Universidad de La Laguna

Machine Learning in Digital Imaging: From Medical to Applied Sciences

Autor:
Sabato Ceruso

Director:
Dra. Alicia Pareja Ríos

Codirector:
Dr. Sergio Bonaque González

Memoria para la obtención del grado de
Doctor en ingeniería informática

en la

Escuela Superior de Ingeniería y Tecnología
Departamento de Ingeniería Informática y Sistemas

Junio de 2021

«He was a fish that lived in a plastic tank. The tank had been filled with cement. The fish died when he got so accustomed to the cement that it had trouble swimming on the new medium.»

«If we behave exactly like he did, what would happen?»

Megatron-LM

A mi compañera de vida

Agradecimientos

Quiero agradecer en primer lugar a mis directores de Tesis. Alicia Pareja Ríos por su ayuda y confianza a lo largo de estos años y Sergio Bonaque González por ser guía y ejemplo durante todo este tiempo. Es indudable que sin su inestimable ayuda e implicación, este trabajo no sería el mismo. A Vicente Blanco Pérez, tutor de la ULL, por toda su ayuda ante mis múltiples preguntas durante el depósito de tesis. A Carlos Cairós Barreto por sus sabios consejos sobre esta memoria. A José Manuel Rodríguez Ramos por apoyarme desde Wooptix.

No puedo concluir sin mencionar a mi familia, a mis padres por darme la oportunidad de estudiar.

Por último, pero no menos importante, quiero agradecer con todo mi amor a Antonella, compañera de vida, por su apoyo, ánimos y comprensión. Esta tesis no fueron 10 años, pero aún así, el tiempo que le he dedicado te pertenecía.

UNIVERSIDAD DE LA LAGUNA

Resumen

Escuela Superior de Ingeniería y Tecnología

Departamento de Ingeniería Informática y Sistemas

Doctor en ingeniería informática

Machine Learning in Digital Imaging: From Medical to Applied Sciences

por Sabato Ceruso

La presente tesis pretende identificar aplicaciones donde el uso de la inteligencia artificial (IA) puede suponer una ventaja con respecto a otros métodos, desarrollando una IA especifica para cada caso identificado y detallando las ventajas que ésta ofrece. Concretamente se desarrollan tres líneas de estudio: detección automática de retinopatía diabética a partir de imágenes de fondo de ojo, reconstrucción de fase de frente de onda a partir de sus derivadas parciales, y estimación de las distancias entre los objetos de una escena a partir de un número limitado de imágenes de la misma. Para cada aplicación se desarrolla una metodología específica de IA que mejora de los métodos existentes en la literatura.

Índice general

Agradecimientos VII

Resumen IX

Acrónimos XXI

Glosario XXIII

1. Introducción 1
1.1. Introducción . . . 1
1.2. Objetivos . . . 4

2. Estado del arte 5
2.1. Estado del arte . . . 5
2.2. Clasificación . . . 7
2.2.1. LeNet . . . 7
2.2.2. AlexNet . . . 8
2.2.3. VGG . . . 11
2.2.4. Arquitecturas Inception-vX . . . 12
Inception-v1 . . . 12
Inception-v2 y v3 . . . 13
Inception-v4 . . . 15
2.2.5. Arquitecturas residuales . . . 18
ResNet . . . 18
2.2.6. Composición de arquitecturas . . . 20
Inception-ResNet . . . 20
ResNext . . . 21
SENet . . . 22
2.3. Interpretabilidad . . . 24
2.3.1. Análisis mediante oclusiones . . . 24
2.3.2. Análisis de gradientes . . . 27
2.3.3. Análisis de mapas de activaciones . . . 29

2.4. Segmentación 33
2.4.1. *Fully Convolutional Network* 34
2.4.2. U-Net 36
2.4.3. Seg-Net 37
2.4.4. PSP-Net 38
2.4.5. RefineNet 39
2.4.6. DeepLab 40

3. Retinopatía diabética **43**
3.1. Introducción 43
3.1.1. Retinopatía diabética en Canarias 45
3.2. Retisalud 47
3.3. *Deep learning* aplicado a Retisalud 47
3.3.1. Datos 47
3.3.2. Método 49
3.3.3. Evaluación 51
3.3.4. Resultados 52
Evaluación sobre los segmentos A y D 54
Evaluación sobre los segmentos B y C 55
Evaluación sobre los segmentos D, E y F 56
Reevaluación de resultados 56
3.4. Interpretabilidad 58
3.4.1. Métodos 60
Análisis de gradientes 60
Análisis de mapas de activación 63
3.4.2. Resultados 65
3.5. Conclusiones 67

4. Reconstrucción de fase de frente de onda **69**
4.1. Introducción 69
4.2. Estado del arte 72
4.2.1. Southwell 72
4.2.2. *Higher order finite difference* (Li et al.) 73
4.2.3. *Splines* (Huang) 74
4.3. Método propuesto 74
4.4. Simulaciones 76
4.5. Resultados 78
4.6. Conclusiones 80

5. *Depth from focus* **83**
5.1. Introducción . . . 83
5.2. Estado del arte . . . 84
5.3. El método . . . 86
5.3.1. Conjunto de datos . . . 86
Generación de distancias . . . 88
Generación de *focal stacks* . . . 89
Generación del mapa de índices de planos . . . 91
5.3.2. Arquitectura . . . 91
Codificador 2D - Decodificador 3D en multiescala . . . 91
Codificador 2D siamés . . . 91
Decodificador 3D . . . 92
Regresión . . . 93
5.3.3. Implementación . . . 94
Datos . . . 94
Función de costes . . . 95
5.4. Experimentos cuantitativos . . . 96
5.4.1. Métrica de evaluación . . . 96
5.4.2. Análisis del método . . . 97
Selección del conjunto de datos . . . 97
Configuraciones de arquitectura . . . 99
Análisis del entrenamiento . . . 100
5.4.3. Comparación de métodos . . . 101
5.5. Experimentos cualitativos . . . 102
5.5.1. Comparación de conjuntos de datos . . . 103
5.5.2. Comparación de métodos . . . 105
5.5.3. Análisis *focal stacks* dinámicos . . . 108
5.6. *Depth from focus* dinámico . . . 110
5.6.1. Conjunto de datos . . . 111
Selección de datos . . . 112
Extracción de distancias . . . 112
Proceso completo . . . 114
5.6.2. Arquitectura . . . 115
Feature Alignment Block . . . 117
Feature Alignment Head . . . 117
Feature Combination . . . 119
5.6.3. Implementación . . . 120
5.7. Resultados finales . . . 121

5.8. Conclusiones 125

6. Discusiones **127**
6.1. Retinopatía diabética 128
6.1.1. Posible línea futura 130
6.2. Reconstrucción de fase de frente de onda 134
6.2.1. Posible línea futura 135
6.3. *Depth from focus* 136
6.3.1. Posible línea futura 137
6.4. Conclusiones generales 139
6.4.1. Retinopatía diabética 139
6.4.2. Reconstrucción de fase de frente de onda 139
6.4.3. *Depth from focus* 140
6.5. Resultados de la investigación 140

Índice de figuras

2.1. MNIST 8
2.2. LeNet-5 8
2.3. AlexNet 9
2.4. VGG 12
2.5. Inception *block* 13
2.6. Factorización 5x5 14
2.7. Factorización *nxn* 15
2.8. Expansión de los filtros de salida 16
2.9. Inception-v4 16
2.10. *Stem* de Inception-v4 17
2.11. Módulos de Inception-v4 17
2.12. Reducción de Inception-v4 17
2.13. Módulo ResNet 18
2.14. Arquitecturas ResNets 19
2.15. Módulos de Inception-ResNet 20
2.16. Reducción de Inception-ResNet 20
2.17. *Stem* de Inception-Resnet-v1 21
2.18. Inception-ResNet 21
2.19. Diagrama de bloque ResNext 22
2.20. Bloque SE 23
2.21. ZFNet 25
2.22. AlexNet *features* 26
2.23. Análisis de oclusiones 27
2.24. Retro-propagación guiada 28
2.25. Comparativa de métodos de retro-propagación 29
2.26. CAM 31
2.27. Grad-CAM 32
2.28. FCN 34
2.29. VGG-FCN-32s 35
2.30. Comparativa FCN 35
2.31. U-Net 36

2.32. Seg-Net 37
2.33. PSP-Net 38
2.34. Bloque RefineNet 39
2.35. RefineNet 40
2.36. DeepLabv3 40

3.1. Protocolo de Retisalud 47
3.2. Imágenes de Retisalud 49
3.3. Red neuronal RD 50
3.4. Distribución del conjunto de datos de RD 52
3.5. Comparativa de criterio de la inteligencia artificial (IA) frente a la evaluación del oftalmólogo 55
3.6. Ejemplo de retinografía RDNP leve 59
3.7. Retropropagacíon guiada 61
3.8. Agregación de gradientes 62
3.9. Arquitectura de red para extracción de CAM 63
3.10. CAM de baja resolución 64
3.11. Arquitectura CAM multi-escala 64
3.12. CAMs a diferentes escalas 65
3.13. CAMs agregados 65
3.14. Resultados de interpretación DR 66

4.1. Diagrama frente de onda 69
4.2. Arquitectura de red neuronal para la reconstrucción de fases . 75
4.3. Fase sintética 77
4.4. Gradientes en x e y 78
4.5. Muestras de fases a reconstruir 79
4.6. Error de reconstrucción 80

5.1. Ejemplo de *focal stack*. 90
5.2. Arquitectura de red para *depth from focus* 91
5.3. *Multiscale aggregation block* 93
5.4. Conjunto de *test* cualitativo 103
5.5. Comparación cualitativa, diferentes conjuntos de datos 104
5.6. Comparación con VDFF 105
5.7. Comparación con VDFF y DDFF 106
5.8. Comparación con MiDaS 107
5.9. *Focal stack* con movimiento de escena 109
5.10. *Focal stack* con movimiento agresivo 109

5.11. Resultados sobre *Focal stacks* con movimiento 110
5.12. Ejemplo de selección de imágenes de una escena 112
5.13. Diagráma del algoritmo RAFT 113
5.14. Par estéreo de una imagen y *optical flows* obtenidos 113
5.15. Proceso de preprocesamiento de secuencias 114
5.16. Secuencia de imágenes desenfocadas sintéticamente 115
5.17. Arquitectura de red para *depth from focus* contemplando el movimiento 116
5.18. Diagrama del *Feature Alignment Block* 116
5.19. Comparación de *focal stacks* con movimiento de escena 122
5.20. Segunda comparación de *focal stacks* con movimiento de escena 123
5.21. Tercera comparación de *focal stacks* con movimiento de escena 124

6.1. Protocolo de verificación 131
6.2. Protocolo de aplicación 133

Índice de tablas

3.1. Resultados de la evaluación de la IA 54

4.1. Resultados del primer análisis del método de reconstrucción de fase 78
4.2. Resultados del segundo *test* del método de reconstrucción de fase 80

5.1. Resumen de los conjuntos de datos RGB-D públicos analizados. 87
5.2. Comparación de conjuntos de datos 98
5.3. Análisis de arquitectura 99
5.4. Análisis de entrenamiento 100
5.5. Comparativa con otros métodos 101

Acrónimos

AE **A**tención **E**specializada.

AP **A**tención **P**rimaria.

BN **B**atch **N**ormalization.

CAM **C**lass **A**ctivation **M**apping.

CRF **C**onditional **R**andom **F**ields.

DDFF **D**eep **D**epth **F**rom **F**ocus.

DFF **D**epth **F**rom **F**ocus.

DFT **D**iscrete **F**ourier **T**transform.

DRN **D**ilated **R**esidual **N**etwork.

FCN **F**ully **C**onvolutional **N**etworks.

FPS **F**rames **P**er **S**econd.

Grad-CAM **Grad**ient-weighted **C**lass **A**ctivation **M**apping.

IA **I**nteligencia **A**rtificial.

ILSVRC **I**mageNet **L**arge **S**cale **V**isual **R**ecognition **C**hallenge.

mDFF **m**obile **D**epth **F**rom **F**ocus.

mIoU **m**ean **I**ntersection **o**ver **U**nion.

MP **M**ega **P**íxeles.

PND **P**atología **N**o **D**iabética.

RCU **R**esidual **C**onvolution **U**nit.

RD **R**etinopatía **D**iabética.

RDNP **R**etinopatía **D**iabética **N**o **P**roliferativa.

RDP **R**etinopatía **D**iabética **P**roliferativa.

ReLU **R**ectified **L**inear **U**nit.

RGB **R**ed **G**reen **B**lue.

RGB-D **R**ed **G**reen **B**lue-**D**epth.

RMSE **R**oot **M**ean **S**quared **E**rror.

RRSE **R**oot **R**elative **S**quared **E**rror.

SFM **S**tructure **F**rom **M**otion.

SSIM **S**tructural **Sim**ilarity.

VDFF **V**ariational **D**epth **F**rom **F**ocus.

Glosario

ADAM

Algoritmo de gradiente descendiente que utiliza la información de gradiente y de momentos de primer y segundo orden para calcular el salto de cada parámetro.

Aliasing

Efecto que causa que diferentes señales continuas se tornen indistinguibles tras su muestreo.

Aprendizaje supervisado

Método de aproximación de una función a partir del análisis de un conjunto de datos que contiene muestras de posibles entradas con sus respectivas salidas esperadas.

Array

Conjunto de elementos dispuestos consecutivamente.

Average pooling

Operación de muestreo lineal que toma el valor medio del área de la muestra sobre la que aplica.

Backbone

En el ámbito del *deep learning*, se refiere al extractor de características de una red neuronal.

Backtesting

Término utilizado para referirse a la evaluación de un modelo predictivo sobre datos históricos.

BatchNorm

Operación de normalización a nivel de lote de iteración de entrenamiento. Esta operación da como salida datos normalizados a una media y varianza conocida.

Conditional random fields

Modelo estocástico utilizado para segmentar secuencias .

Convolución traspuesta

Operación de convolución con *stride* fraccionario.

Dimensiones espaciales

Aquellas dimensiones de un tensor correspondientes con el alto y ancho de una imagen.

Focal stack

Conjunto de imágenes enfocadas a distintas distancias.

Global average pooling

Operación de muestreo que consiste en obtener el valor medio a lo largo de las dimensiones espaciales de un tensor.

Gradiente descendiente

Algoritmo iterativo de optimización matemática que utiliza el gradiente de la función de costes respecto a cada uno de los parámetros a optimizar para calcular la modificación a aplicar en cada iteración.

Ground truth

Término utilizado para referirse a la información que se considera verdadera.

Hiperparámetros

Parámetros del modelo que no pueden ser aprendidos mediante el proceso de entrenamiento.

Logits

Vector de probabilidades no normalizadas.

Matting

Proceso de extracción de los objetos o zonas de una imagen que han de diferenciarse del fondo.

Max pooling

Operación de muestreo no lineal que consiste en tomar el valor máximo del área de la muestra sobre la que se aplica.

Optical flow

Algoritmo que calcula el desplazamiento de píxeles entre una imagen y otra, dando como resultado un vector de desplazamiento para cada píxel.

Padding

Relleno que se añade a los bordes de una imagen.

Pooling

Conjunto de operaciones de muestreo. Pueden utilizar como operador de muestreo el valor máximo, media, mediana, etc.

Rectificación lineal (ReLU)

Función que modifica los valores negativos de una función dejándolos a 0.

Regularizar

Proceso de añadir o modificar información con el objetivo de resolver un problema mal condicionado o evitar el sobreajuste.

Retropropagación

Método de cálculo de gradiente utilizado en algoritmos de aprendizaje supervisado.

Sigmoide

Función de activación no lineal: $sigmoid(x) = \frac{1}{1+e^{-x}}$.

Sobreajuste

Situación que se da cuando el modelo es capaz de aproximar muy bien los datos de entrenamiento pero no los de validación.

Softmax

Función que traduce un vector K-dimensional con valores en el rango $(-\infty, \infty)$ en otro K-dimensional en el rango $[0, 1]$ que cumple con la propiedad de que la suma de todos sus elementos es 1: $\sigma(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$.

Stride

Tamaño del salto en el momento de deslizar la ventana del *kernel* en la operación de convolución.

Tensor

Vector multidimensional.

Transfer learning

Técnica que busca transferir el conocimiento aprendido en un contexto y problema específico para resolver un problema de un contexto diferente.

Unpooling

Operación inversa al *pooling*.

Upsampling

Operación de sobre muestreo. Consiste en expandir la señal de entrada y filtrarla mediante interpolación.

