1.677	31,037,698	$apool(rCon(sconi(mod, 16, 5, 5, 2)))$	0.920±0.055	0.856±0.087	8.752±1.843	370,995
4	0.934±0.047	0.880±0.076	8.671±1.600	31,037,698	$apool(rCon(sconi(mod, 16, 5, 5, 1)))$	0.917±0.054	0.852±0.084	8.665±1.764	370,995
5	0.930±0.047	0.873±0.077	8.801±2.023	31,037,698	$apool(rCon(sconi(mod, 32, 5, 3, 2)))$	0.920±0.055	0.855±0.088	8.710±1.826	1,423,797
6	0.940±0.048	0.891±0.078	8.353±1.757	31,037,698	$apool(sconi(mod, 32, 5, 5, 1))$	0.922±0.058	0.860±0.092	9.001±2.275	1,087,027
7	0.935±0.040	0.880±0.065	8.876±1.325	31,037,698	$apool(rCon(sconi(sconi(mod, 32, 5, 3, 2), 16, 7, 3, 1)))$	0.920±0.057	0.857±0.091	8.882±1.565	705,765
8	0.938±0.047	0.886±0.076	8.496±1.418	31,037,698	$apool(rCon(sconi(mod, 32, 5, 5, 1)))$	0.925±0.049	0.864±0.081	8.535±1.794	1,438,227
9	0.931±0.049	0.875±0.079	8.712±1.605	31,037,698	$apool(dCon(sconi(sconi(mod, 16, 3, 5, 1), 8, 3, 3, 1), 0.6))$	0.928±0.053	0.871±0.086	8.354±1.716	2,569,593
10	0.927±0.040	0.866±0.065	8.785±1.467	31,037,698	$sconi(sconi(sconi(mod, 16, 5, 3, 1), 32, 5, 3, 2), 16, 3, 3, 2), 0.68, rCon(sconi(mod, 16, 7, 7, 1), 0.65))$	0.923±0.052	0.869±0.085	8.813±2.097	2,165,483

 TABLA 5.5: Estadísticas obtenidas con base en las 10 ejecuciones. Se realiza la prueba estadística *U de Mann Whitney* tomando en cuenta las métricas de Dice, IoU y Hausdorff Distance. Se marcan en negritas los mejores resultados entre ambos métodos.

Estadística	Métricas							
	Dice			IoU			H.D.	
	U-Net	NASGP-Net	U-Net	NASGP-Net	U-Net	NASGP-Net	U-Net	NASGP-Net
Mejor	0.943	0.929	0.891	0.871	8.353	8.354	8.354	8.354
Peor	0.927	0.917	0.866	0.852	8.876	9.007	9.007	9.007
Media	0.934	0.922	0.879	0.861	8.656	8.727	8.727	8.727
Mediana	0.934	0.921	0.880	0.859	8.691	8.731	8.731	8.731
Desv. Std.	0.0046	0.0037	0.0066	0.0068	0.1792	0.1985	0.1985	0.1985
<i>p-value</i> (5%)	3.10E-04			4.25E-04			0.4727	

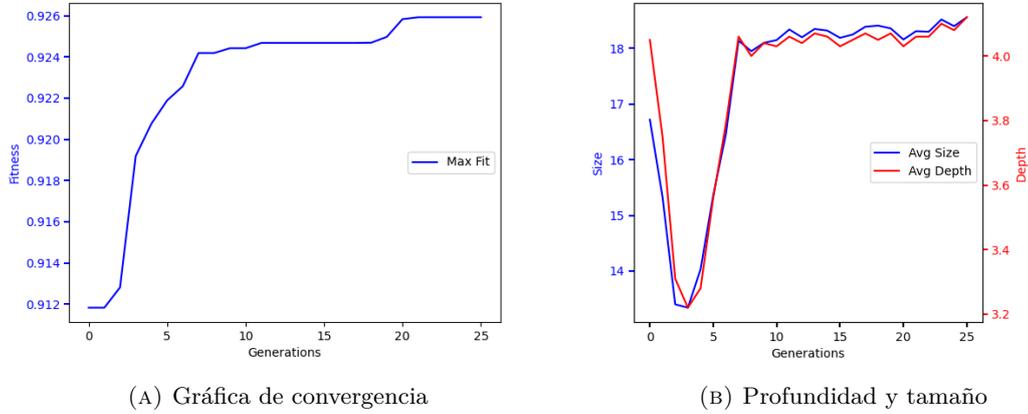


FIGURA 5.2: Gráfica de convergencia, profundidad y tamaño promedio de la ejecución correspondiente a la mediana de acuerdo con la tabla 5.5

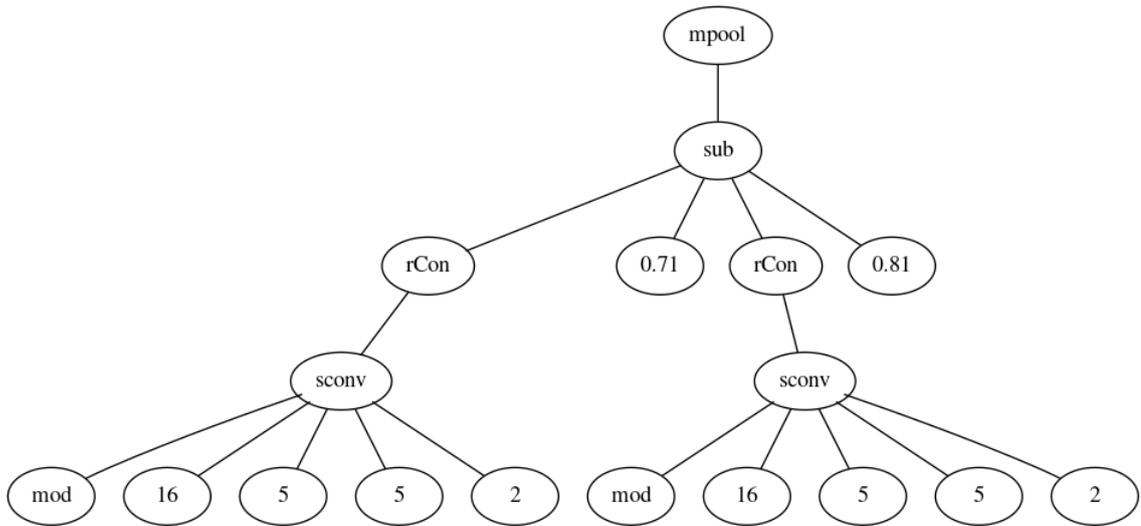


FIGURA 5.3: Árbol que representa la primera célula de la red convolucional correspondiente a la mediana de acuerdo con la tabla 5.5