Capítulo 1

Introducción

1.1. Introducción

Según la Real Academia Española (RAE) [1], la inteligencia artificial se define como: "Disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana, como el aprendizaje o el razonamiento lógico". Estas "operaciones" que se consideran "comparables a las que realiza la mente humana" están presentes en diversos campos de estudio y aplicación, como puede ser la visión por computador, reconocimiento del habla, toma de decisiones o traducción de lenguajes.

La presente tesis se centra en la aplicación de la inteligencia artificial, específicamente en el área de la visión por computador. Esta área, también conocida como "visión artificial", o más aún con su nombre en inglés, "*computer vision*", es la disciplina científica que abarca los métodos de adquisición, procesamiento y análisis de imágenes.

En los últimos años, la inteligencia artificial ha experimentado grandes avances, especialmente en la rama de aprendizaje automático, más conocida por su nombre en inglés "*machine learning*" que será el término a utilizar en la presente memoria.

Aunque en los capítulos siguientes se desarrolla en profundidad la evolución del *machine learning* desde sus comienzos a la actualidad, se puede identificar ya como punto de inflexión el trabajo realizado por Yann LeCun en 1989 donde implementa un modelo de aprendizaje automático para la clasificación de dígitos escritos a mano [2]. A pesar del éxito de este trabajo, no será hasta

2012 cuando ocurra una "revolución" del *machine learning* con la metodología propuesta por Krizhevsky et al [3]. Este trabajo hace aparición en la competición de clasificación de imágenes "*ImageNet Large Scale Visual Recognition Challenge*" (ILSVRC) [4]. Esta competición consiste en resolver el problema de asignación de la etiqueta correcta a cada imagen de un conjunto de datos. En concreto, se utiliza como conjunto de datos ImageNet [5], una base de datos compuesta por millones de imágenes catalogadas en 1000 categorías diferentes.

El trabajo de Krizhevsky et al utiliza una red neuronal convolucional para lograr resolver el problema de clasificación. Una red neuronal es un modelo matemático que define una serie de operaciones parametrizables a realizar sobre una entrada determinada (este concepto será desarrollado en el capítulo 2). Estas operaciones parametrizables son los elementos a ser aprendidos durante un proceso de optimización realizado para minimizar el error del modelo. A este proceso se le denomina "entrenamiento". El término convolucional viene dado del hecho de que las operaciones utilizadas por este modelo matemático incluyen convoluciones. De ahí, la solución propuesta por Krizhevsky et al es un modelo matemático compuesto de una serie de convoluciones y funciones no lineales (llamadas activaciones) que operan sobre una imagen de entrada para dar una clasificación de la misma. Este conjunto de operaciones, agrupadas en subconjuntos llamados "capas", conforman una red neuronal convolucional. Como resultado, se logró una mejora sin precedentes respecto a los algoritmos anteriores.

Con este resultado, se hace patente la capacidad de las redes neuronales convolucionales para resolver problemas que requieran la agregación y correlación de grandes cantidades de información espacial y/o temporal para poder dar una solución correcta.

En el campo de la visión por computador, esta gran capacidad de correlación de información permite a la red neuronal capturar características consideradas de "alto nivel", contrastando con las de "bajo nivel". Una característica en el ámbito de la visión por computador es un elemento visual que puede ser reconocido en distintos objetos o escenas. Se considera de "bajo nivel" a aquellas características que pueden ser extraídas de una imagen realizando únicamente un análisis a nivel de señal. Por ejemplo, un borde se considera una característica de "bajo nivel" pues, este puede ser detectado analizando las diferencias de píxeles adyacentes de una imagen. Por otro lado, se considera de "alto nivel" a las características que requieren conocimiento de

semántica del problema a resolver. Un ejemplo de característica de alto nivel puede ser la posición de los ojos en una imagen. Esta capacidad de detectar formaciones complejas en una imagen, específicas en muchos casos a un contexto concreto, permite afrontar problemas hasta la fecha irresolubles. Por ejemplo, detectar los diferentes objetos de una escena, clasificarlos según una lista de clases preestablecidas o incluso hacer un estimativo de la distancia relativa de cada objeto a la cámara. Es por esta razón, de aprender la semántica, que una red neuronal puede ser considerada una inteligencia artificial según la definición de la RAE, pues tanto el aprendizaje como la comprensión de la semántica son operaciones propias de la mente humana.

En esta línea, la presente tesis pretende identificar y plantear casos de aplicación de métodos de solución basados en redes neuronales. Estando estructurada como sigue: en el capítulo 1 se realiza una contextualización donde se presenta la rama de *machine learning*, seguida de una explicación de los ámbitos de aplicación identificados así como su motivación y, por último, el planteamiento formalizado de los objetivos de esta tesis. En el capítulo 2 se hace un recorrido por el estado del arte de los temas de interés para la tesis. En el capítulo 3 se presenta el desarrollo realizado para la aplicación de técnicas de *machine learning* al problema de la detección de retinopatía diabética a partir de imágenes de fondo de ojo. En el capítulo 4 se muestra el segundo ámbito de aplicación identificado: el problema de reconstrucción de fases de frente de onda a partir de sus derivadas parciales. En el capítulo 5 se presenta la investigación realizada y los resultados obtenidos en el problema de extracción de distancias a partir de un conjunto de imágenes desenfocadas (este conjunto se le conoce mejor por su término en inglés, "*focal stack*", que será el utilizado en la presente memoria). Por último, en el capítulo 6 se exponen las conclusiones obtenidas a partir de los resultados de la tesis y se plantean una serie de posibles líneas futuras de investigación.

1.2. Objetivos

Tal y como se expuso en la sección 1.1, la inteligencia artificial, concretamente la rama de *machine learning*, gana especial interés dado el gran éxito logrado a la hora de resolver problemas hasta la fecha de difícil resolución. En la misma línea, esta tesis pretende identificar aquellas áreas en que la utilización de técnicas basadas en *machine learning* puedan aportar mejoras, identificando, por un lado, la aplicación directa a la reconstrucción de fases de frente de onda a partir de sus derivadas parciales, por otro, como método de asistencia al cribado de la retinopatía diabética, y, por último, como método de resolución al problema de extracción de distancias a partir de *focal stacks*.

Una vez identificados estas tres posibles aplicaciones, se plantean los siguientes objetivos:

1. Analizar el problema del cribado automático de la retinopatía diabética.
2. Lograr la aplicación de técnicas de *machine learning* para el cribado de la retinopatía diabética.
3. Verificar la viabilidad de la mejora propuesta para el cribado de la retinopatía diabética sobre muestras reales.
4. Lograr la correcta aplicación de técnicas de *machine learning* al problema de reconstrucción de fase de frente de onda a partir de sus derivadas parciales.
5. Analizar los resultados obtenidos con nuestro método frente a los algoritmos clásicos en simulaciones.
6. Lograr aplicar técnicas de *machine learning* para obtener distancias utilizando cámaras pasivas monoculares. Específicamente, mediante el análisis de imágenes con distintos enfoques.
7. Comparar los resultados obtenidos para la obtención de distancias con métodos del estado del arte.
8. Dar solución a los problemas inherentes a las técnicas de extracción de distancias a partir del enfoque: la generalización a distintas escenas y cámaras, y el movimiento entre imágenes de un mismo *focal stack*.

Capítulo 2

Estado del arte

2.1. Estado del arte

Tal y como se explica en el capítulo 1, el propósito del presente trabajo es la aplicación de técnicas de inteligencia artificial, específicamente de *machine learning*, al campo de detección de retinopatía diabética, la reconstrucción de fase de frente de onda a partir de sus derivadas parciales, y, estimación de distancias a partir de imágenes desenfocadas. Si bien son problemas de contextos muy diferentes, cuentan con una serie de características comunes:

- Todos los problemas parten de imágenes para dar una respuesta: retinografías, derivadas parciales (considerando a las matrices de derivadas parciales como imágenes) o un conjunto de imágenes desenfocadas.
- Todos necesitan del conocimiento específico para resolverse correctamente: en el caso del análisis de retinografías es necesario el conocimiento de un médico experto para poder detectar los signos de retinopatía diabética. En el caso de reconstrucción de fase es necesario conocer la geometría subyacente al problema para poder aproximar correctamente la superficie a reconstruir. En el caso de la estimación de distancias es necesario el conocimiento de la óptica del sistema en cuestión para poder analizar correctamente el enfoque y así estimar distancias.
- Este conocimiento necesario puede ser aprendido de los datos: en el caso de retinografías puede aprenderse a partir de imágenes ya etiquetadas bajo los estándares deseados. En el caso de la reconstrucción de fase puede aprenderse a partir de fases sintéticas generadas con la distribución de probabilidad conocida del problema a resolver. En el caso de la estimación de distancias puede aprenderse a partir de conjuntos de imágenes de distintas escenas generadas con diferentes cámaras.

Dadas estas características, los avances recientes en redes neuronales resultan una herramienta excepcional para afrontar estos tres problemas.

Por parte de la detección de retinopatía diabética resultan cruciales los últimos avances en materia de clasificación de imágenes, donde se puede obtener una respuesta de si se ha detectado la presencia de algún signo de retinopatía diabética, así como los avances en materia de aprendizaje poco supervisado e interpretabilidad de modelos, pues, lograr obtener junto a la clasificación realizada, la justificación de la razón por la que se llega a dicha conclusión resulta deseable.

Por parte de la reconstrucción de fase de frente de onda, resultan de interés los avances realizados en materia de segmentación de imágenes, pues, si bien una segmentación es una clasificación a nivel de píxel de una imagen, para realizarla es necesario primero descomponer la imagen en características esenciales de alto nivel mediante un codificador (una red neuronal que descompone una imagen en vectores de características), para luego recuperar mediante un decodificador (otra red neuronal que a partir del vector de características estima la salida) el volumen de mapas de características necesario para luego aplicar la clasificación lineal. Son justamente estas arquitecturas codificador-decodificador, sustituyendo su última capa de clasificación lineal por una regresión y su función de costes durante el entrenamiento de entropía cruzada por el error cuadrático medio (como se detallará más adelante), las que permitirán afrontar el problema de reconstrucción de fase de frente de onda a partir de sus derivadas aprendiendo la geometría específica del problema en cuestión.

Por parte de la estimación de distancias, al igual que en el caso del problema anterior, resultan de interés los avances en materia de segmentación. Especialmente de interés son las arquitecturas codificador-decodificador, pues, los codificadores serán los utilizados para extraer la información de enfoque de cada imagen de entrada, mientras que el decodificador será el encargado de recuperar el mapa de distancias estimado.

Dadas estas razones, el presente estado del arte se estructura como sigue: en la secciones 2.2 y 2.4 se hace un recorrido por los últimos avances en materia de clasificación y segmentación con redes neuronales, y en la sección 2.3 se presentan los últimos avances en materia de interpretabilidad de modelos.

2.2. Clasificación

El problema de clasificación consiste en asignar a una imagen de entrada una etiqueta de un conjunto preestablecido de categorías. A pesar de la simplicidad del problema, es uno de los poblemas clave en visión por ordenador, pues, muchos otros problemas como pueden ser la detección de objetos o la segmentación pueden ser reducidos a un problema de clasificación.

2.2.1. LeNet

En 1989 Yann LeCun propuso la primera red neuronal convolucional multicapa entrenada mediante aprendizaje supervisado utilizando el algoritmo de retropropagación para resolver el problema de clasificación de dígitos escritos a mano [2].

El aprendizaje supervisado es un método de aproximación de una función a partir del análisis de un conjunto de datos que contiene muestras de dicha función. El algoritmo de retropropagación es un método de cálculo del gradiente de una función respecto a una serie de parámetros.

En 1998, LeCun propone una versión mejorada que será conocida como LeNet-5 [6]. De este trabajo resultan clave dos aspectos: el primero, el modo de entrenamiento, pues, el algoritmo de gradiente descendiente (o variantes del mismo) sigue siendo el utilizado hoy en día; y el segundo, la utilización de convoluciones para extraer las características clave de cada imagen aprovechando la coherencia espacial inherente a este tipo de dato. Es esta nueva forma de extraer características, utilizando convoluciones frente a los productos internos utilizados clásicamente por redes neuronales artificiales, la que da el nombre a este nuevo tipo de método: red neuronal convolucional.

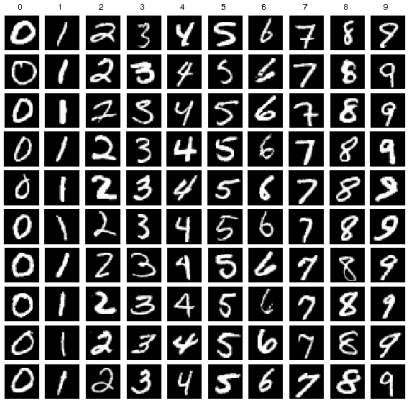


FIGURA 2.1: Ejemplo del dataset MNIST para el reconocimiento de dígitos escritos a mano [7].

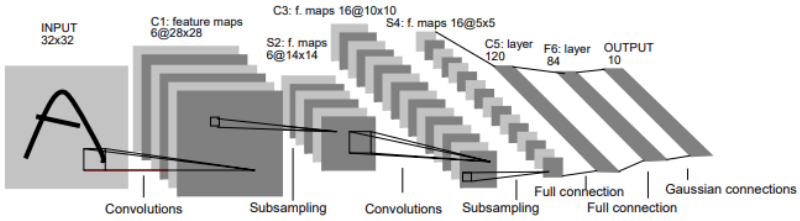


FIGURA 2.2: Arquitectura de LeNet-5. Imagen extraída de [6].

En la figura 2.1 se muestra un extracto del conjunto de datos de dígitos escritos a mano MNIST [7], mientras que en la figura 2.2 se muestra la arquitectura de LeNet-5 para la clasificación de dígitos.

2.2.2. AlexNet

A pesar del éxito en el problema de clasificación de dígitos, el uso de redes neuronales no se generalizó hasta años más tarde debido a problemas como el excesivo coste computacional para los dispositivos de la época o a las limitaciones arquitecturales como la que supone el desvanecimiento o explosión de gradiente que sufren las funciones de activación utilizadas en aquel

momento (estos términos son mejor conocidos por su nombre en inglés, "*vanishing gradient*" y "*exploding gradient*", que serán los utilizados en la presente tesis y serán explicados a continuación). En este sentido, avances en las arquitecturas como la rectificación lineal (ReLU) [8] o la aparición de herramientas como CUDA [9] permitieron marcar el siguiente hito en esta rama de la inteligencia artificial, este es, la entrada de AlexNet [3] en la competición ILSVRC [4] del 2012. La entrada de AlexNet marca un punto de inflexión sin precedentes, pues, logra reducir el error top-5 del año anterior en un 10 %.

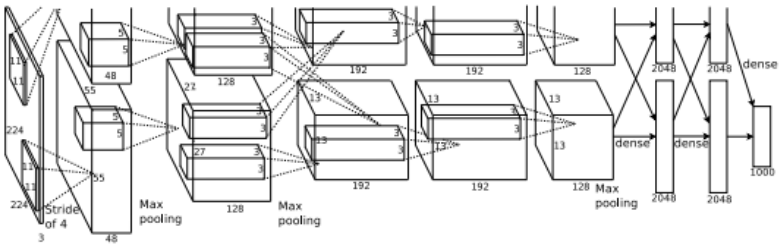


FIGURA 2.3: Arquitectura de AlexNet. Imagen extraída de [3].

AlexNet tiene una arquitectura similar a LeNet-5, pero más profunda, 8 en lugar de 5 capas, y, con más filtros por capa (o más "ancha"), presentada en la figura 2.3. Al ser más profunda, goza de mejor capacidad de generalización pero aparecen los problemas de sobre ajuste y de *vanishing gradient*.

El problema de sobre ajuste consiste en el ajuste del modelo a las características específicas de los datos de entrenamiento en lugar de las características clave o generales del problema a resolver, por lo que, en caso de tener un modelo sobreajustado se tendrá un buen resultado sobre el conjunto de entrenamiento pero un desempeño muy pobre sobre cualquier otro conjunto de datos. Para afrontar este problema, AlexNet incorpora dos métodos: el *dropout* [10], que consiste en eliminar aleatoriamente algunas conexiones durante el entrenamiento para así forzar al modelo a aprender características más robustas y el *data augmentation* que consiste en aplicar ciertas transformaciones a las imágenes de entrada para así aumentar artificialmente el conjunto de datos de entrenamiento.

Por otro lado, en lo que respecta al problema del *vanishing gradient*, este surge de forma natural al entrenar una red neuronal mediante el algoritmo de

gradiente descendiente. Para realizar una iteración de entrenamiento es necesario calcular el gradiente de la función de costes respecto a cada uno de los pesos del modelo. Por tanto, según la regla de la cadena, a mayor profundidad del mismo, más larga será la composición de productos para obtener las derivadas de las primeras capas. En caso de que las derivadas estén compuestas en el rango $[0, 1]$ se caería en el caso de *vanishing gradient* pues llegará el punto en que no se tenga precisión para representar números tan pequeños. En caso contrario, se presentaría el problema del *exploding gradient*, pues, se tendrían derivadas que tenderían a $\pm\infty$. Uno de los responsables de este problema es la función de activación sigmoide, pues, su derivada vale 0 en caso de saturación.

$$S(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

$$\frac{\delta S(x)}{\delta x} = \frac{e^{-x}}{(1 + e^{-x})^2} \tag{2.2}$$

Así, en la ecuación 2.1 y 2.2 se muestra la ecuación de la función sigmoide. Para paliar el problema en cuestión, AlexNet sustituye esta activación por ReLU [8], presentada en la ecuación 2.3 y su derivada en 2.4. Esta función permite añadir la no-linealidad necesaria de una función de activación a la vez que evita el problema de *vanishing gradient* al tener como derivada 1 para los casos en que se tiene una entrada positiva.

$$ReLU(x) = \begin{cases} x & \text{si } x > 0 \\ 0 & \text{en otro caso} \end{cases} \tag{2.3}$$

$$\frac{\delta ReLU(x)}{\delta x} = \begin{cases} 1 & \text{si } x > 0 \\ 0 & \text{en otro caso} \end{cases} \tag{2.4}$$

Estas modificaciones, la utilización de convoluciones con máscaras de convolución (término mejor conocido en la literatura como "*kernel*") más grandes en las primeras capas y el uso de *max pooling* [11] ("*pooling*" es una operación de muestreo mientras que el término "*max*" indica que se utiliza la función "máximo" para reducir cada ventana de muestreo a un valor) para la reducción de dimensionalidad espacial junto con el protocolo de entrenamiento

multi-GPU permitieron obtener el resultado excepcional que logró AlexNet en su entrada en el ILSVRC del 2012.

El éxito de AlexNet marca el inicio de una nueva era de interés recobrado en las redes neuronales convolucionales, en hacerlas cada vez más grandes y profundas para lograr conseguir cada vez mejor precisión.

2.2.3. VGG

En esta misma línea, en 2014 hace aparición la arquitectura VGG [12]. Este modelo es mucho más profundo que sus predecesores, pues cuenta con 19 capas siguiendo una arquitectura de capas convolucionales modulares.

Es en este momento en que inicia una tendencia a hacer los modelos cada vez más profundos (grandes) para lograr mejores resultados. De esta tendencia de utilizar modelos cada vez más profundos nace el término "aprendizaje profundo", o más conocido por su nombre en inglés, "*deep learning*", que será el utilizado en la presente tesis. Este término hace referencia a la técnica de *machine learning* aplicada a modelos profundos y/o a grandes volúmenes de datos.

Otro aspecto innovador de este modelo es el tamaño de sus convoluciones, pues solo utiliza convoluciones $3x3$, demostrando así la capacidad de extracción de características con la concatenación de convoluciones pequeñas evitando así la necesidad de utilizar *kernels* más grandes.

El aspecto modular de este modelo permite obtener una arquitectura divisible en módulos simples y parametizables en cuanto al tamaño de convolución y número de filtros que, apilados uno tras otro dan lugar a las 19 capas de la red neuronal. Quizás, es esta simplicidad la característica por la que este modelo ha sido tan ampliamente utilizado a pesar de lograr el segundo puesto en el ILSVRC de 2014. A pesar de ser una arquitectura simple, su principal defecto es la gran cantidad de parámetros del modelo, más de 130 millones, lo que lo vuelve computacionalmente costoso.

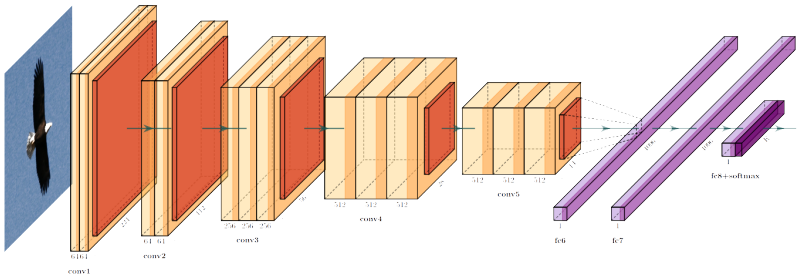


FIGURA 2.4: Arquitectura de VGG-16. Diagrama creado con [13].

2.2.4. Arquitecturas Inception-vX

Inception-v1

En la línea de construir modelos a partir de la concatenación de módulos se encuentra el ganador del ILSVRC de 2014: GoogleNet, también conocido como Inception-v1 [14]. Este modelo sustituye los módulos basados en capas convolucionales por bloques llamados "Inception *blocks*", presentado en la figura 2.5, de forma similar a como la arquitectura *Network in Network* [15] propone sustituir cada capa por micro redes neuronales. Estos bloques contienen filtros de diferentes tamaños, desde 1x1 a 5x5, permitiendo así captar la información a distintas escalas. Además de variar los tamaños de las convoluciones, dentro de un mismo bloque se aplican a distintas resoluciones para luego concatenar al final el resultado de cada camino, logrando así afrontar el problema del aprendizaje de distintas características pertenecientes a una misma categoría pero a distintas resoluciones. Para evitar el aumento excesivo del coste computacional se regulan los tamaños de las entradas a las convoluciones con *kernels* más grandes mediante la aplicación previa de una convolución 1x1. Por otro lado, a diferencia de la arquitectura VGG, Inception-V1 utiliza una operación de reducción de dimensionalidad, llamada *global average pooling*, antes de la primera capa de producto interno, permitiendo así reducir el uso de parámetros a tan solo 4 millones respecto a los más de 130 que requiere VGG. Además, otro concepto importante que introduce esta arquitectura es el uso de funciones de coste auxiliares para así afrontar el problema del *vanishing gradient* añadiendo estas funciones en distintos puntos del camino de la arquitectura.

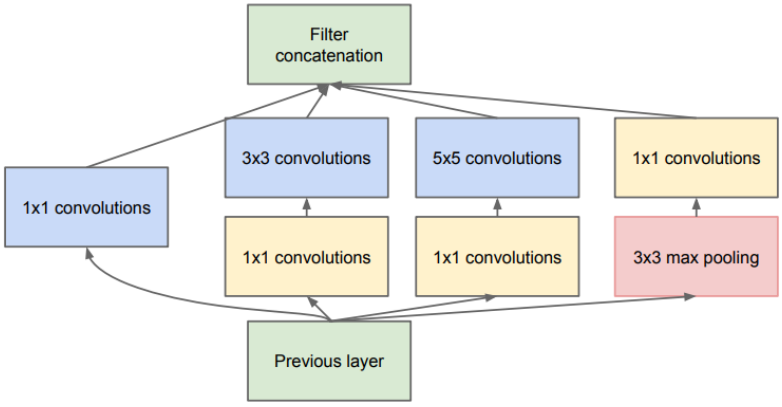


FIGURA 2.5: Arquitectura del Inception *block*. Imagen extraída de [14].

Inception-v2 y v3

Continuando con el trabajo de Inception-v1, los autores proponen en [16] una serie de mejoras para reducir la complejidad computacional a la vez que aumentan la precisión del modelo.

En primer lugar, en lo que respecta a la reducción del coste computacional, como cualquier convolución mayor a 3x3 se puede factorizar en una secuencia de convoluciones 3x3, se propone reemplazar las convoluciones 5x5 de los módulos *Inception* por una concatenación de dos convoluciones 3x3, ahorrando así un 64 % de operaciones. En la figura 2.6 se muestra el diagrama del bloque *Inception* luego de aplicar esta factorización.

Siguiendo con esta factorización, en teoría, cualquier convolución se podría factorizar con dos convoluciones asimétricas: la primera de tamaño $1xn$ y la segunda $nx1$, donde n es el tamaño original del *kernel*. El ahorro aumentaría dramáticamente a medida que n aumenta. En la práctica, los autores afirman que esta modificación no da buenos resultados en las capas tempranas, pero si que resulta beneficioso en las intermedias, encontrando así buenos resultados para valores de $n = 7$. En la figura 2.7 se muestra el bloque de *Inception* resultante de aplicar esta factorización.

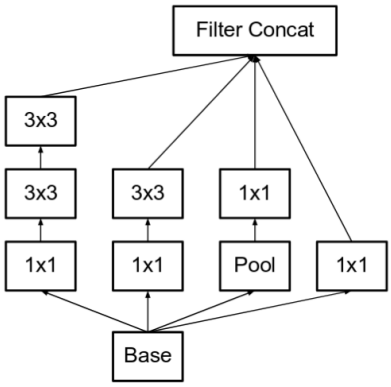


FIGURA 2.6: Diagrama del bloque *Inception* sustituyendo la convolución 5x5 de la rama de la izquierda por dos convoluciones 3x3. Imagen extraída de [16].

Por otro lado, en lo que respecta a la mejora de precisión, se propone reducir lo que los autores denominan "*representational bottleneck*", esto es, la reducción drástica de dimensiones, pues, se intuye que ello puede provocar pérdida de información. Para ello, se propone modificar el bloque de *Inception* de las últimas capas expandiendo el número de filtros de salida tal y como se muestra en la figura 2.8

Estas 3 modificaciones al bloque *Inception* se aplican en conjunto a la arquitectura original, modificando los primeros 3 bloques con la factorización 3x3 de la figura 2.6, los siguientes 5 con la factorización *nxn* de la figura 2.7 y los últimos 2 con la expansión de la figura 2.8. Esta nueva arquitectura será nombrada Inception-V2.

En el mismo artículo se proponen las siguientes modificaciones para mejorar aún más la precisión del modelo:

- Aplicar la operación *Batch Norm* [17] en los clasificadores auxiliares.
- Factorizar las convoluciones 7x7.
- Utilizar el optimizador RMSProp [18].
- Regularizar utilizando la técnica de suavizado de etiquetas.

La aplicación de todas estas mejoras a las ya añadidas consigue mejorar aún más la precisión, dando lugar así al modelo Inception-V3.

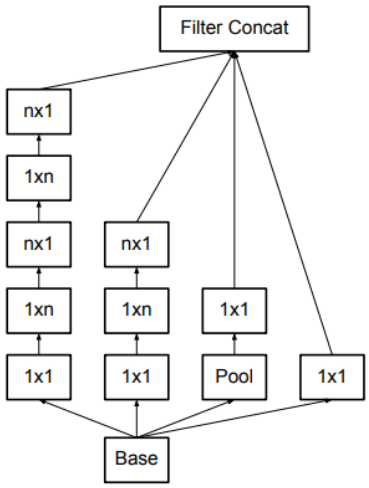


FIGURA 2.7: Diagrama del bloque *Inception* luego de factorizar la convolución nxn. Imagen extraída de [16].

Inception-v4

Inception-v4 se propone en el trabajo [19] con la intención de simplificar y uniformizar la arquitectura de Inception.

Concretamente, se propone:

- Modificar las operaciones previas a la concatenación de módulos *Inception*. A estas operaciones, los autores le llaman "*stem*".
- Simplificar los bloques *Inception* estableciendo de forma unívoca distintos bloques para cada tamaño de cuadrícula. A estos bloques se les llamará con las letras A, B y C.
- Diferenciar los bloques donde se realizan una reducción espacial. Las versiones anteriores no realizaban esta diferenciación, pues, implementaban la funcionalidad en diferentes etapas del modelo.

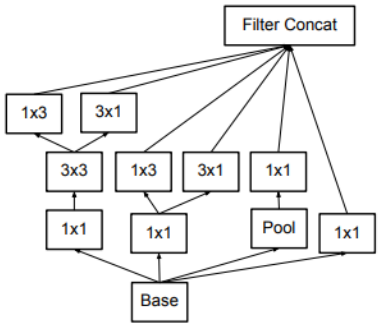


FIGURA 2.8: Diagrama del bloque *Inception* expandiendo los filtros de salida. Imagen extraída de [16].

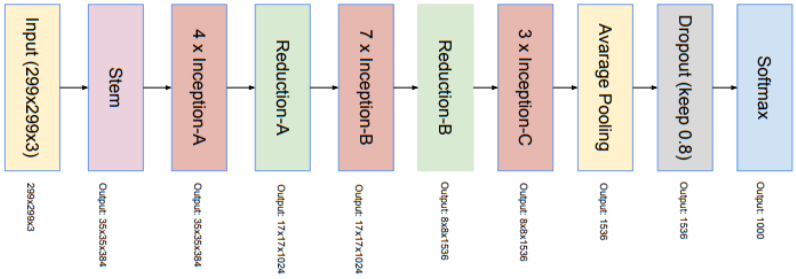


FIGURA 2.9: Diagrama de Inception-v4. Imagen extraída de [19].

En la figura 2.9 se muestra la arquitectura de Inception-v4. El bloque *stem* se muestra en la figura 2.10 mientras que los diferentes módulos de Inception y los bloques de reducción se muestran en las figuras 2.11 y 2.12 respectivamente.

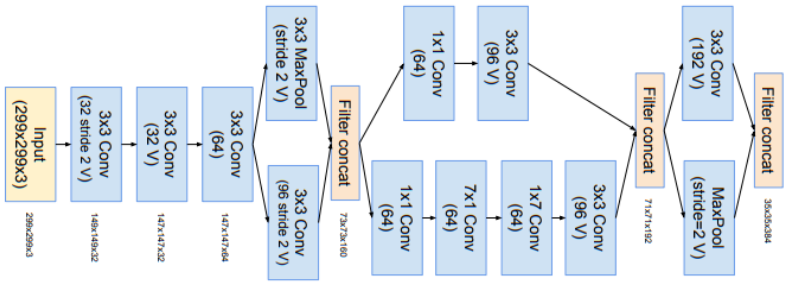
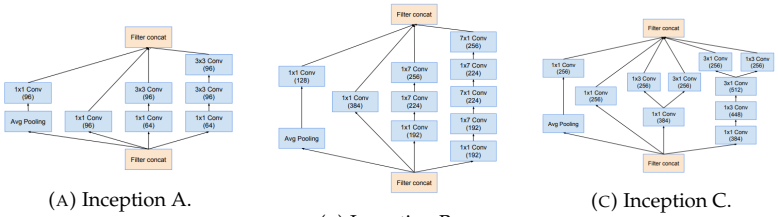


FIGURA 2.10: Diagrama del *stem* de Inception-v4. Imagen extraída de [19].

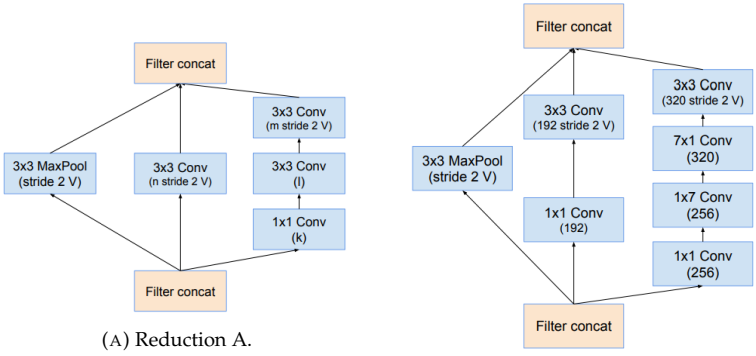


(A) Inception A.

(B) Inception B.

(C) Inception C.

FIGURA 2.11: Módulos Inception. Imagen extraída de [19].



(A) Reduction A.

(B) Reduction B.

FIGURA 2.12: Módulos de reducción. Imagen extraída de [19].

2.2.5. Arquitecturas residuales

ResNet

Tras el éxito de las arquitecturas VGG e Inception, queda demostrado que modelos más profundos son capaces de extraer mejor las características clave y por ende lograr mejores resultados a la hora de resolver el problema de clasificación. En esta línea de aumentar la profundidad de los modelos se enmarca el ganador del ILSVRC de 2015: ResNet [20]. El problema de entrenar modelos profundos era bien conocido en el momento, por lo que, arquitecturas como Inception-V1 desarrollaron métodos para sortear esta dificultad. No obstante, entrenar modelos extremadamente profundos seguía siendo todo un desafío. Finalmente, con la entrada de ResNet se desarrolla un módulo con conexiones residuales permitiendo así mejor flujo del gradiente logrando así entrenar modelos muy profundos.

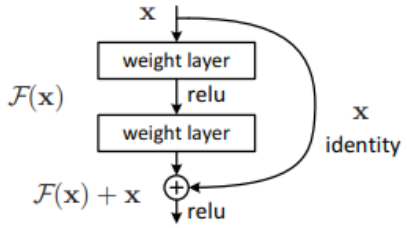


FIGURA 2.13: Modulo de conexión residual. Imagen extraída de [20].