5.2.2. Segmentación de vasos en la retina

NASGP-Net también fue utilizado con el conjunto de datos *Retinal Vessel Segmentation* [58]. Cinco ejecuciones se llevaron a cabo con U-Net y NASGP-Net para efectos de comparación. Es importante mencionar que para estos experimentos el peso w se estableció con un valor de 0.05, el número de épocas ha aumentado a 10 y el número de generaciones se ha disminuido a 20 en comparación con los parámetros utilizados en el experimento anterior. El parámetro β de la función de costo (ver ecuación 4.1) toma un

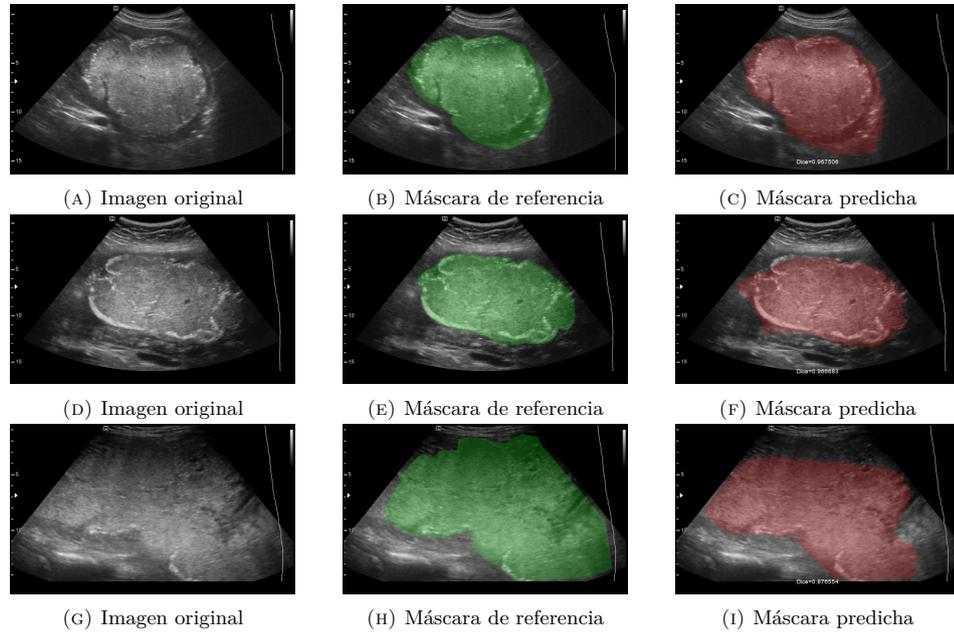


FIGURA 5.4: Mejores 3 imágenes segmentadas por NASGP-Net

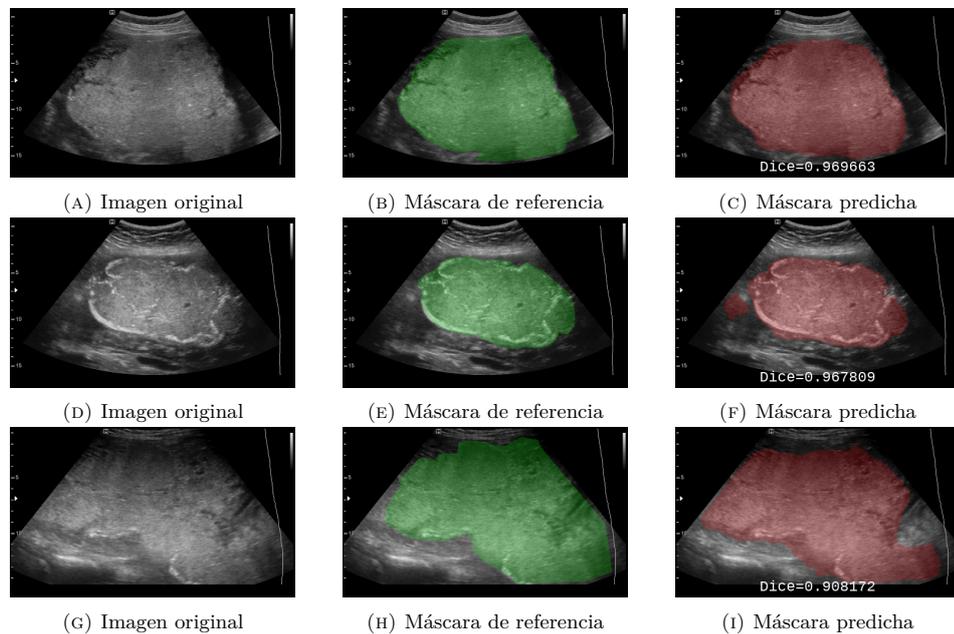


FIGURA 5.5: Mejores 3 imágenes segmentadas por U-Net

valor de 0.4, para penalizar más a los falsos positivos, es decir, píxeles que corresponden al fondo etiquetados como la región de interés [73]. En la tabla 5.6 se muestran los parámetros de NASGP-Net para estas ejecuciones.