En la figura 2.13 se muestra el diagrama del módulo con conexión residual. Este sigue la ecuación:

$$y = ReLU(\mathcal{F}(x) + x) \tag{2.5}$$

Donde x e y son las entradas y salidas al módulo respectivamente y $\mathcal{F}(x)$ es la función que convoluciona la entrada con una o mas capas convolucionales, para el caso representado en el diagrama:

$$\mathcal{F}(x) = ReLU(x * W_1) * W_2 \tag{2.6}$$

Donde W_1 y W_2 son las matrices de pesos de las dos convoluciones y $*$ es el operador de convolución. En caso de que el número de filtros de $\mathcal{F}(x)$ sea diferente al de x, se aplicaría una convolución a la conexión residual para lograr

obtener el mismo número de filtros. Por tanto, la ecuación 2.5 se reescribiría como:

$$y = ReLu(\mathcal{F}(x) + x * W_r) \tag{2.7}$$

Donde W_r son los pesos necesarios para lograr obtener el mismo número de mapas de características que $\mathcal{F}(x)$ para así realizar la suma punto a punto. Gracias a estas conexiones residuales se vuelve posible entrenar modelos muy profundos, pues, el uso de estas conexiones que "se saltan" capas convolucionales facilita la propagación del gradiente hacia las primeras capas del modelo, no solo evitando el problema del *vanishing gradient* sino que también mejorando la convergencia.

Esta técnica logró la entrada del ganador ResNet-152, un modelo de 152 capas, 20 y 8 veces más profundo que AlexnNet y VGG respectivamente pero con una complejidad computacional inferior a sus predecesores. De gran importancia es la composición de este modelo ganador, pues, a excepción de la primera convolución con *stride* superior a 1 (tamaño del salto entre posiciones que se muestrea la imagen de entrada con el *kernel* de convolución) seguida de un *max-pool*, es una concatenación de módulos de conexiones residuales terminada en un *average pooling* de mapas de características de tamaño 32 veces inferior a la imagen original, con un clasificador basado en un producto interno al final. Gracias a esta modularidad, diferentes configuraciones son posibles desde 18 capas, permitiendo así adaptar el modelo al problema en cuestión. En la figura 2.14 se muestra la tabla de configuraciones de arquitecturas ResNet propuesta en [20].

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

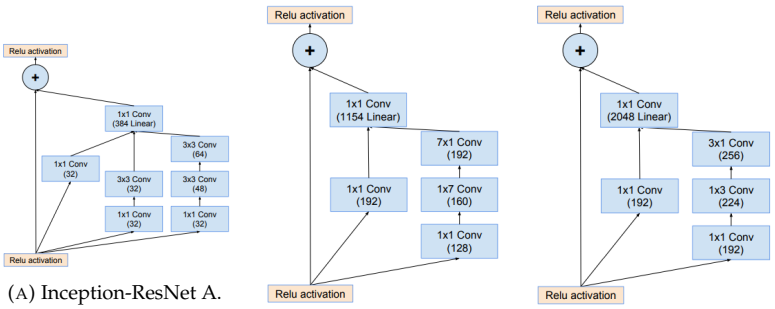
FIGURA 2.14: Diferentes arquitecturas ResNet. Imagen extraída de [20].

2.2.6. Composición de arquitecturas

Inception-ResNet

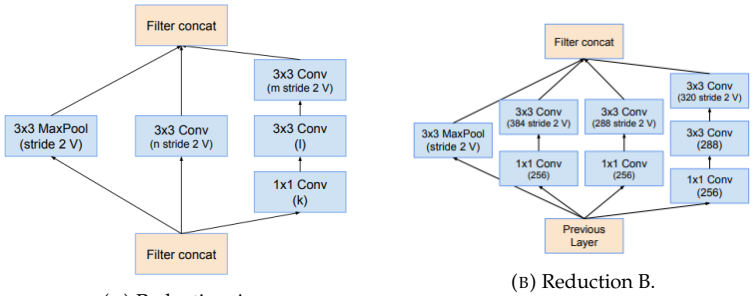
En el mismo trabajo donde se presenta Inception-v4 también se explora la posibilidad de utilizar el éxito de las conexiones residuales en la arquitectura Inception, dando lugar así a Inception-ResNet.

Esta arquitectura sigue la misma idea de Inception-V4: homogeneizar los módulos Inception dividiéndolos en 3 tipos (figura 2.15), diferenciarlos de los bloques de reducción (figura 2.16) y modificar el *stem* (figura 2.10 y 2.17) de la arquitectura Inception original.



(A) Inception-ResNet A. (B) Inception-ResNet B. (C) Inception-ResNet C.

FIGURA 2.15: Módulos Inception-ResNet. Imagen extraída de [19].



(A) Reduction A. (B) Reduction B.

FIGURA 2.16: Módulos de reducción. Imagen extraída de [19].

Finalmente, estos módulos se componen según la figura 2.18 para formar la arquitectura Inception-ResNet. Esta arquitectura se presenta en dos versiones: Inception-ResNet-v1 e Inception-ResNet-v2. La única diferencia entre

ambas es la arquitectura del módulo *stem* y los hiperparámetros. En cuanto al módulo *stem*, Inception-ResNet-v2 2.10 e Inception-V4 utilizan 2.10 mientras que Inception-ResNet-v2 utiliza 2.17. En cuanto a hiperparámetros, estos difieren principalmente en número de filtros por capa.

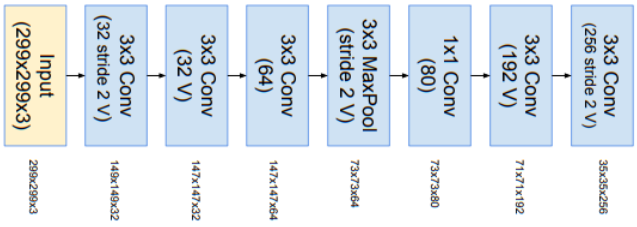


FIGURA 2.17: Diagrama del *stem* de Inception-Resnet-v1. Imagen extraída de [19].

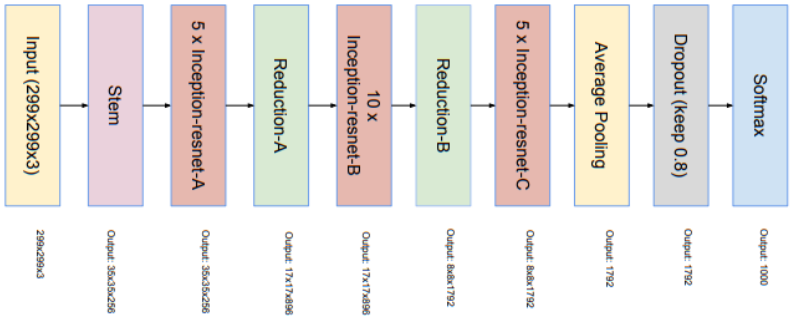


FIGURA 2.18: Diagrama de Inception-ResNet. Imagen extraída de [19].

ResNext

Tras el éxito de ResNet, quedó patente la posibilidad de obtener gran precisión con un modelo profundo correctamente diseñado para permitir su entrenamiento. Para lograr aún mejor precisión en el problema de clasificación se podría o bien aumentar aún más la profundidad de los modelos o bien aumentar el "ancho", es decir, aumentar el tamaño de cada capa. Cualquiera de estas dos modificaciones implica un aumento en coste computacional. En lugar de ello, Xi et al [21] proponen una arquitectura diferente que logra

aumentar la precisión a la vez que reduce la complejidad y el número de parámetros del modelo.

Esta nueva arquitectura, llamada ResNext hereda componentes de las arquitecturas VGG, Inception y ResNet. De VGG sigue el camino de implementar una arquitectura a base de concatenación de capas, o módulos, parametrizables. De Inception sigue la estrategias de dividir la entrada en múltiples caminos, aplicar una transformación a cada camino para finalmente unir los resultados de cada uno de ellos. Y, finalmente, de ResNet, utiliza su principal innovación: las conexiones residuales entre capa y capa.

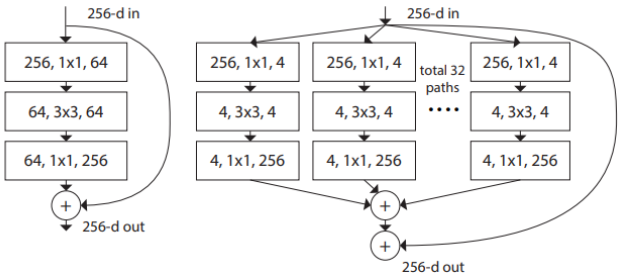


FIGURA 2.19: Diagrama de bloque ResNext de cardinalidad 32 (derecha) en comparación con un bloque ResNet (izquierda). Imagen extraída de [21].

Este modelo logra el segundo puesto en el ILSVRC de 2016.

SENet

Hu et al [22] proponen el uso del bloque *Squeeze-and-Excitation* (SE) en arquitecturas del estado del arte.

El objetivo del bloque SE es ponderar cada mapa de característica de la capa sobre la que se aplica por un peso en el rango $[0, 1]$, de modo que se pueda diferenciar qué canales de dicha capa se debe de dar más importancia. La forma de obtener el peso por el que ponderar cada mapa de característica es el resultado de la agregación global de cada canal seguido de una serie de convoluciones 1x1 terminando en sigmoide para normalizar en el rango deseado; obteniendo así una medida de la importancia de cada canal a nivel global.

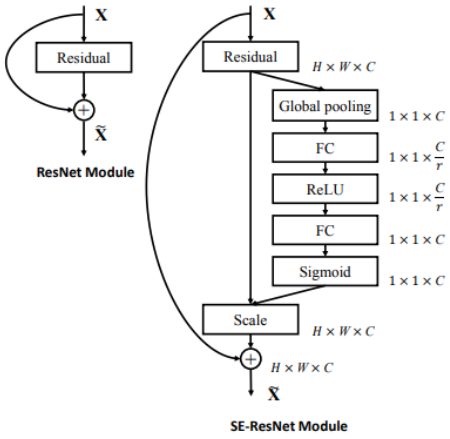


FIGURA 2.20: Aplicación del bloque SE sobre la arquitectura ResNet. Imagen extraída de [22].

En la figura 2.20 se muestra un ejemplo de un bloque SE aplicado sobre una arquitectura ResNet.

La principal ventaja de este bloque es su fácil implementación en distintas arquitecturas del estado del arte, permitiendo así lograr una ligera mejora en la precisión del modelo con un coste marginal en complejidad computacional.

2.3. Interpretabilidad

Como se ha visto en la sección 2.2, existen diversos modelos capaces de obtener muy buenos resultados en la tarea de clasificación de imágenes. Estos modelos son considerados "cajas negras", pues, si bien son capaces de dar una respuesta a un problema, carecen de una justificación apropiada de por qué han llegado a tal conclusión. Esta carencia es especialmente importante cuando no es suficiente dar una clasificación, sino que su justificación es requerida. Un ejemplo práctico puede ser el ámbito de la detección de signos de retinopatía diabética a partir de imágenes de fondo de ojo. En este ámbito, los signos de dicha enfermedad pueden ser tan sutiles que pudiera darse la circunstancia que el modelo sea capaz de detectar signos que un humano pase por alto; por lo que, sin una debida justificación, una mera clasificación podría ser insuficiente.

A pesar de que el problema de la interpretabilidad y justificación es bien conocido, no existe aún una solución global y generalizable; por lo que se analizarán las distintas aproximaciones del estado del arte. El objetivo de estas aproximaciones es, para cada posible clasificación, dar un mapa de "calor" de la importancia que tiene cada zona de la imagen de entrada. Continuando con el ejemplo de la detección de retinopatía diabética, esta importancia podría interpretarse como la presencia de signos de dicha enfermedad.

Estas aproximaciones pueden dividirse en 3 grupos según el tipo de análisis que realizan: análisis mediante oclusiones, análisis de mapas de activaciones y análisis de gradientes.

2.3.1. Análisis mediante oclusiones

El análisis mediante oclusiones se estudia en el trabajo [23] donde se propone el análisis de los mapas de característica internos a los modelos del estado del arte. Este análisis viene impulsado por el hecho de que hasta el momento (2013, época del éxito de AlexNet) las arquitecturas se realizaban a base de prueba y error, sin saber exactamente la razón de una mejora o empeoramiento de los resultados.

Para realizar el análisis se propone una arquitectura deconvolucional multicapa (*DeconvNet*) que será finalmente conocida como ZFNet. El objetivo de ZFNet es visualizar la actividad de las activaciones internas de los modelos del momento tales como AlexNet. Para ello, ZFNet invierte el orden de las convoluciones y operaciones de *pooling*, permitiendo así, proyectar la salida

de una capa convolucional a una imagen con patrones visualmente perceptibles, dando así la posibilidad de analizar la representación interna de cada característica aprendida.

En la práctica, esta inversión de operaciones se realiza con convoluciones traspuestas y *un-pooling* utilizando los índices de las operaciones de *pooling* realizadas. En la figura 2.21 se muestra el diagrama de aplicación de una capa deconvolucional a una capa convolucional así como la aplicación de los índices obtenidos de la operación de *pooling* a la operación de *un-pooling*.

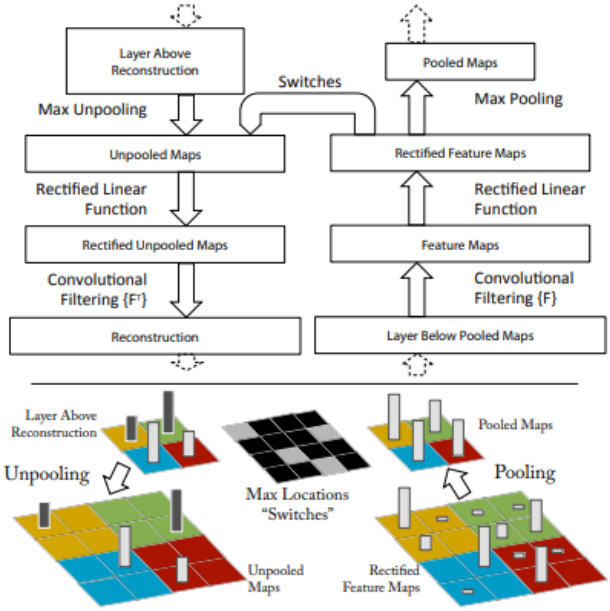


FIGURA 2.21: Capa deconvolucional (izquierda) aplicada a una capa convolucional (derecha). En la parte de abajo se muestra el diagrama de aplicación de la técnica de *pool-unpool*. Imagen extraída de [23].

Esta idea de visualizar las características internas fue validada experimentalmente sobre AlexNet. Se demostró que tan solo unas pocas neuronas estaban realmente activas. Muchas de las neuronas "muertas" se encontraban en las primeras y segundas capas. Además, detectaron que muchas de las características extraídas de la segunda capa mostraban artefactos de submuestreo

o *aliasing*. En base a estos análisis se propuso una modificación de la topología de la arquitectura. Este ajuste permitió obtener un mejor resultado en términos de precisión, sugiriendo así que la capacidad de visualizar el estado interno de un modelo permitiría guiar más directamente las futuras modificaciones.

En la figura 2.22 se muestra un ejemplo de visualización de AlexNet. Se puede apreciar como las características de las primeras capas son de "bajo nivel", pues, se asemejan al resultado de filtros de bordes o altas frecuencias, mientras que a las últimas capas es posible visualizar características de "alto nivel" tal como letras.

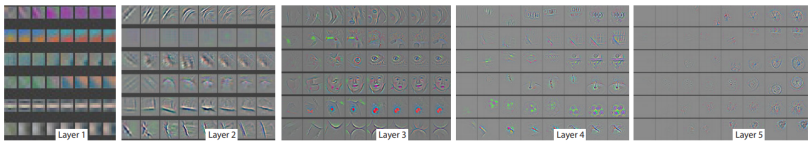


FIGURA 2.22: Visualización de subconjuntos de características internas a cada capa del modelo AlexNet. Imagen extraída de [23].

En este mismo trabajo, se propone además analizar el comportamiento tanto del clasificador final como del estado interno del modelo en función de qué zonas de la imagen se ocluyen. En la figura 2.23 se muestra un ejemplo de aplicación de esta técnica. En la primera columna (a) se muestra la imagen de entrada con un ejemplo de parche de oclusión. En la segunda y tercera columna (b y c) se muestra como cambia el estado interno del modelo en función de la posición del parche de oclusión mientras que en las cuarta y quinta columnas (d y e) se muestra el resultado del clasificador. De importancia es como varía el resultado del clasificador (columna d) en función de la posición del parche, pues, las zonas que más penalizan la correcta predicción pueden utilizarse para interpretar qué zonas de la imagen son más importantes para el modelo y, en consecuencia, dónde se encuentra la zona de interés que justifica la clasificación dada por el modelo.