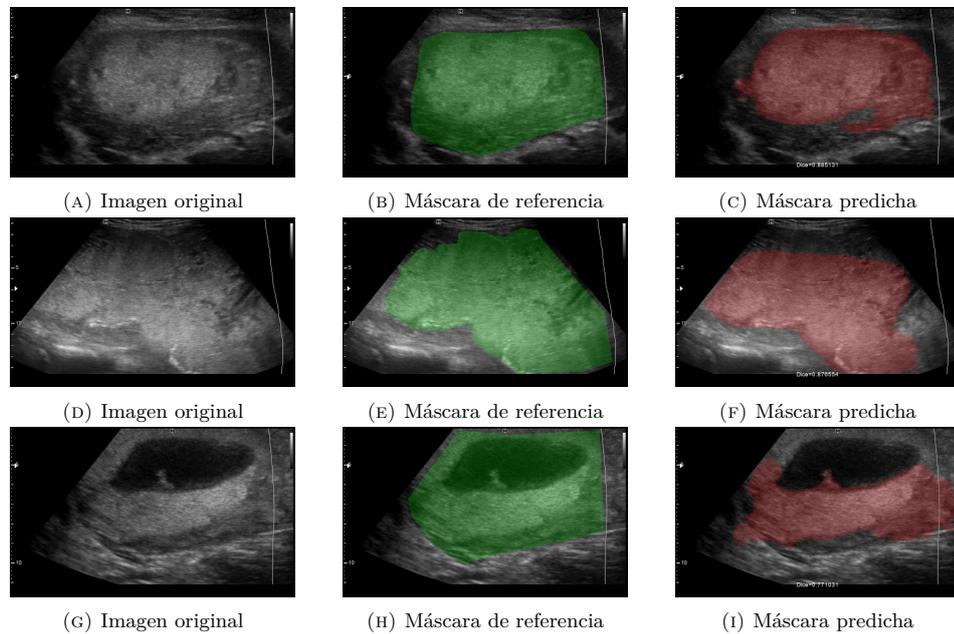


FIGURA 5.6: Peores 3 imágenes segmentadas por NASGP-Net

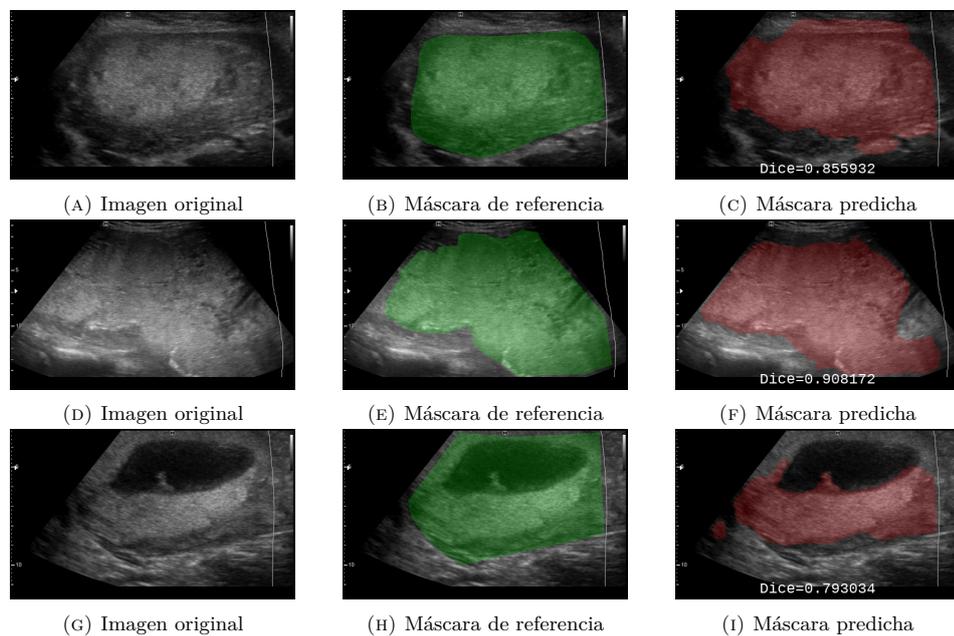


FIGURA 5.7: Peores 3 imágenes segmentadas por U-Net

En la tabla 5.7 se muestran las 5 ejecuciones realizadas por cada método, árbol sintáctico que representa la primera célula de la red neuronal convolucional neuroevolucionada y las métricas obtenidas. Las estadísticas obtenidas se resumen en la tabla 5.8.

TABLA 5.6: Parámetros de NASGP-Net utilizando *Retinal Vessel Segmentation*

Parámetro	Valor	Parámetro	Valor
Tamaño de población	100	Peso de complejidad	0.05
Número de generaciones	20	Máximo no. de parámetros	31 038 000
Probabilidad de cruza	0.8	Función de costo	combo loss
Probabilidad de mutación	0.19	Número de épocas	10
Tamaño de torneo	3	Tasa de aprendizaje	0.0001

Con base en las estadísticas y el valor p calculado por cada métrica, el modelo diseñado por el algoritmo NASGP-Net superó al modelo U-Net considerablemente. Con un valor de peso de complejidad $w = 0.05$ se obtuvieron modelos cuya complejidad considera entre el 4% y 20% de los 31M de parámetros que componen U-Net. Nótese que la disminución del peso de complejidad w en comparación con el experimento anterior aumenta la complejidad de los modelos obtenidos por NASGP-Net. La diferencia del índice de Dice obtenido entre NASGP-Net y U-Net es 0.043 en el caso promedio. El desempeño de segmentación no solo es competitivo sino superior.

En la figura 5.9 se muestra la gráfica de convergencia, la profundidad y tamaño de la población a lo largo de las generaciones. Se toma la ejecución cuyo desempeño en términos del índice de Dice corresponde a la mediana de acuerdo con la tabla 5.8. El mejor y peor resultado obtenido por NASGP-Net se muestran en las figuras 5.10 y 5.11

TABLA 5.7: 5 Ejecuciones de U-Net vs NASGP-Net utilizando *Retinal Vessel Segmentation*. Se marcan en negritas los mejores resultados por cada método del conjunto de ejecuciones

U-Net				NASGP-Net					
	Dice Prom	IoU Prom.	H.D. Prom.	No. Param	Árbol Sintáctico	Dice Prom	IoU Prom.	H.D. Prom.	No. Param
1	0.893 ±0.035	0.809 ±0.056	6.873 ±1.193	31,037,698	<i>apool(rCon(scont(mod, 32, 7, 5, 1)))</i>	0.924±0.006	0.859±0.010	6.526±0.285	1,452,657
2	0.884±0.066	0.798±0.102	7.471±2.045	31,037,698	<i>apool(scont(mod, 32, 5, 3, 1))</i>	0.909±0.003	0.834±0.004	6.772±0.371	1,072,597
3	0.856±0.065	0.753±0.097	7.463±1.906	31,037,698	<i>apool(sub(dCon(const(mod, 16, 5, 3, 2), 0.5), 0.87, dCon(scont(mod, 32, 7, 3, 1), 0.7), 0.62))</i>	0.927±0.003	0.865±0.008	6.354±0.227	5,726,749
4	0.870±0.062	0.776±0.093	7.249±1.669	31,037,698	<i>apool(rCon(scont(mod, 32, 7, 5, 2)))</i>	0.911±0.007	0.863±0.009	6.347±0.270	1,452,657
5	0.884±0.039	0.795±0.062	6.961±1.244	31,037,698	<i>npool(adal, dCon(scont(mod, 8, 7, 5, 2), 0.4), 0.91, rCon(const(const(scont(mod, 32, 7, 5, 1), 8, 7, 5, 1), 32, 5, 3, 1)), 0.72))</i>	0.928 ±0.004	0.867 ±0.006	6.187 ±0.231	6,954,340