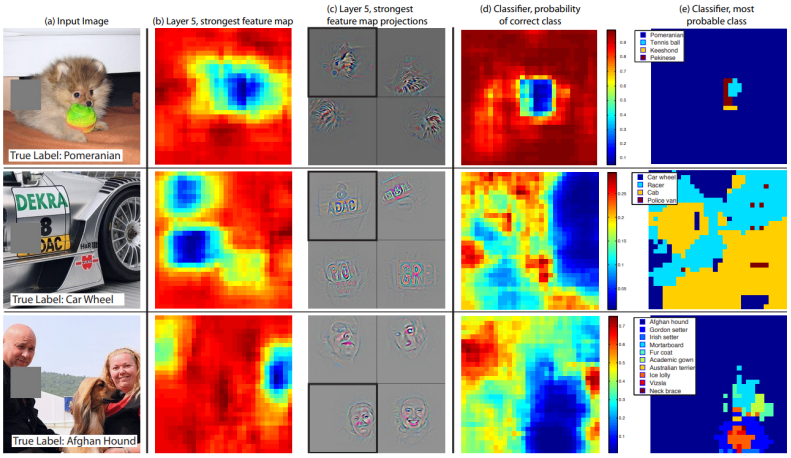


FIGURA 2.23: Análisis de oclusiones y sus efectos en el clasificador y características internas. Imagen extraída de [23]

2.3.2. Análisis de gradientes

El análisis de gradientes consiste en estudiar el comportamiento de los gradientes para así obtener una visualización del comportamiento del modelo.

Para este análisis, se utiliza el vector de probabilidades no normalizadas, conocido como "*logits*". Intuitivamente, el gradiente de los *logits* de una clase en concreto respecto a la imagen de entrada representa en qué medida afecta un cambio en cada píxel de la imagen de entrada a la identificación de esa imagen como dicha clase. Por tanto, un gradiente positivo indicará que un aumento en la intensidad del píxel en cuestión afectará positivamente, mientras que un gradiente negativo indica lo contrario. Siguiendo esta idea, Springenberg et al [24] proponen, no solo la utilización de dichos gradientes para identificar las zonas de la imagen que justifican la clasificación, sino que también proponen un método alternativo de retropropagación para mejorar la localización de las características de interés para la clase que se quiere identificar. Esta modificación consiste en, durante el cálculo de la retropropagación, permitir el flujo del gradiente únicamente en caso que este sea positivo, acuñando así el término de "retropropagación guiada". En la práctica, esta modificación se traduce en modificar la operación de gradiente de la función de activación.

Sea $f^l(x,y,k)$ la posición (x,y) del mapa de características k de la capa l y sea la capa $l+1$ una operación de ReLU, el gradiente de los *logits* f^c de la clase de interés c respecto f^l viene dado por la ecuación:

$$\frac{\delta f^c}{\delta f^l(x,y,k)} = \begin{cases} \frac{\delta f^c}{\delta f^{l+1}(x,y,k)} & \text{si} f^l(x,y,k) > 0 \\ 0 & \text{en otro caso} \end{cases} \tag{2.8}$$

La ecuación 2.8 muestra el cómputo del gradiente a través de la función de activación ReLU. Para lograr el comportamiento deseado, el de permitir el flujo del gradiente solo en aquellas posiciones que logren un gradiente positivo, se modificará la ecuación de gradiente del ReLU por la siguiente:

$$\frac{\delta f^c}{\delta f^l(x,y,k)} = \begin{cases} \text{máx}(\frac{\delta f^c}{\delta f^{l+1}(x,y,k)}, 0) & \text{si} f^l(x,y,k) > 0 \\ 0 & \text{en otro caso} \end{cases} \tag{2.9}$$

De este modo, solo se transmitirán los gradientes que influyen positivamente en la clase que se quiere detectar. En la figura 2.24 se muestra un diagrama ejemplificando cómo influye esta modificación en el gradiente calculado comparándolo con el cálculo real del gradiente y con ZFNet explicado en la sección 2.3.1. En la figura 2.25 se muestra una comparativa cualitativa de estos métodos.

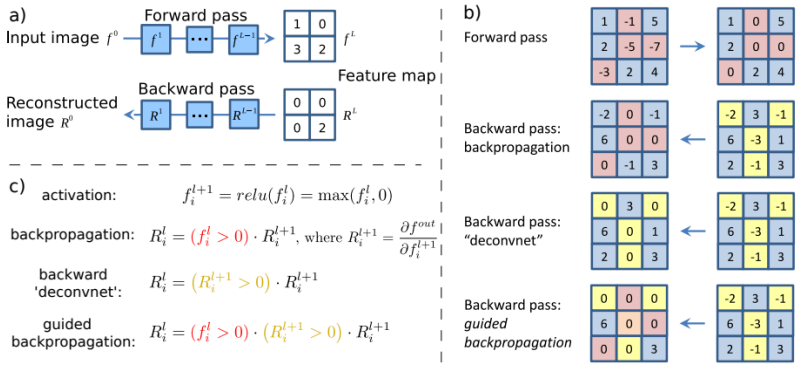
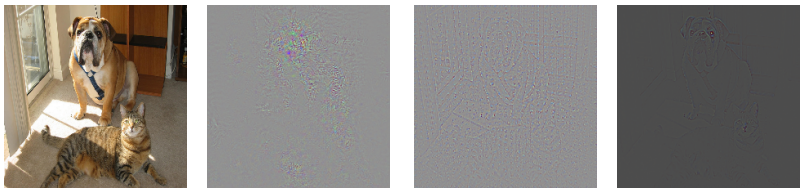


FIGURA 2.24: Diagrama de cómputo de retro-propagación guiada. Imagen extraída de [24].



(A) Imagen de entrada. (B) Retro-propagación sin modificación. (C) Resultado de ZF-Net. (D) Retro-propagación guiada.

FIGURA 2.25: Comparativa de métodos de retro-propagación.

Este método cuenta con la característica de obtener como resultado imágenes de alta resolución. No obstante, una de sus principales desventajas es la necesidad de modificar el comportamiento del cálculo del gradiente. En esta misma linea y afrontando ese problema, surge el método de integración de gradientes [25]. Este método consiste en, primero, establecer una entrada neutra, llamada línea de base o "*baseline*" (una imagen en negro por ejemplo), y luego calcular la integral del gradiente de los *logits* respecto la entrada a lo largo de un camino que va desde el *baseline* a la imagen que se quiere analizar.

Sea $X \in \mathbb{R}^{W,H,3}$ la imagen que se quiere analizar y $X' \in \mathbb{R}^{W,H,3}$ su *baseline* con W y H como el tamaño de las dimensiones espaciales. $f^c(X)$ será el *logit* obtenido para la entrada X.

$$Ig(x,y,k) = \big(X(x,y,k) - X'(x,y,k)\big) \int_{\alpha=0}^{1} \frac{\delta f^c(X' + \alpha(X - X'))}{\delta X(x,y,k)} d\alpha \quad (2.10)$$

En la ecuación 2.10 se muestra la fórmula de gradientes integrados, donde $Ig(x,y,k)$ es el gradiente integrado para la imagen X con *baseline* X' en la posición (x,y,k) y $\alpha \in [0,1]$ el camino lineal entre X y X'.

2.3.3. Análisis de mapas de activaciones

El análisis de mapa de activaciones consiste en inspeccionar los mapas de activación de la penúltima capa de la red neuronal, justo la capa previa a las operaciones de producto interno que preceden a la clasificación final. Con esta inspección se genera un mapa de activación por clases, obteniendo así como resultado un mapa de calor de qué zona de la imagen de entrada contribuye en mayor medida a la clasificación de dicha imagen como la clase

en cuestión. Pionero en este campo es el trabajo de Zhou et al [26] donde se acuña el término *Class Activation Mapping*, posteriormente conocido como (CAM).

Para generar el CAM, primero ha de utilizarse una red neuronal del estado del arte, como puede ser una de la familia ResNet, para poder clasificar una imagen. Para el caso concreto de la arquitectura ResNet, se tiene $f(x,y,k)$ como la activación en la posición espacial (x,y) del mapa de características k de la última capa convolucional. Las siguientes operaciones del modelo para obtener la predicción no normalizada (*logit*) para clasificar la imagen como perteneciente a la clase c son: un *global average pooling*: $f_{avg}(k) = \frac{1}{N}\sum_{x,y} f(x,y,k)$, con N como el número de elementos espaciales, seguido de la capa de producto interno: $f^c = \sum_{k=0}^{K} f_{avg}(k) w^c(K)$ donde $w^c \in \mathbb{R}^K$ es el vector de pesos de la clase c para cada uno de los K mapas de activación. Uniendo estas últimas operaciones en una sola ecuación se obtiene:

$$f^c = \sum_{k=0}^{K} \left(\frac{1}{N} \sum_{x,y} f(x,y,k) \right) w^c(k) \tag{2.11}$$

Siguiendo las propiedades elementales de la suma y multiplicación, se obtiene:

$$f_{CAM}(x,y) = \sum_{k=0}^{K} f(x,y,k) w^c(k) \tag{2.12}$$

$$f^c = \frac{1}{N} \sum_{x,y} f_{CAM}(x,y) \tag{2.13}$$

De este modo, $f_{CAM}(x,y)$ representa el CAM para la posición (x,y) y contará con la misma resolución espacial que los mapas de activaciones del a última capa convolucional.

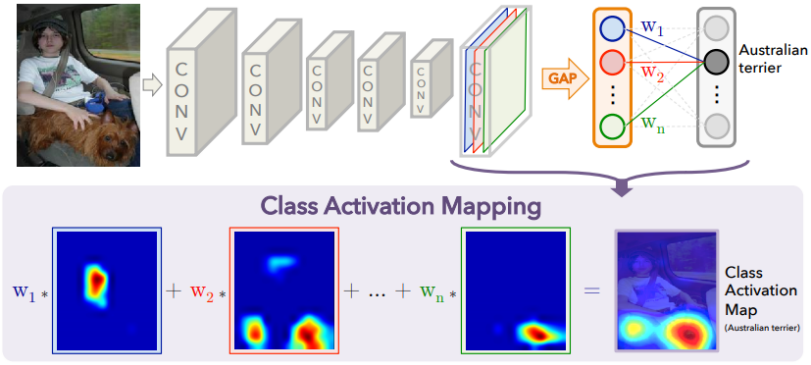


FIGURA 2.26: Diagrama de cómputo de CAM. Imagen extraída de [26].

En la figura 2.26 se muestra el diagrama del calculo de CAM así como el ejemplo de visualización para una clase concreta. En el diagrama, los pesos $(w_1, \ldots, w_n)$ son los pesos aprendidos durante el entrenamiento para la clase "Australian terrier". Como puede apreciarse de la imagen CAM resultante, esta ilumina las zonas donde efectivamente se encuentra el animal en cuestión, explicando así la razón por la que se llega a la clasificación de "Australian terrier".

A pesar de lograr una correcta identificación de las zonas de interés para la clasificación, este método tiene dos grandes desventajas. La primera es la resolución espacial del CAM pues esta es muy baja. Por ejemplo, para el caso concreto de una arquitectura ResNet será de apenas $7x7$. La segunda, esta metodología solo vale para algunas arquitecturas en concreto, pues extrae los pesos a utilizar para la generación del CAM directamente de los pesos aprendidos durante el entrenamiento.

En esta misma línea de analizar mapas de activación surge el método *Gradient-weighted Class Activation Mapping* (Grad-CAM) [27]. Este método difiere con el método CAM principalmente en la forma de calcular los pesos que ponderan cada mapa de características. El método Grad-CAM propone extraer los pesos a utilizar en la ponderación de los gradientes. Siguiendo la notación anterior, sea f^c los *logits* que se obtienen para la clase c, los pesos $w^c(k)$ se obtienen calculando la media global espacial de los gradientes de los *logits* f^c respecto los mapas de activación $f(x, y, k)$, siguiendo la ecuación:

$$w^c(k) = \frac{1}{N}\sum_{x,y}\frac{\delta f^c}{\delta f(x,y,k)} \tag{2.14}$$

Intuitivamente, estos pesos obtenidos representarán un medida de la importancia de cada mapa de característica para la clasificación final. Con estos pesos, se repite la misma operación que realiza el método CAM, con la salvedad que en este caso se aplica una rectificación lineal para evitar las zonas donde se encuentran los contra-ejemplos.

Además de proponer un método diferente para el cálculo de los pesos, también propone una forma de obtener mapas de localización en alta resolución. Para ello, se calcula, además del Grad-CAM, los gradientes de f^c respecto la imagen de entrada siguiendo retro-propagación guiada [24] tal y como se explica en la sección 2.3.2. Con estos resultados, se redimensiona mediante interpolación bilineal el Grad-CAM a la resolución de imagen original y se multiplica punto a punto junto con el gradiente respecto la imagen, y con la imagen original. En la figura 2.27 se muestra el diagrama completo de cálculo del Grad-CAM.

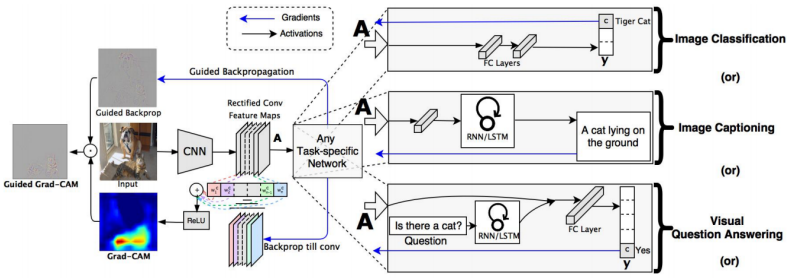


FIGURA 2.27: Diagrama de cómputo de Grad-CAM. Imagen extraída de [27].

De este modo, Grad-CAM sortea la dificultad de tener que utilizar arquitecturas en concreto, al poder extraer los pesos directamente de los gradientes, a la vez que propone un método de visualizar en mayor resolución el resultado.

2.4. Segmentación

El problema de la segmentación consiste, en lugar de asignar una etiqueta a una imagen, en asignar una etiqueta a cada uno de los píxeles de la imagen. Este problema es de interés para la presente tesis no tanto por el problema de segmentación en sí, sino por las arquitecturas de redes neuronales que se utilizan para su resolución.

Como se ha visto en secciones anteriores, una red neuronal de clasificación es un extractor de características seguido de un clasificador. Consiste en, a grandes rasgos, una sucesión de diferentes capas compuestas por distintas operaciones (convoluciones, normalizaciones, funciones de activación, de *pooling*, etc...) que poco a poco van dividiendo la imagen en las características a extraer a la vez que van reduciendo su resolución espacial para finalizar en el clasificador final que transformará el último vector de mapas de características en un vector de probabilidades sin normalizar, también llamado *logits*, obteniendo luego mediante la operación de *softmax* las probabilidades normalizadas de pertenencia a cada una de las clases preestablecidas.

Como se ha visto en la sección 2.3.3, un análisis del último vector de mapas de características puede dar la localización de la evidencia de cada una de las clases preestablecidas dentro de la imagen. Esto demuestra que sería posible obtener una segmentación de la imagen utilizando este último vector de características si se contara con un conjunto de datos debidamente etiquetado y entrenara para ello.

Utilizar únicamente el último vector de mapas de características daría como resultado una segmentación con resolución espacial igual a la resolución espacial de este último vector, $7x7$ en el caso de una ResNet por ejemplo. Para afrontar este problema se han propuesto distintos tipos de arquitecturas orientadas a la segmentación de imágenes. El diseño básico de estas arquitecturas consiste en un codificador-decodificador: el codificador es el encargado de extraer las características necesarias de la imagen, mientras que el decodificador será aquel que, utilizando las características extraídas por el codificador, recupere una segmentación con la resolución de la imagen original. En la práctica, el codificador es una red neuronal de clasificación sin la última operación de clasificación, mientras que el decodificador es donde difieren principalmente los distintos modelos de segmentación.

De interés son los distintos tipos de decodificadores del estado del arte, pues

son los que permiten recuperar toda la resolución original y no son exclusivos para el problema de segmentación: variando la función de costes y el conjunto de datos utilizado durante el entrenamiento se pueden convertir en modelos que resuelvan otro tipo de problemas que requieran una solución pixel a pixel.

En la presente sección se referenciarán algunas de las arquitecturas de decodificación más influyentes del estado del arte.