TABLA 5.8: Estadísticas obtenidas con base en las 5 ejecuciones. Se realiza la prueba estadística *U de Mann Whitney* tomando en cuenta las métricas de Dice, IoU y Hausdorff Distance. Se marcan en negritas los mejores resultados entre ambos métodos.

Estadística	Métricas					
	Dice			IoU		
	U-Net	NASGP-Net	H.D.	U-Net	NASGP-Net	H.D.
Mejor	0.893	0.928	0.809	0.867	6.873	6.187
Peor	0.856	0.909	0.753	0.834	7.471	6.772
Media	0.877	0.920	0.784	0.858	7.141	6.437
Mediana	0.884	0.924	0.789	0.863	8.691	6.354
Desv. Std.	0.0145	0.0091	0.0212	0.0135	0.2371	0.222
<i>p-value</i> (5%)	0.0079			0.0079		

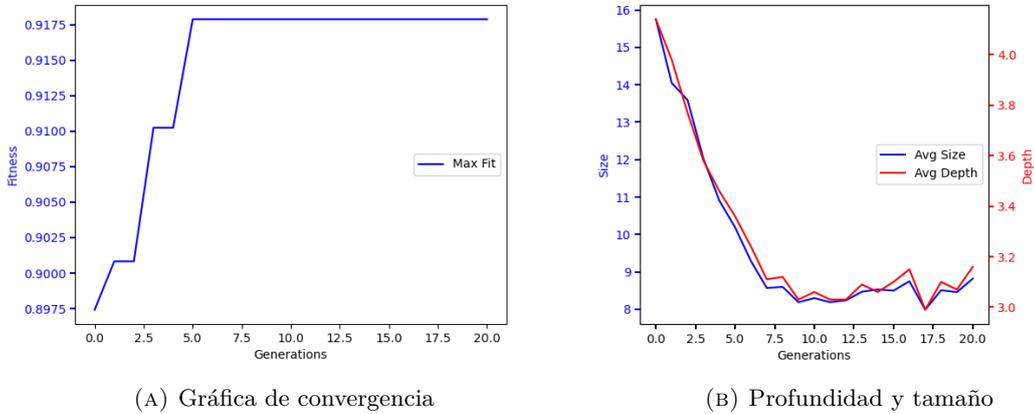


FIGURA 5.8: Gráfica de convergencia, profundidad y tamaño promedio de la ejecución correspondiente a la mediana de acuerdo con la tabla 5.8

5.3. Comentarios

Los resultados obtenidos muestran el potencial del algoritmo NASGP-Net para diseñar redes neuronales tipo U-Net con un desempeño competitivo de segmentación y un bajo costo computacional. Es importante destacar el porcentaje de reducción de parámetros en comparación con U-Net (entre el 80 y 98%). La competitividad radica en la diferencia mínima entre las métricas de segmentación monitoreadas. Lo anterior se traduce en un menor tiempo de entrenamiento y latencia del modelo.

Uno de los principales obstáculos al inicio de la implementación fue el costo computacional del algoritmo. Las estrategias implementadas para una mayor rapidez y seguridad del algoritmo permitieron reducir el tiempo de ejecución de 96 horas a 24 horas, aproximadamente.

Los parámetros de NASGP-Net que influyen directamente en el costo computacional es el número de épocas para entrenar cada red durante la etapa de evaluación y el número de generaciones del proceso evolutivo. La PG se caracteriza por considerar una población abundante y pocas generaciones. Se recomienda un tamaño de población de al menos 100 individuos, 20 generaciones y un valor de $w \leq 0.3$ para obtener resultados favorables. La elección de estos parámetros permite una diversidad en la población inicial que evoluciona durante un número de generaciones considerable. El bajo valor de w permite obtener modelos con un desempeño de segmentación competitivo y complejidad compacta. Un valor más bajo de w se traduce en una mayor complejidad en los modelos obtenidos que toma con mayor consideración el desempeño de segmentación.

En las soluciones obtenidas sobresale el uso de la operación *apool*, en lugar de *mpool* que considera originalmente U-Net. Además, las conexiones residuales (*rCon*) figuran en la mayoría de las soluciones y, con una menor frecuencia, las conexiones del tipo denso

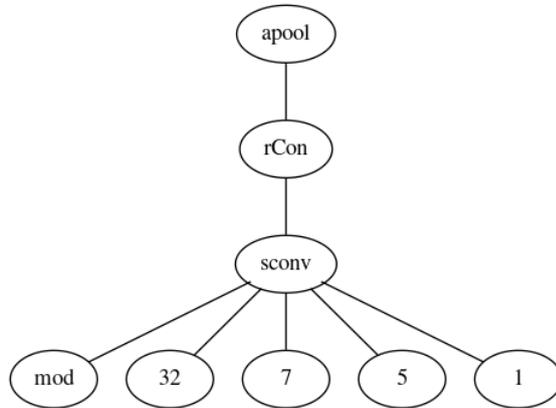


FIGURA 5.9: Árbol que representa la primera célula de la red convolucional neuroevolucionada para *Retinal Vessel Segmentation*

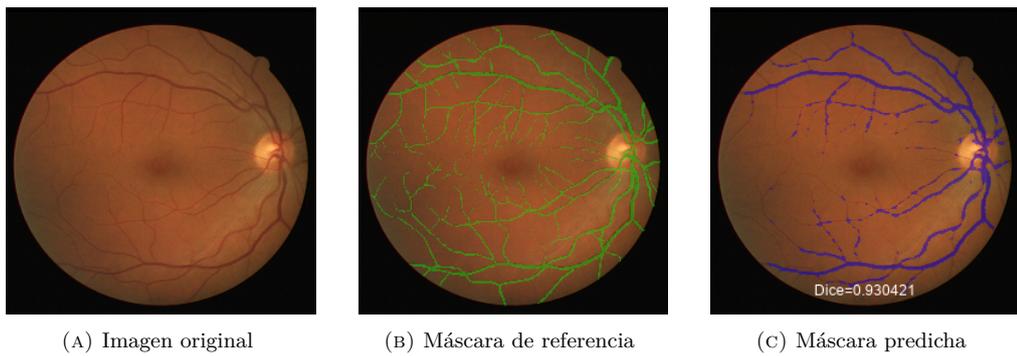


FIGURA 5.10: *Retinal Vessel Segmentation*: Mejores resultados de segmentación con *NASGP-Net*

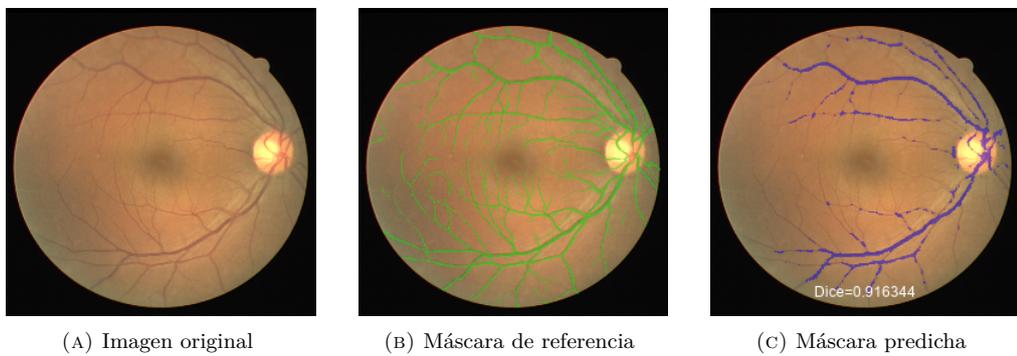


FIGURA 5.11: *Retinal Vessel Segmentation*: Peores resultados de segmentación con *NASGP-Net*

(*dCon*). Las operaciones de construcción de características (ADD, SUB, CAT) figuran también en las mejores soluciones reportadas. Particularmente la operación de recalibración (*se*) no se presenta en ninguna de las mejores soluciones reportadas.

Capítulo 6

Conclusiones y trabajo futuro

Los resultados obtenidos nos dejan claro el potencial de la PG basada en árboles como mecanismo para realizar neuroevolución. Las características que nos ofrece STGP nos permite diseñar y organizar distintas funciones afines al aprendizaje profundo, como operaciones de pooling, convolución y conectividad. Además, permite la integración de nuevas funciones de forma sencilla. Para cada función también es necesario considerar los argumentos y tipos de entrada, es decir, el conjunto de terminales. Los resultados preliminares nos muestran que un aumento en el número de filtros o en su tamaño impacta considerablemente la complejidad de los modelos obtenidos al guiar la búsqueda únicamente con el índice de Dice.

Nuestro enfoque toma como función de costo *combo loss* y utiliza capas del tipo *group normalization*, operaciones diseñadas específicamente para la tarea de segmentación de imágenes. Por otro lado, *depth-wise-separable-convolution* permite reducir el costo computacional de la operación de convolución mientras que las operaciones de *convolución dilatada* considera un campo receptivo de mayor área sin aumentar la complejidad del modelo. En nuestra propuesta, dichas operaciones pueden aplicarse de forma compuesta, permitiendo aplicar la operación *dept-wise-separable-convolution* en su versión dilatada. Las conexiones del tipo residual y denso se toman en cuenta, dado que han mostrado un mejor desempeño en comparación a sus versiones secuenciales. La ventaja que ofrece la operación *squeeze-and-excitation* para asignar un nivel de importancia a cada canal y su facilidad de incorporarla en la salida de cada capa de convolución fue la principal razón para incluirla en el espacio de búsqueda.

El algoritmo propuesto, *NASGP-Net*, permite diseñar redes neuronales convolucionales para tareas de segmentación *ah doc* al dominio del problema (placentas y retinas en este caso). Además, la complejidad de estas redes se ve reducida en comparación con U-Net en términos del número de parámetros, sin afectar significativamente el desempeño de segmentación con base en el índice de Dice. Esto confirma la hipótesis planteada.

Los resultados de los experimentos con el conjunto de datos de placentas muestran un desempeño competitivo de nuestro algoritmo en comparación con U-Net (diferencia de 0.012 unidades en el índice de Dice) y notablemente superior al evaluarlo en el conjunto de datos de retina (diferencia de 0.043 unidades en el índice de Dice). La complejidad de los modelos obtenidos oscila entre el 2 - 20% con respecto a los parámetros de U-Net,

resultado destacable de nuestro estudio. Esta reducción depende en gran medida del valor seleccionado w en la función de aptitud (ver 4.2).

Como trabajo futuro se plantea estudiar la generalidad del algoritmo propuesto al evaluarse en más bases de datos. Se contemplan por ahora *Skin Lesson* [63], *Brain CT images with Intracranial Hemorrhage* [6] y *COVID-19* [12].

La inclusión de las operaciones propuestas en la literatura aplicadas a nuestro enfoque pueden contribuir a la robustez de NASGP-Net con un costo computacional menor. Por mencionar algunos: *convolución por grupos* [79], *pooling polinomial* [80] y *módulos de atención* [25]. Una de las desventajas de nuestro enfoque es que se limita a evolucionar la *micro-estructura* topológica de U-Net [8], esto es, la configuración de las operaciones consideradas en cada célula. Por otra parte, las conexiones atajo o *skip connections*, número de células o hiperparámetros como la tasa de aprendizaje, función de costo y optimizador no son considerados en nuestra representación.

Finalmente, la elección del valor w en la función de aptitud 4.2 no es una tarea trivial. Para contender contra este problema un enfoque de optimización multi-objetivo contribuiría a evitar probar con distintos valores de w , dado que el costo computacional del algoritmo es considerable (entre 18-24 horas) a pesar de las estrategias implementadas (*checkpoint*, *memoria caché*, *multiprocesamiento*). En este mismo sentido, la aplicación de estrategias como *modelos subrogados* puede contender contra el alto costo computacional.