2.4.1. *Fully Convolutional Network*

La arquitectura *Fully Convolutional Networks* (FCN) [28] aparece en uno de los primeros trabajos donde se utilizan redes neuronales convolucionales para lograr una segmentación. Este modelo transforma redes neuronales utilizadas en clasificación en modelos capaces de obtener una segmentación como resultado. Para ello, modifican arquitecturas como la VGG eliminando el clasificador final y agregando una capa de sobremuestreo (término mejor conocido como "*upsampling*") mediante deconvoluciones (también llamadas *backwards convolutions* o convoluciones traspuestas). Estas convoluciones de *upsampling* no son más que convoluciones ordinarias con un *stride* fraccionario. Como resultado, se obtiene un modelo entrenable punto a punto capaz de obtener predicciones píxel a píxel en la misma resolución de la imagen de entrada. En la figura 2.28 se muestra el diagrama de la arquitectura propuesta.

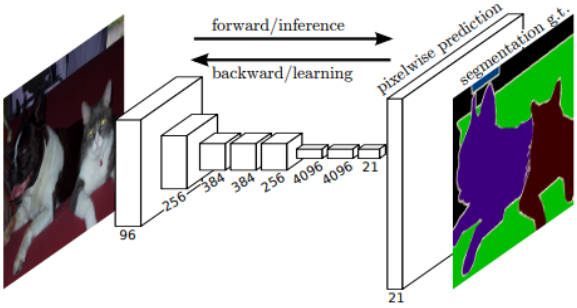


FIGURA 2.28: Diagrama de FCN. Imagen extraída de [28].

Si bien esta arquitectura es capaz de segmentar, el *upsampling* se realiza con

un factor de 32, lo que provoca un resultado final poco refinado y con artefactos de haber realizado el cómputo en baja resolución. Como solución a este problema, se propone utilizar los mapas de características intermedios del modelo utilizado como "codificador" para así retener la información en alta resolución.

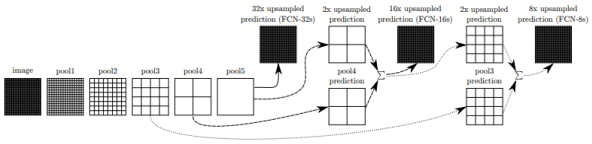


FIGURA 2.29: Arquitecturas de FCN-8s, FCN-16s y FCN-32s. Imagen extraída de [28].

En la imagen 2.29 se muestra el diagrama de como se combina la información de los mapas intermedios sobre una arquitectura VGG. Se muestran 3 posibilidades: FCN-32s solo utiliza el último vector de características y hace un *upsampling* de 32. FCN-16s utiliza el último vector de características de las últimas dos resoluciones, aplica un *upsampling* de factor 2 del de menor resolución para poder combinarlos y finalmente realiza un *upsampling* factor 16. Por último, FCN-8s realiza las mismas operaciones con las últimas 3 resoluciones. En la imagen 2.30 se muestra una comparativa cualitativa del resultado de cada uno de estos modelos.

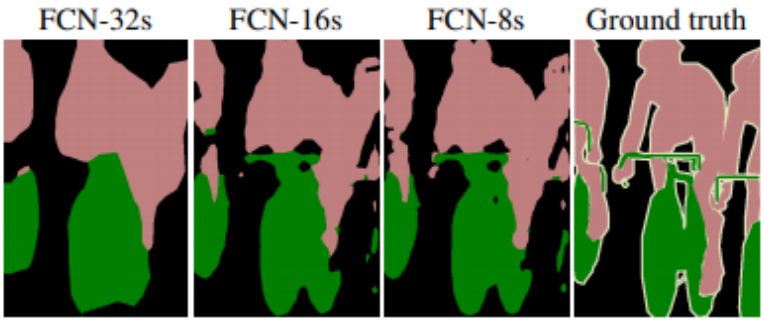


FIGURA 2.30: Comparativa de refinamiento de las diferentes FCN. Imagen extraída de [28].

En el trabajo también se explica otra forma de lograr resultados más refinados, y esta es limitar el número de *poolings*, lo que a su vez aumenta la resolución espacial de los mapas de característica finales. Esta modificación tiene 2 principales inconvenientes: el primero y claro, aumenta el coste computacional. El segundo y de más gravedad, aunque se dispusiera de los recursos computacionales para aplicar dicha modificación, una mayor resolución implicaría menor campo receptivo en los *kernels* de más alto nivel, lo que conllevaría a un peor resultado de clasificación y, por ende, peor segmentación.

2.4.2. U-Net

Inspirada en FCN se desarrolla la arquitectura U-Net[29]. En FCN se continúa la idea de dividir el modelo en dos partes, un codificador (llamado *contracting path*) y un decodificador (*expansive path*). Quizás, una de las aportaciones más importantes de esta arquitectura es el modo en que propone aprovechar la información procedente del codificador para mejorar la calidad de la reconstrucción obtenida por el decodificador.

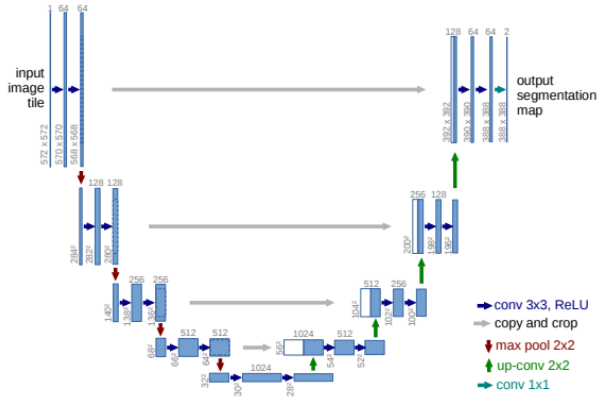


FIGURA 2.31: Arquitectura U-Net. Imagen extraída de [29].

En la figura 2.31 se muestra el diagrama de la arquitectura de U-Net. De esta, cabe destacar el modo en que se realizan las conexiones entre el codificador y decodificador: a diferencia de FCN donde se combinan mediante sumas, U-Net propone concatenar cada resolución del decodificador con su homólogo en el codificador, para luego aplicar una capa convolucional.

De esta arquitectura, por un lado cabe destacar que su codificador no es ningún modelo de clasificación, lo que da como consecuencia el inconveniente de no poder utilizar ningún modelo preentrenado para así facilitar el aprendizaje. Por otro, este codificador es más ligero que los utilizados en otras arquitecturas de clasificación, dando lugar así a un modelo de segmentación más eficiente en comparación con otros como FCN por ejemplo.

2.4.3. Seg-Net

La arquitectura Seg-Net [30] sigue una aproximación diferente a las vistas anteriormente a pesar de coincidir en la filosofía "codificador-decodificador". Esta arquitectura consiste en un codificador del estado del arte en clasificación, VGG por ejemplo, y en añadir un decodificador simétrico que utiliza los índices del *pooling* utilizados por el codificador para realizar el *upsampling* mediante operaciones de *un-pooling*, similar a como se realiza en ZFNet [23].

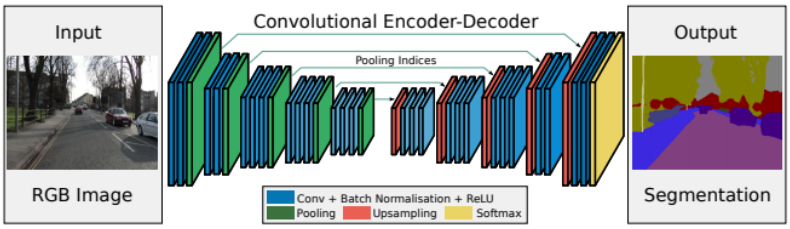


FIGURA 2.32: Arquitectura Seg-Net. Imagen extraída de [30].

En la figura 2.32 se muestra la arquitectura de Seg-Net sobre una VGG-16 a la que se le han eliminado, además del clasificador, las últimas capas de producto interno para reducir el número de parámetros en un factor de x10.

Esta arquitectura cuenta con la ventaja de poder utilizar una arquitectura VGG preentrenada en clasificación y entrenar únicamente el decodificador, reduciendo así el tiempo de entrenamiento y logrando mejores resultados que partiendo desde cero.

El principal inconveniente de este tipo de arquitectura es propiamente su idea principal: la de utilizar los índices de las operaciones de *pooling*, pues,

esto limita este método a aquellos codificadores que realicen sus redimensiones únicamente mediante operaciones de *pooling*; excluyendo así a arquitecturas como ResNet que utilizan convoluciones con *stride* para redimensionar.

2.4.4. PSP-Net

PSP-Net [31] logra el primer puesto en "*ImageNet scene parsing challenge 2016*", marcando un récord de precisión mIoU del 85.4 % sobre PASCALVOC [32] 2012 y 80.2 % sobre Cityscapes [33].

Este modelo consiste en un codificador, ResNet-101 utilizado originalmente, y un decodifcador que utiliza el módulo de *spatial pyramid pooling* [34] para decodificar. Este módulo consiste en redimensionar al último vector de mapa de características con operaciones de *pooling* de distinto *stride*, convolucionar cada una de esas ramas, recuperar su resolución previa y utilizar como entrada a la capa convolucional final la concatenación de cada una de estas ramas y el vector de mapas de características original. De esta forma, se consigue obtener información de distintas escalas y resoluciones para lograr así un mejor resultado de segmentación.

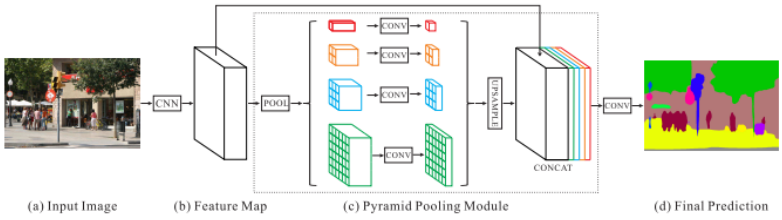


FIGURA 2.33: Arquitectura PSP-Net. Imagen extraída de [31].

En la figura 2.33 se muestra el diagrama de PSP-Net. Como puede apreciarse, la convolución final se realiza en la misma resolución de los mapas de característica finales, que en el caso de una ResNet son de resolución 7x7. En la práctica, PSP-Net utiliza una versión modificada de ResNet, llamada DRN [35] (*Dilated Residual Network*). Esta modificación consiste en eliminar los últimos dos *strides* de la arquitectura original y añadir la dilatación correspondiente a las convoluciones que le siguen a estas capas que originalmente aplicaban una redimensión para así mantener el campo receptivo a la vez

que se logra evitar reducir la resolución. Con esto, se logran mapas de características de 1/8 de resolución original en lugar de 1/32.

2.4.5. RefineNet

La arquitectura RefineNet [36] sigue la idea de utilizar un codificador del estado del arte y un decodificador que sea capaz de extraer una predicción por píxel. Esta arquitectura continúa en la línea de FCN y U-Net de aprovechar múltiples salidas intermedias del codificador para mejorar la predicción del decodificador. A diferencia de los trabajos anteriores, RefineNet propone la utilización del bloque "*RefineNet*" presentado en la figura 2.34.

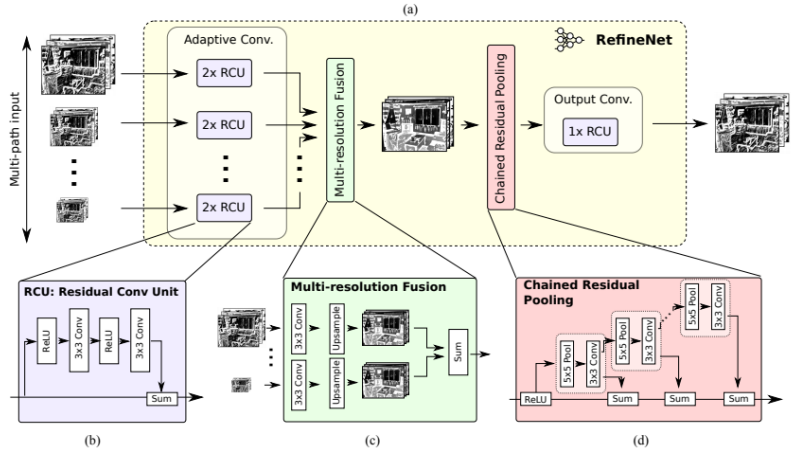


FIGURA 2.34: Arquitectura de un bloque *RefineNet*. Imagen extraída de [36].

La figura 2.34 (a) muestra el diagrama del bloque "*RefineNet*". A cada entrada se le aplican dos *Residual Convolution Unit* (RCU), dos convoluciones con conexiones residuales inspiradas en ResNet. Tras esto se combinan cada una de las resoluciones de entrada en 2.34 (b) con una convolución seguida de una suma para la fusión. Por último, se aplica el bloque *Chained Residual Pooling* (figura 2.34 (d)) para captar el contexto global de cada mapa de características, similar al *spatial pyramid poolong* de PSP-Net.

Este bloque se combina con las diferentes resoluciones de salida tal y como se muestra en la figura 2.35. Aquí, todos los bloques *RefineNet* reciben dos entradas excepto el de menor resolución que tan solo recibe una.

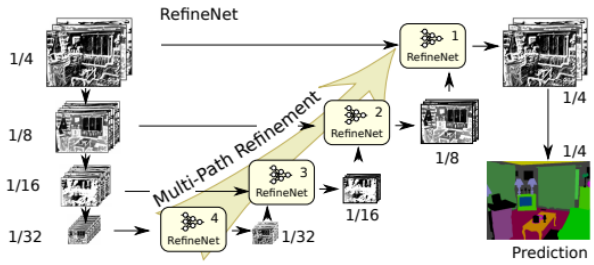


FIGURA 2.35: Arquitectura de RefineNet. Imagen extraída de [36].

2.4.6. DeepLab

DeepLab [37] es una arquitectura de segmentación similar a DRN [35], donde se hace uso de convoluciones dilatadas para preservar la resolución espacial a la vez que se aumenta el campo receptivo para así lograr un resultado de segmentación en mayor resolución.

Originalmente, el modelo de DeepLab produce una salida en resolución 1/16 del tamaño de original, por lo que, tras la predicción, se hace uso de *Conditional Random Fields* (CRF) para refinar esta predicción en baja resolución.

Siguiendo la línea de utilizar convoluciones dilatadas y con el objetivo de mejorar DeepLab, surge DeepLabv3 [38]. Este modelo incorpora un módulo similar al *spatial pyramid pooling* de PSP-Net con la diferencia que, en lugar de realizar operaciones de *pooling*, realiza diferentes convoluciones dilatadas con distintos niveles de dilatación.

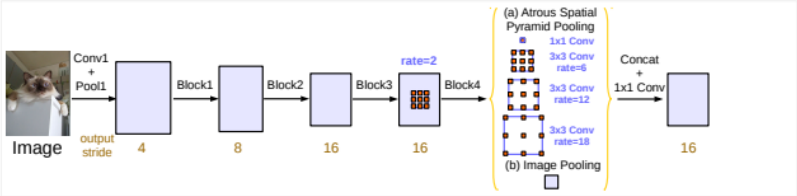


FIGURA 2.36: Arquitectura de DeepLabv3. Imagen extraída de [38].

En la figura 2.36 se muestra el diagrama de DeepLabv3. En ella se aprecia la utilización de convoluciones dilatadas no solo para mantener el campo receptivo al no reducir la resolución espacial, sino que también se muestra como se aplica el módulo *atrous spatial pyramid pooling* (figura 2.36 (a)) para captar las características globales de la imagen. Gracias a la utilización de esta técnica, esta nueva versión de DeepLab consigue mejores resultados que su predecesor incluso sin la utilización de CRFs.

Capítulo 3

Retinopatía diabética

3.1. Introducción

El antiguo presidente electo de la Federación Internacional de la diabetes, George Alberti, dijo en 2003 en una conferencia de prensa en el Palacio de Congresos de París: "Esta (la diabetes) es una de las mayores catástrofes sanitarias que el mundo ha visto. Los costes económicos y sociales de la enfermedad serán intolerables si los gobiernos no despiertan ya y los toman en consideración". Alberti se refería a que es un hecho que el número de pacientes con diabetes tipo 1 se está incrementando mundialmente [39], y que el número de pacientes que padecen la de tipo 2 está teniendo un incremento exponencial [40]. Se estima, que en Europa, aproximadamente 53 millones de adultos tienen diabetes, cifra que se corresponde aproximadamente al 8,1 % de la población adulta, con la previsión de que en 2030 el 9,5 % de la población adulta padecerá diabetes [41]. En España la diabetes afecta a un total de un 13,8 % de la población. El 7,8 % tiene una diabetes conocida mientras que el 6 % restante la padece sin saberlo [42]. En este escenario, los sistemas de salud se enfrentan al reto de evaluar rápidamente a una población muy grande y en crecimiento.