Apéndice A

Publicaciones del Proyecto



Universidad Veracruzana
Instituto de Ciencias de la Salud

La Universidad Veracruzana
a través del Instituto de Ciencias de la Salud
Otorga la presente



Instituto
de Ciencias
de la Salud

CONSTANCIA

a: **José Antonio Fuentes Tomás**

coautores:

Dr. Héctor Gabriel Acosta Mesa, Dr. Efrén Mezura Montes,
Dr. Rodolfo Hernández Jiménez

por la Ponencia Modalidad Oral-Trabajos libres de Maestría

**“Neuroevolución para la segmentación de placentas
en imágenes de ultrasonido 2D”**

Presentada durante el
3er. Coloquio de Investigación en Ciencias de la Salud:
De la célula a la comunidad

llevado a cabo en la Ciudad de Xalapa, Enríquez
los días 1,2 y 3 de Junio de 2022.

Dra. María Gabriela Nachón García
Instituto de Ciencias de la Salud
DIRECTORA

Dra. Patricia Pavón León
Coordinadora de Investigación
Instituto de Ciencias de la Salud

Apéndice B

Carta de consentimiento informado

Carta de consentimiento informado

Título de la investigación: Segmentación de placentas en imágenes de ultrasonido 2D evolucionando redes neuronales convolucionales

Ing. José Antonio Fuentes Tomás

Dr. Rodolfo Hernández-Jiménez

Dr. Héctor Gabriel Acosta Mesa

Dr. Efrén Mezura Montes

Estimada participante:

Le agradezco su interés de participar en este estudio, el cual forma parte de mi trabajo de tesis para obtener el grado de Maestro en Inteligencia Artificial dentro del Instituto de Investigaciones en Inteligencia Artificial de la Universidad Veracruzana. El objetivo de la presente investigación es **segmentar las imágenes de placentas obtenidas por ultrasonido**. Con este estudio permitirá que, en un futuro, detectar y prevenir oportunamente calcificaciones en la placenta a fin de procurar la salud del feto.

Los datos obtenidos en línea de la presente investigación serán utilizados exclusivamente con fines científicos manteniendo la confidencialidad de los participantes, cuidando su integridad y apegados a las normas de investigación en salud vigentes en México. Cabe mencionar que sus datos serán manejados a través de un código numérico, el cual, sólo los investigadores tendrán acceso y su uso será única y exclusivamente para fines de investigación. Los resultados del estudio podrán ser publicados bajo términos de confidencialidad de datos, en revistas de circulación internacional sin proporcionar datos personales. Su participación en este estudio no representa riesgos.

Es importante que Usted conozca la siguiente información:

- Su decisión es completamente voluntaria y puede rehusarse a participar.
- No habrá ninguna consecuencia desfavorable en caso de no aceptar la invitación.
- Si decide participar en el estudio, puede retirarse en el momento que lo desee, sin necesidad de expresar las razones de su decisión.
- No tendrá que hacer gasto alguno durante el estudio.
- No recibirá pago por su participación.

Si considera que no hay dudas ni preguntas acerca de otorgar el permiso, puede, si así lo desea, firmar la Carta de Consentimiento Informado.

Si usted tiene alguna pregunta, comentario o preocupación con respecto al proyecto de investigación, puede comunicarse a cualquiera de correos electrónicos de las investigadoras responsables:

- Ing. José Antonio Fuentes Tomás. Alumno de Maestría en Inteligencia Artificial, Universidad Veracruzana. josefuentes.at@gmail.com

- Dr. Rodolfo Hernández Jiménez: Consultorio de Ginecología-obstetricial y esterilidad. Diego Leño No. 22 Col. Centro, 91000, Xalapa-Enríquez, Veracruz. roheji@msn.com
- Dr. Héctor Gabriel Acosta Mesa: Instituto de Investigaciones en Inteligencia Artificial. Universidad Veracruzana. Campus Sur, Calle Paseo No. 112, Lote II, Sección Segunda, Col. Nuevo Xalapa, 91097, Xalapa-Enríquez, Veracruz. heacosta@uv.mx
- Dr. Efrén Mezura Montes: Instituto de Investigaciones en Inteligencia Artificial. Universidad Veracruzana. Campus Sur, Calle Paseo No. 112, Lote II, Sección Segunda, Col. Nuevo Xalapa, 91097, Xalapa-Enríquez, Veracruz. emezura@uv.mx

Nombre

Firma

Fecha

Bibliografía

1. Ba, J. L., Kiros, J. R. e Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).
2. Baldeon-Calisto, M. y Lai-Yuen, S. K. AdaResU-Net: Multiobjective adaptive convolutional neural network for medical image segmentation. *Neurocomputing* **392**, 325-340 (2020).
3. Bankman, I. *Handbook of medical image processing and analysis* (Elsevier, 2008).
4. Bengio, Y., Simard, P. y Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks* **5**, 157-166 (1994).
5. Bi, Y., Xue, B. y Zhang, M. *An automatic feature extraction approach to image classification using genetic programming* en *International Conference on the Applications of Evolutionary Computation* (2018), 421-438.
6. *Brain CT images with Intracranial Hemorrhage* <https://www.kaggle.com/datasets/vbookshelf/computed-tomography-ct-images>. Accessed: 2021-05-25.
7. Burton, G. J. y Jauniaux, E. What is the placenta? *American journal of obstetrics and gynecology* **213**, S6-e1 (2015).
8. Calisto, M. B. y Lai-Yuen, S. K. EMONAS-Net: Efficient multiobjective neural architecture search using surrogate-assisted evolutionary algorithm for 3D medical image segmentation. *Artificial Intelligence in Medicine* **119**, 102154 (2021).
9. Castejón, O., Belouche, R. y Morett, V. El proceso de calcificación en la placenta humana. *Gac Méd Caracas* **106**, 496-501 (1998).
10. Chauvin, Y. y Rumelhart, D. E. *Backpropagation: theory, architectures, and applications* (Psychology press, 2013).
11. Chollet, F. *Xception: Deep learning with depthwise separable convolutions* en *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), 1251-1258.
12. *COVID-19-MedicalSegmentation* <https://www.kaggle.com/datasets/vbookshelf/computed-tomography-ct-images>. Accessed: 2021-05-25.
13. Dumoulin, V. y Visin, F. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285* (2016).
14. Eiben, A. E. y Smith, J. From evolutionary computation to the evolution of things. *Nature* **521**, 476-482 (2015).

15. Evans, B., Al-Sahaf, H., Xue, B. y Zhang, M. *Evolutionary deep learning: A genetic programming approach to image classification* en *2018 IEEE Congress on Evolutionary Computation (CEC)* (2018), 1-6.
16. Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A. G., Parizeau, M. y Gagné, C. DEAP: Evolutionary algorithms made easy. *The Journal of Machine Learning Research* **13**, 2171-2175 (2012).
17. Gallegos, S. A. y col. La placenta humana: Revisión. *Perinatología y Reproducción Humana* **22**, 230-245 (2008).
18. Galván, E. y Mooney, P. Neuroevolution in deep neural networks: Current trends and future challenges. *IEEE Transactions on Artificial Intelligence* (2021).
19. Glorot, X. y Bengio, Y. *Understanding the difficulty of training deep feedforward neural networks* en *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (2010), 249-256.
20. Glorot, X., Bordes, A. y Bengio, Y. *Deep sparse rectifier neural networks* en *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (2011), 315-323.
21. Gonzalez, R. C., Woods, R. E. y col. *Digital image processing* 2002.
22. Goodfellow, I., Bengio, Y., Courville, A. y Bengio, Y. *Deep learning* **2** (MIT press Cambridge, 2016).
23. Grannum, P. A., Berkowitz, R. L. y Hobbins, J. C. The ultrasonic changes in the maturing placenta and their relation to fetal pulmonic maturity. *American journal of obstetrics and gynecology* **133**, 915-922 (1979).
24. Han, M. y col. Automatic segmentation of human placenta images with U-Net. *IEEE Access* **7**, 180083-180092 (2019).
25. Hassanzadeh, T., Essam, D. y Sarker, R. Evolutionary Deep Attention Convolutional Neural Networks for 2D and 3D Medical Image Segmentation. *Journal of Digital Imaging*, 1-18 (2021).
26. Hassanzadeh, T., Essam, D. y Sarker, R. *EvoU-Net: An evolutionary deep fully convolutional neural network for medical image segmentation* en *Proceedings of the 35th Annual ACM Symposium on Applied Computing* (2020), 181-189. <https://doi.org/10.1145/3341105.3373856>.
27. He, K., Zhang, X., Ren, S. y Sun, J. *Deep Residual Learning for Image Recognition* 2015. arXiv: 1512.03385 [cs.CV].
28. He, K., Zhang, X., Ren, S. y Sun, J. *Identity mappings in deep residual networks* en *European conference on computer vision* (2016), 630-645.
29. Hendee, W. R. y Ritenour, E. R. *Medical imaging physics* (John Wiley & Sons, 2003).
30. Holzman, J., Zalud, I. y Bartholomew, M. Ultrasound of the placenta. *Donald Sch J Ultrasound Obstet Gynaecol* **1**, 47-60 (2007).

31. Housley, R. *A 224-bit one-way hash function: SHA-224* inf. téc. (RFC 3874, September, 2004).
32. Howard, A. G. *y col.* Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).
33. Hu, J., Shen, L. y Sun, G. *Squeeze-and-excitation networks* en *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), 7132-7141.
34. Hu, R. *Automated analysis of the placenta in ultrasound* Tesis doct. (University of British Columbia, 2019).
35. Hu, R., Singla, R., Deeba, F. y Rohling, R. N. Acoustic shadow detection: study and statistics of B-mode and radiofrequency data. *Ultrasound in medicine & biology* **45**, 2248-2257 (2019).
36. Hu, R., Singla, R., Yan, R., Mayer, C. y Rohling, R. N. *Automated placenta segmentation with a convolutional neural network weighted by acoustic shadow detection* en *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (2019), 6718-6723.
37. Huang, G., Liu, Z., van der Maaten, L. y Weinberger, K. Q. *Densely Connected Convolutional Networks* 2018. arXiv: 1608.06993 [cs.CV].
38. Hubel, D. H. y Wiesel, T. N. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology* **195**, 215-243 (1968).
39. INEGI. *Características de las defunciones fetales registradas en México durante 2019* url https://www.inegi.org.mx/contenidos/saladeprensa/boletines/2020/EstSociodemo/DefuncionesFetales2019_08.pdf. Accessed 01-31-2021. 2020.
40. Ioffe, S. y Szegedy, C. *Batch normalization: Accelerating deep network training by reducing internal covariate shift* en *International conference on machine learning* (2015), 448-456.
41. Irwin-Harris, W., Sun, Y., Xue, B. y Zhang, M. *A graph-based encoding for evolutionary convolutional neural network architecture design* en *2019 IEEE Congress on Evolutionary Computation (CEC)* (2019), 546-553.
42. Kamnitsas, K. *y col.* Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. *Medical image analysis* **36**, 61-78 (2017).
43. Koza, J. R. y Koza, J. R. *Genetic programming: on the programming of computers by means of natural selection* (MIT press, 1992).
44. Lauria, M. *y col.* The use of second-trimester transvaginal sonography to predict placenta previa. *Ultrasound in Obstetrics and Gynecology: The Official Journal of the International Society of Ultrasound in Obstetrics and Gynecology* **8**, 337-340 (1996).
45. Liu, H., Simonyan, K. y Yang, Y. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055* (2018).