Los estadios avanzados de las principales afecciones oculares de la diabetes, la retinopatía diabética y el edema macular diabético, constituyen a su vez dos de las principales causas de disminución irreversible de la visión en los países desarrollados [43, 44], siendo también una de las principales causas de ceguera [45]. De acuerdo con la Asociación Americana de Diabetes y la Academia Americana de Oftalmología, un examen de fondo de ojo al menos una vez al año en pacientes diabéticos es necesario para identificar posibles lesiones [46, 47]. Esta recomendación se basa en el hecho de que, según la Organización Mundial de la Salud, el tratamiento precoz de la retinopatía

diabética puede reducir el riesgo de pérdida visual severa en más de un 90 % [48]. Dado que el número de oftalmólogos es limitado, la población se enfrenta a un verdadero reto asistencial.

Se han propuesto varias soluciones para mejorar la atención a estos pacientes sin saturar las listas de espera, basadas principalmente en la realización de un cribado previo que detecte de manera precoz a los pacientes susceptibles de tratamiento para poder administrárselo antes de que la pérdida visual sea sustancial e irreversible. Una de ellas ha sido el uso de la teleoftalmología. La fotografía digital del fondo de ojo y su evaluación a distancia se está promoviendo como estrategia para facilitar el acceso al cuidado de la salud en todo el mundo. La teleoftalmología se ha mostrado efectiva y eficiente en coste [49, 50, 51, 52, 53, 54, 55, 56, 57, 58] y está especialmente indicada como un método para extender los cuidados médicos a poblaciones remotas [59, 60, 61, 62].

En las Islas Canarias, debido a su naturaleza ultraperiférica actualmente no existen oftalmólogos de forma habitual en la isla de El Hierro o en La Graciosa, mientras que en La Gomera, La Palma, Fuerteventura y Lanzarote hay un número reducido. Debido a esto, el Servicio Canario de Salud decidió instaurar el programa Retisalud en 2006. Este programa consiste en la toma de fotografías de fondo de ojo (retinografía) de forma periódica a las personas diabéticas en los centros de salud. Las retinografías son valoradas por los médicos de familia en el propio centro de salud, diferenciando entre casos “normales”, “dudosos” y “patológicos”. A los casos normales, aquellos en los que el médico de familia no encuentra ningún signo de retinopatía diabética (RD) y no tienen ningún factor de riesgo acompañante se les planifica una nueva retinografía a los dos años. En caso de que no tengan signos de retinopatía diabética pero si algún factor de riesgo asociado se citan en la cámara no midriática en 1 año. Mientras que, las retinografías patológicos o dudosas se las envía para que sean valoradas por el oftalmólogo, siendo éste el que decida, según el grado de RD, si el paciente debe ser valorado en el oftalmólogo de zona en el centro ambulatorio de especialidades o en el hospital.

Tras más de una década desde la implantación del Programa, se ha logrado incluir a menos del 32 % de los pacientes diabéticos de la comunidad. Una de las principales razones de que este porcentaje sea bueno pero aun esté lejos del 100 % puede ser el trabajo que supone para los médicos de familia la incorporación de la tarea de valoración de retinografías al ya saturado conjunto

de tareas que deben realizar habitualmente. Un sistema de cribado automático facilitaría en gran medida el flujo de pacientes patológicos al oftalmólogo.

En este sentido, las líneas de investigación basadas en inteligencia artificial, que ya han demostrado su utilidad en campos como cardiología [63] o en el diagnóstico del cáncer [64], avanzan cada vez más hacia este necesario sistema automatizado.

Las imágenes de fondo de ojo son especialmente adecuadas para el diagnóstico mediante inteligencia artificial ya que poseen cierta homogeneidad, se puede llegar a un diagnóstico únicamente a través de ellas y los signos de la afección ocular de la diabetes son estructuralmente diferenciables. Es por ello que ya ha habido algún intento de adaptar esta tecnología al campo del cribado de la retinopatía diabética [65]. Si bien estas técnicas suponen un gran avance hacia la mejora del cribado de este tipo de patologías, aún quedan sin resolver muchos problemas relacionados con el mismo, destacando entre ellos la baja precisión o especificidad, la poca capacidad de justificación de los nuevos modelos basados en técnicas de aprendizaje artificial o la dificultad de integración del sistema en su totalidad. Por estas razones, se entiende que, hasta donde llega nuestro conocimiento, hasta la fecha no se ha realizado una implementación real y completa de un sistema automático de diagnóstico.

3.1.1. Retinopatía diabética en Canarias

La Comunidad Autónoma Canaria cuenta con una población de aproximadamente 2.100.000 habitantes repartidos en 7 islas principales, con una población estimada de diabéticos de 221.000 siendo la retinopatía diabética la principal causa de ceguera de la población en edad laboral activa. Dado que el número de oftalmólogos de las islas ronda los 160, sería necesario que cada oftalmólogo examinara a 7 pacientes diabéticos por día laborable para poder cribarlos a todos con una periodicidad anual. Por este motivo, se decidió implantar en 2006 un sistema de tele-oftalmología que involucre al sistema de atención primaria en el ámbito de la Comunidad Autónoma de Canarias, el programa Retisalud.

El programa se basa en que el médico de familia, que ya conoce el control metabólico del paciente y las comorbilidades que haya generado la enfermedad a lo largo de su vida, pudiera también conocer a grandes rasgos el estado de la retina. El médico de familia ya lleva el control de otras complicaciones microvasculares de la diabetes como la nefropatía o la neuropatía,

remitiéndola al nefrólogo o neurólogo solo cuando se han sobrepasado determinados límites. Esto los convierte en buenos candidatos para valorar otra de las complicaciones microvasculares: el fondo de ojo de sus pacientes. Para que pudieran actuar como primer escalón en el cribado de la RD se les impartió un curso presencial y debieron realizar un examen de más de 200 retinografías que debían valorar como normales o patológicas.

En el informe del programa Retisalud publicado en "Archivos de la Sociedad Española de Oftalmología" [66], se muestra el aumento anual del número total de pacientes cribados, habiéndose detectado en el año 2015 a 3.353 patológicos que no habrían sido detectados con los medios anteriores al programa. Además, el porcentaje de pacientes con retinopatía severa o grave ha ido descendiendo año a año, lo que sugiere que el control sobre la población objetivo es cada vez más efectivo, permitiendo detectar la patología antes de que aumente de gravedad.

Así, los primeros años del programa, ante cualquier duda los médicos de familia remitían las retinografías para que fueran valoradas por un oftalmólogo, mientras que más recientemente y tras años de experiencia han ido incorporado a su práctica clínica habitual la lectura de las retinografías, clasificando como normales a un porcentaje cada vez mayor. Así, los últimos tres años han sido los de mayor número de retinografías clasificadas como normales por parte de los médicos de familia [66]. A pesar de esto, un total de 7.742 pruebas fueron evaluadas en 2015 de forma telemática por oftalmólogos de forma innecesaria ya que, tras ser valoradas por el médico de familia como patológicas o dudosas, resultaron ser normales. Dado que los datos de carácter clínico obtenidos durante este programa pueden ser cruciales para futuros planes a nivel nacional, todos los datos, obtenidos mensualmente mediante registro informático automatizado desde su inicio, han sido almacenados de manera segura alcanzándose la cifra de 600.000 imágenes de fondo de ojo clasificadas según los estándares oftalmológicos europeos. Esto constituye una de las bases de datos de imágenes de fondo de ojo clasificadas más grandes a nivel mundial, siendo únicamente superada por la obtenida por Google, con de 1.5 millones de imágenes obtenidas en 3 clínicas de la India y cuyo principal problema, según los propios autores, es la falta de clasificación según los estándares americanos que tienden a ser más restrictivos [67].

Para el diseño, entrenamiento e implementación de una inteligencia artificial es necesario un conjunto de entrenamiento, que será más conveniente cuanto más grande sea y mejor clasificado esté. Por tanto, la posibilidad de poder

utilizar esta base de datos recabada durante todos los años del programa Retisalud permite desarrollar un método realmente efectivo que asista a los médicos de familia de Canarias en el cribado de la retinopatía diabética.

3.2. Retisalud

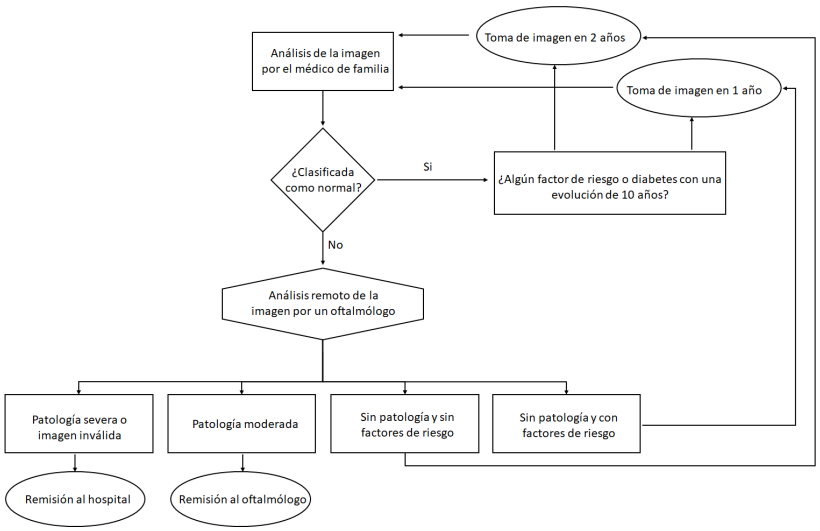


FIGURA 3.1: Protocolo de evaluación de retinografías de Retisalud. Imagen traducida del trabajo de Ríos et al [66].

3.3. *Deep learning* aplicado a Retisalud

3.3.1. Datos

Tras 10 años de obtención de datos, se logra elaborar una base de datos con un total de 422.531 casos formados por parejas de imágenes (izquierda/derecha), un campo para la evaluación por parte de atención primaria (AP), otro campo para la evaluación por parte de atención especializada (AE), fecha e identificador de paciente debidamente anonimizado para que no exista modo de relacionarlo con ningún individuo.

Los casos están distribuidos según se muestra en la figura 3.2 de la siguiente forma: 75.931 sin evaluar, 258.922 evaluados únicamente por AP, 33.591 valorados únicamente por AE y 54.087 por ambos. Siguiendo el protocolo de

Retisalud de la figura 3.1, se deduce que, no debería ser posible que existan casos sin evaluaciones ni tampoco casos evaluados únicamente por AE, pues, para llegar a AE deberían haber sido evaluados previamente por AP como "No normal".

A pesar de contar con imágenes resultado del protocolo presentado en la figura 3.1, aún así existen casos en la base de datos sin ningún tipo de evaluación. Estos son considerados erróneos, pues, al no contar con ningún tipo de evaluación, no pueden ser utilizados ni para entrenar un modelo ni para evaluar su rendimiento. Por otro lado, los casos con evaluación de AE pero sin ninguna clasificación por parte de AP si son utilizados, pues, cuentan con la evaluación del especialista. En cuanto a la falta de clasificación por parte de AP, puede explicarse como casos en que AP no considera que haya signos de retinopatía diabética pero sí detecta alguna patología no diabética, o como casos en que AP tiene dudas acerca de la retinografía. A falta de una etiqueta de "patología no diabética" o "dudosa", el campo de clasificación por parte de AP queda vacío.

Los datos evaluados por AE, unos 87.678, están clasificados como "No signos de retinopatía diabética" (No RD), "Retinopatía diabética no proliferativa leve" (RDNP leve), "Retinopatía diabética no proliferativa moderada" (RDNP moderada), "Retinopatía diabética no proliferativa severa" (RDNP severa), "Retinopatía diabética no proliferativa muy severa" (RDNP muy severa), "Retinopatía diabética proliferativa " (RDP), "Retinopatía diabética proliferativa con características de alto riesgo" (RDP con CAR).

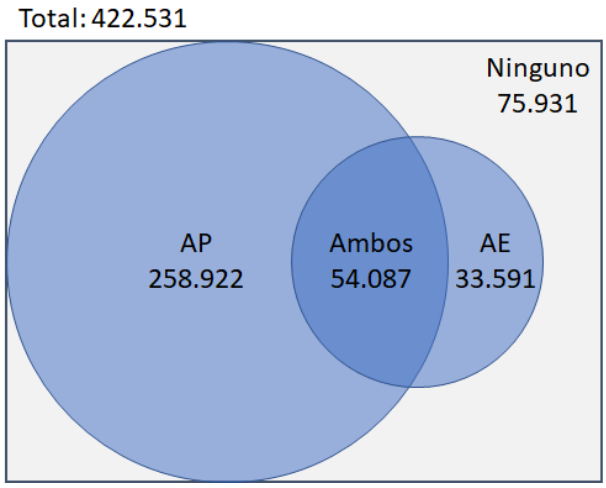


FIGURA 3.2: Distribución de los datos obtenidos por el programa Retisalud entre los años 2007 y 2017. Un total de 422.531 parejas de imágenes (izquierda/derecha), de las cuales, 75.931 no cuentan con ninguna evaluación, 258.922 fueron evaluadas únicamente por atención primaria (AP), 33.591 únicamente por atención especializada (AE) y 54.087 por ambos.

Una pequeña fracción de estos datos obtenidos por el programa Retisalud es utilizada para entrenar una red neuronal con la que se realizará el estudio que tratará de responder a la pregunta:

¿Qué hubiera pasado si Retisalud hubiera contado con la asistencia de un algoritmo de cribado automático?

Concretamente, de los 87.678 casos evaluados por AE, se extraen 13.840 para el desarrollo del modelo. Estos 13.840 casos son 27.680 imágenes, 19.224 clasificadas como "No RD" y 8.456 como "RDNP leve" o superior.

3.3.2. Método

El modelo base utilizado para la clasificación es ResNet-50 [20] preentrenado sobre ImageNet[5] con el añadido de bloques Squeeze-Excitation [22] ya expuestos en la sección 2.2.6. Teniendo en cuenta que muchos de los signos de retinopatía diabética son diminutos micro-aneurismas, en lugar de utilizar la resolución estándar de 224x224, se duplica a 448x448 para así evitar perder estos pequeños signos por falta de resolución. Al duplicar la resolución,

se añaden convoluciones dilatadas en las últimas capas del modelo para así mantener el campo receptivo en las etapas finales. El resultado de este modelo es la probabilidad de que la imagen que se le presenta contenga al menos un signo de retinopatía diabética. La arquitectura de esta red neuronal se presenta en la figura 3.3.

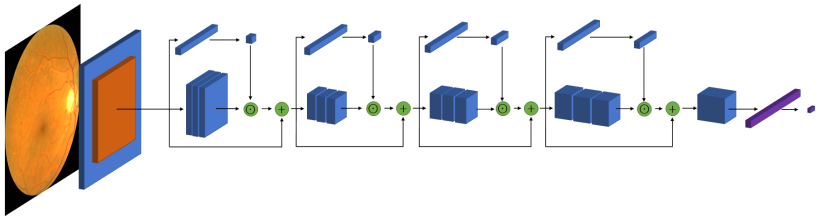


FIGURA 3.3: Arquitectura de la red neuronal para la clasificación de imágenes de fondo de ojo en función de la existencia algún signo de retinopatía diabética.

Este modelo se entrena empleando el optimizador ADAM [68] durante 50 épocas utilizando el conjunto de datos de desarrollo descrito en 3.3.1 dividido en un 80 % para entrenamiento y 20 % para validación. Como se describió en 3.3.1, el conjunto de datos de entrenamiento no está balanceado, es decir, del total de imágenes, el 69 % corresponden a imágenes no patológicas, mientras que tan solo el 31 % corresponden con casos patológicos. Si bien no es un caso extremo, si no se atiende esta descompensación se obtendría un modelo sesgado hacia casos no patológicos. Para solventar este problema, se le aplica a la entropía cruzada, que es utilizada como función de costes, un peso a cada tipo de error correspondiente con la proporción de elementos de cada tipo dentro del conjunto de datos. Esta función de costes será dada por la ecuación:

$$\mathcal{L}(p, p^*) = -(wp^* \log(p) + (1 - p^*) \log(1 - p)) \tag{3.1}$$

Con $p \in [0, 1]$ como la probabilidad estimada de que la imagen sea patológica, $p^* \in \{0, 1\}$ la probabilidad real que lo sea y w el factor de corrección dado por la proporción de casos no patológicos por cada caso patológico, en este caso $w = \frac{69}{31}$. En otras palabras, lo que se logra con esta corrección es dar más peso al coste provocado por equivocarse a la hora de detectar un caso patológico frente al error provocado por un falso positivo.