46. Liu, Y. *y col.* A survey on evolutionary neural architecture search. *IEEE Transactions on Neural Networks and Learning Systems* (2021).
47. Looney, P. *y col.* Automatic 3D ultrasound segmentation of the first trimester placenta using deep learning en *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)* (2017), 279-282.
48. McGhie, A., Xue, B. *y* Zhang, M. *GPCNN: Evolving Convolutional Neural Networks using Genetic Programming* en *2020 IEEE Symposium Series on Computational Intelligence (SSCI)* (2020), 2684-2691.
49. McKenna, D., Tharmaratnam, S., Mahsud, S. *y* Dornan, J. Ultrasonic evidence of placental calcification at 36 weeks' gestation: maternal and fetal outcomes. *Acta obstetricia et gynecologica Scandinavica* **84**, 7-10 (2005).
50. Miller, J. F. *y col.* An empirical study of the efficiency of learning boolean functions using a cartesian genetic programming approach en *Proceedings of the genetic and evolutionary computation conference* **2** (1999), 1135-1142.
51. Mirjalili, S. en *Studies in Computational Intelligence* (Springer, 2019).
52. Montana, D. J. Strongly typed genetic programming. *Evolutionary computation* **3**, 199-230 (1995).
53. Mortazi, A. *y* Bagci, U. *Automatically designing CNN architectures for medical image segmentation* en *International Workshop on Machine Learning in Medical Imaging* (2018), 98-106.
54. Newman, P. G. *y* Rozycki, G. S. The history of ultrasound. *Surgical clinics of north America* **78**, 179-195 (1998).
55. O'Neill, M. *Riccardo Poli, William B. Langdon, Nicholas F. McPhee: a field guide to genetic programming* 2009.
56. Paszke, A. *y col.* en *Advances in Neural Information Processing Systems 32* (eds. Wallach, H. *y col.*) 8024-8035 (Curran Associates, Inc., 2019). <http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>.
57. Ramachandran, P., Zoph, B. *y* Le, Q. V. Searching for activation functions. *arXiv preprint arXiv:1710.05941* (2017).
58. *Retinal Vessel Segmentation* <https://drive.grand-challenge.org/DRIVE/>. Accessed: 2021-05-25.
59. Richardus, J. *y col.* An European concerted action investigating the validity of perinatal mortality as an outcome indicator for the quality of antenatal and perinatal care. *Journal of Perinatal Medicine* **25**, 313-324 (1997).
60. Ronneberger, O., Fischer, P. *y* Brox, T. *U-net: Convolutional networks for biomedical image segmentation* en *International Conference on Medical image computing and computer-assisted intervention* (2015), 234-241.
61. Simard, P. Y., Steinkraus, D., Platt, J. C. *y col.* Best practices for convolutional neural networks applied to visual document analysis. en *Icdar* **3** (2003).

62. Skansi, S. *Introduction to Deep Learning: from logical calculus to artificial intelligence* (Springer, 2018).
63. *Skin Lesion Segmentation* <https://challenge.isic-archive.com/>. Accessed: 2021-05-25.
64. Stevenson, G. N., Collins, S. L., Ding, J., Impey, L. y Noble, J. A. 3-D ultrasound segmentation of the placenta using the random walker algorithm: reliability and agreement. *Ultrasound in medicine & biology* **41**, 3182-3193 (2015).
65. Stöppler, M. C. *Stages of Pregnancy: Week by Week* url https://www.onhealth.com/content/1/pregnancy_tages_rimesters. 2019.
66. Suganuma, M., Kobayashi, M., Shirakawa, S. y Nagao, T. Evolution of deep convolutional neural networks using cartesian genetic programming. *Evolutionary computation* **28**, 141-163 (2020).
67. Suganuma, M., Shirakawa, S. y Nagao, T. *A genetic programming approach to designing convolutional neural network architectures* en *Proceedings of the genetic and evolutionary computation conference* (2017), 497-504.
68. Suganuma, M., Tsuchiya, D., Shirakawa, S. y Nagao, T. *Hierarchical feature construction for image classification using genetic programming* en *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (2016), 001423-001428.
69. Sun, Y., Xue, B., Zhang, M. y Yen, G. Automatically designing CNN architectures using genetic algorithm for image classification. arXiv 2018. *arXiv preprint arXiv:1808.03818* (2018).
70. Sun, Y., Xue, B., Zhang, M. y Yen, G. G. Evolving deep convolutional neural networks for image classification. *IEEE Transactions on Evolutionary Computation* **24**, 394-407 (2019).
71. Szegedy, C. y col. *Going deeper with convolutions* en *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), 1-9.
72. Taghanaki, S. A., Abhishek, K., Cohen, J. P., Cohen-Adad, J. y Hamarneh, G. Deep semantic segmentation of natural and medical images: a review. *Artificial Intelligence Review* **54**, 137-178 (2021).
73. Taghanaki, S. A. y col. Combo loss: Handling input and output imbalance in multi-organ segmentation. *Computerized Medical Imaging and Graphics* **75**, 24-33 (2019).
74. Turner, A. J. y Miller, J. F. *The importance of topology evolution in neuroevolution: a case study using cartesian genetic programming of artificial neural networks* en *International Conference on Innovative Techniques and Applications of Artificial Intelligence* (2013), 213-226.
75. Ulyanov, D., Vedaldi, A. y Lempitsky, V. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022* (2016).
76. UN. *A Neglected Tragedy* url <https://data.unicef.org/resources/a-neglected-tragedy-stillbirth-estimates-report/>. Accessed 01-31-2021. 2020.

77. Vargas, A., Amescua-Guerra, L. M., Bernal, M. A. y Pineda, C. Principios físicos básicos del ultrasonido, sonoanatomía del sistema musculoesquelético y artefactos ecográficos. *Acta Ortopédica Mexicana* **22**, 361-373 (2008).
78. Vargas-Hákim, G. A., Mezura-Montes, E. y Acosta-Mesa, H.-G. A Review on Convolutional Neural Networks Encodings for Neuroevolution. *IEEE Transactions on Evolutionary Computation* (2021).
79. Wang, X., Kan, M., Shan, S. y Chen, X. *Fully learnable group convolution for acceleration of deep neural networks* en *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), 9049-9058.
80. Wei, Z. y col. *Building detail-sensitive semantic segmentation networks with polynomial pooling* en *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), 7115-7123.
81. Weng, Y., Zhou, T., Li, Y. y Qiu, X. NAS-Unet: Neural architecture search for medical image segmentation. *IEEE Access* **7**, 44247-44257 (2019).
82. Wu, Y. y He, K. *Group normalization* en *Proceedings of the European conference on computer vision (ECCV)* (2018), 3-19.
83. Zhao, C., Shi, W. y Deng, Y. A new Hausdorff distance for image matching. *Pattern Recognition Letters* **26**, 581-586 (2005).
84. Zhu, Y. y col. GP-CNAS: Convolutional Neural Network Architecture Search with Genetic Programming. *arXiv preprint arXiv:1812.07611* (2018).
85. Zimmer, V. A. y col. en *Medical Ultrasound, and Preterm, Perinatal and Paediatric Image Analysis* 264-273 (Springer, 2020).