3.3.3. Evaluación

Para la evaluación, se utiliza el conjunto de test con todas los casos considerados válidos que no fueron utilizados para el desarrollo. Un caso se considera válido solo si fue evaluado al menos una vez por al menos un médico (de familia u oftalmólogo). Existen ambas imágenes (izquierda y derecha) y en cada imagen se puede identificar una papila circular, forzando a que cada imagen del conjunto de test tenga una papila con un radio con una variabilidad de ± 10 píxeles como máximo. De este modo se asegura de eliminar de este conjunto aquellas imágenes sobreexpuestas, muy desenfocadas o simplemente mal capturadas. Además, siguiendo las premisas del protocolo de Retisalud, cada caso considerado patológico o dudoso por AP deberá ser evaluado por AE, por lo que, casos clasificados como patológicos por AP pero sin evaluación por parte de AE no deberían existir. Estos casos también son considerados inválidos puesto que no cumplen con el protocolo del programa. De hecho, estas situaciones son marginales en la base de datos (<4 %).

Después de eliminar las imágenes utilizadas para el desarrollo del algoritmo y las inválidas, el conjunto de test final consiste en 237.665 casos: 221.806 evaluados por AP y 71.819 por AE.

La evaluación del algoritmo para cada caso consiste en la ejecución del mismo por cada par de imágenes (izquierda/derecha), considerando el caso como patológico si al menos uno de los ojos fue clasificado de tal forma. No se hace distinción por grado de severidad, tan solo la predicción de si existe o no signos de RD en al menos un ojo.

3.3.4. Resultados

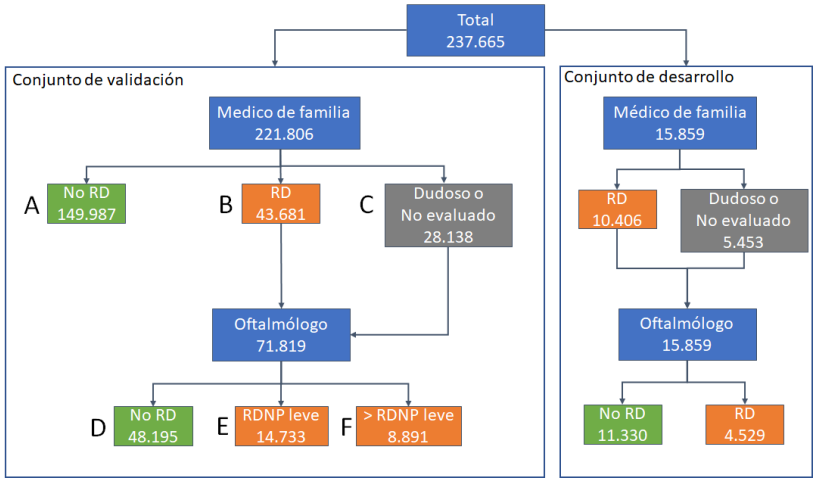


FIGURA 3.4: Distribución del conjunto de desarrollo y *test*: A: Casos vistos únicamente por AP y clasificados como "No RD", B: Casos vistos por AP y clasificados como retinopatía diabética (RD), C: Casos vistos por AP y clasificados como dudosos o como patologías no diabéticas (PND), D: Casos vistos por AE y clasificados como "No RD", E: casos vistos por AE y clasificados como "RDNP leve", F: Casos vistos por AE y clasificados como superior a "RDNP leve".

En la figura 3.4 se muestra la distribución de los datos. La distribución del conjunto de test, fruto del protocolo de cribado de Retisalud, presenta una serie de características a tener en cuenta a la hora de interpretar los resultados:

(a) Los médicos de familia de AP no necesariamente se especializan en oftalmología.

(b) No todos los casos fueron evaluados por AE.

(c) Los casos que evalúa AE son solo aquellos que fueron previamente evaluados como RD, patología no diabética (PND) o dudoso por parte de AP.

(d) AE no verá la imagen de fondo de ojo de un paciente que ha sido clasificado como "No RD" por parte de AP a menos que, en alguna revisión sucesiva, AP lo clasifique como RD, PND o dudoso.

Estas características implican que, teniendo en cuenta (a), si bien se llevan a cabo programas de formación para que AP pueda evaluar cada vez mejor, al no ser la oftalmología su área de especialidad como lo es en el caso de AE, sólo el diagnóstico de AE podrá considerarse como "verdadero" a la hora de evaluar el desempeño de un algoritmo de cribado automático. En esta misma línea y siguiendo la característica (b), el número de casos del conjunto de test que podrán utilizarse para obtener un resultado numérico del desempeño de un algoritmo se reduce drásticamente, al contar con una evaluación por parte de AE de tan solo el 31.9 % de los casos. Siguiendo el protocolo de Retisalud según se resume en la característica (c), se deduce que AE solo recibirá casos de "No RD" si estos son o bien PND, o bien son lo suficientemente difíciles como para hacer dudar o confundir a AP de que se trata de un caso de RD. Por ende, no se tiene una evaluación por parte de AE de casos claramente sanos, dando lugar así al gran número de casos que no ha visto AE.

Dadas estas características, el conjunto de test se divide en los 6 segmentos (A...F) que se muestran en la figura 3.4:

(A) Casos evaluados únicamente por AP que son visiblemente sanos. Como es de esperar, los casos sanos han de ser la mayoría, y en efecto, este segmento constituye la mayor parte del conjunto de test (68.1 %).

(B) Casos evaluados por AP como RD. Estos casos constituyen la mayoría de casos que recibe AE Sin embargo, de este segmento, AE considera que el 62.5 % era en realidad "No RD".

(C) Casos evaluados por AP que son dudosos o PND. De este segmento, AE considera que 74 % era en realidad "No RD".

(D) Casos evaluados por AE como "No RD". Este presenta el conjunto de casos sin retinopatía que son o bien PND o fácilmente confundibles con un caso de RD.

(E) Casos evaluados por AE como "RDNP leve".

(F) Casos evaluados por AE como "RDNP moderada" o superior.

Para cada uno de estos segmentos se ejecuta el algoritmo de cribado automático y se analizan los resultados. En la tabla 3.1 se muestra el resumen de resultados para cada segmento.

	AP			AE			IA	
	No DR	DR	D o PND	No DR	MDR	MDR	No DR	DR
(A)	149.987 (100 %)	0 (0 %)	0 (0 %)	-	-	-	134.880 (90 %)	15.107 (10 %)
(B)	0 (0 %)	43.681 (100 %)	0 (0 %)	27.319 (62 %)	10.480 (24 %)	5.882 (13 %)	22.834 (52 %)	20.847 (48 %)
(C)	0 (0 %)	0 (0 %)	28.138 (100 %)	20.876 (74 %)	4.253 (15 %)	3.009 (11 %)	17.198 (61 %)	10.940 (39 %)
(D)	0 (0 %)	27.319 (57 %)	20.876 (43 %)	48.195 (100 %)	0 (0 %)	0 (0 %)	35.104 (73 %)	13.091 (27 %)
(E)	0 (0 %)	10.480 (71 %)	4.253 (29 %)	0 (0 %)	14.733 (100 %)	0 (0 %)	4.485 (30 %)	10.248 (70 %)
(F)	0 (0 %)	5.882 (66 %)	3.009 (34 %)	0 (0 %)	0 (0 %)	8.891 (100 %)	443 (5 %)	8.448 (95 %)

TABLA 3.1: Evaluación por parte de AP, AE y nuestro algoritmo para cada segmento de la figura 3.4. Los porcentajes son respecto del total de casos de cada segmento.

Evaluación sobre los segmentos A y D

Para valorar el desempeño de nuestro algoritmo detectando casos "No RD", éste se evalúa sobre el segmento D (No RD para AE) de la figura 3.4. En el 72 % de los casos nuestro algoritmo coincide con AE Como se mencionó anteriormente, este segmento contiene solo aquellos casos "difíciles" o PND del total de casos "No RD", por tanto, este resultado no representa la especificidad real de nuestro algoritmo. La especificidad real del algoritmo, tal y como está compuesto el conjunto de test, no podrá ser calculada al no contar con una evaluación por parte de AE sobre el segmento A (No RD según AP).

No obstante, al evaluar nuestro algoritmo sobre dicho segmento, éste coincide con AP en el 90 % de los casos. Si se asumiera "verdadero" el diagnóstico de AP para el segmento A (No RD según AP), y, considerando también el resultado obtenido en el segmento D (No RD según AE), nuestro algoritmo sería capaz de clasificar correctamente el 85 % de los casos "No RD".

Sin embargo, en lugar de considerar la evaluación de AP sobre el segmento A como "verdadera" se podría hacer un análisis diferente. Los casos en los que nuestro algoritmo y AP discrepan, el 10 %, representa 11.825 pacientes diferentes, 6.578 de ellos fueron reevaluados en una fecha posterior. De estos,

2.798 fueron vistos nuevamente por AE, de los cuales, 1.258 fueron evaluados con una clasificación de "RDNP leve" o superior. Esto significa que, nuestro algoritmo hubiera enviado estos casos de RD al oftalmólogo antes aún.

Evaluación sobre los segmentos B y C

De las 71.819 visitas que fueron evaluadas por AP como DR o dudosas, 48.195 (67 %) fueron finalmente evaluadas como sanos por AE Esto significa que los oftalmólogos recibieron un cierto número de casos extras para evaluar (el 21 % del total de pacientes cribados por Retisalud). En la figura 3.5 se expone el posible desempeño de una IA, al mostrar la concordancia de criterio de ésta con el de AE (tomado como *ground truth*). Si se hubiera utilizado únicamente el criterio de la IA para evaluar los pacientes de los segmentos analizados, el número de casos sanos enviados a AE se hubiera reducido en 13.091. Sin embargo, hubieran habido también 4.928 pacientes patológicos no detectados, 443 de ellos con un grado de retinopatía superior a leve. Cabe destacar que la mayor parte de este error reside en los casos leves (89 %). Este caso específico será discutido en detalle más adelante.

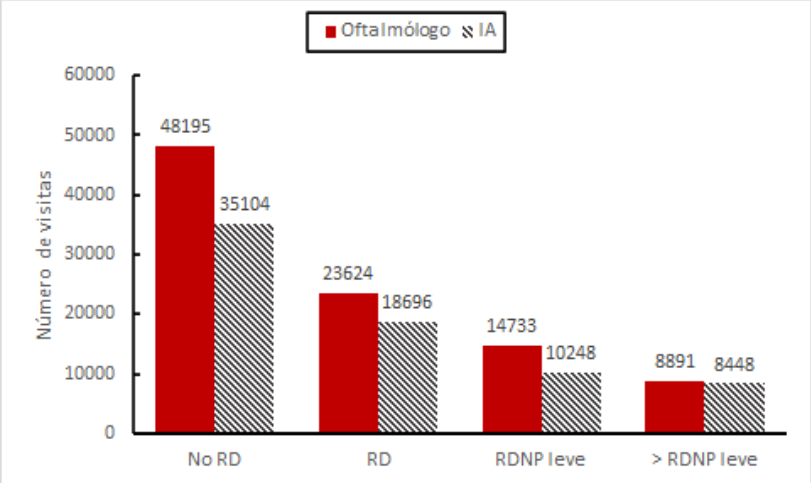


FIGURA 3.5: Comparativa de criterio de la inteligencia artificial (IA) frente a la evaluación del oftalmólogo, separado por severidad.

Evaluación sobre los segmentos D, E y F

Los segmentos D (No signos de RD según AE), E (RDNP leve según AE) y F (RDNP moderada según AE) de la tabla 3.1 pueden ser utilizados para realizar una estimación de la especificidad y sensibilidad de la IA. En este sentido, la especificidad resultante de utilizar los datos de estos tres segmentos es de 0.73. Sin embargo, en el segmento D (No signos de RD según AE), únicamente se encuentran los casos más difíciles o los que presentan alguna patología no diabética, por tanto, este valor solamente representa la especificidad de este subconjunto y no de todo del conjunto de datos completo. Esto se debe al protocolo de Retisalud, pues, en el segmento D se encuentran únicamente aquellos casos que fueron lo suficientemente difíciles como para confundir a los médicos de familia de AP en la primera evaluación. Si se considerara los casos del segmento A (No RD según AP) como correctamente evaluados por AP, la especificidad total sería del 0.85.

La sensibilidad obtenida fue del 0.79 para los segmentos E y F (cualquier grado de RD según AE). Sin embargo, este valor varía en gran medida en cada segmento. Analizando únicamente los casos diagnosticados como leves, se tiene una sensibilidad del 0.7, mientras que para los casos superiores a leve es de 0.95.

Reevaluación de resultados

Teniendo en cuenta que el conjunto de *test* no ha sido sometido a un proceso exhaustivo de revisión, y dado que la mayoría de los errores se tienen a la hora de clasificar los casos "No RD" y "RDNP leve", se diseña una serie de reevaluaciones en las que un especialista en oftalmología revisita una serie de casos. Por caso, se entiende a la pareja de imágenes de fondo de ojo (izquierda y derecha) tomadas durante una misma visita y pertenecientes al mismo paciente. Estas reevaluaciones se hacen siguiendo un proceso a ciegas. El especialista recibe un conjunto de parejas de retinografías (una de cada ojo por cada caso) que evalúa de forma individual según las categorías de "No RD", RD o patología no diabética (PND). El caso en su conjunto es clasificado a su vez en una de estas tres categorías siguiendo las condiciones:

- "No RD" si ambas retinografías son clasificadas como tal.
- PND si al menos una de las dos es clasificada como tal y la otra no es considerada RD.
- RD si al menos una es clasificada como tal.

Adicionalmente, para poder obtener resultados estadísticamente significativos, para cada reevaluación, el tamaño de la muestra se calcula siguiendo la siguiente ecuación:

$$n = \frac{z^2_{\alpha/2} p(1-p)N}{z^2_{\alpha/2} p(1-p) + (N-1)m^2} \tag{3.2}$$

Donde $z_{\alpha/2}$ es el cuantil de la distribución normal para una confianza del 95 %, p la probabilidad de éxito (por ejemplo, probabilidad de tener una correcta evaluación para cada segmento, utilizamos $p = 0{,}5$ asumiendo el peor caso), N el tamaño de cada segmento y m el margen de error (en nuestro estudio asumimos un margen de error del 5 %).

Reevaluación sobre el segmento A

Se diseñan dos análisis diferentes para este segmento A (No RD según AP).

Un primer análisis que toma muestras aleatorias de los 1.258 pacientes que recibieron una primera evaluación de "No RD" por parte de AP y, que en una visita posterior, fueron evaluados nuevamente por AP como "RD" y llegaron a recibir un diagnóstico patológico por parte de AE En este primer análisis, la visita en cuestión que se analiza es la primera visita en que la IA clasifica como patológica y AP como no patológica de cada paciente elegido.

El segundo análisis, más general, formado por muestras aleatorias del conjunto de 11.824 pacientes con una clasificación discrepante entre la IA y AP.

El análisis se realiza siguiendo el procedimiento de reevaluación explicado en la sección 3.3.4. El primero, con 222 casos. 25 (11 %) fueron clasificados como No DR, 46 (21 %) como PND, y 151 (68 %) como RD. El segundo con 262. 80 (30 %) fueron clasificados como No DR, 88 (34 %) como PND, y 94 (36 %) como RD.

Reevaluación sobre el segmento D

Sobre este segmento D (No RD según AE) se diseña un análisis en el que se toman muestras aleatorias de este segmento en las que la IA discrepa con AE y se analizan siguiendo el mismo procedimiento anterior y explicado en la sección 3.3.4. De los 13.091 casos discrepantes del segmento D, 266 son reevaluados. De los cuales, 88 (33 %) fueron reclasificados como RD, 30 (11 %) como PND, y 148 (56 %) como No RD.

Reevaluación sobre el segmento E

En el caso de este segmento, el correspondiente a casos "RDNP leve", se realiza un análisis similar al anterior, siguiendo el mismo procedimiento. De los 4.485 casos discrepantes, se reevalúan 256. De los cuales, 177 (69 %) fueron reclasificados como RD, 46 (18 %) como No RD, y 33 (13 %) como PND.

3.4. Interpretabilidad

El trabajo realizado permitió crear un sistema capaz de clasificar automáticamente imágenes de fondo de ojo en función de si presentan o no signos de retinopatía diabética. Si bien se demostró un desempeño positivo, aún quedan interrogantes que resolver para que el sistema sea de utilidad en la práctica. Específicamente, es necesario estudiar la interpretabilidad del sistema, o, en otras palabras, poder justificar la razón por la cual se llega a un determinado resultado. Esta capacidad de poder interpretar los resultados permitiría no sólo validar el modelo al verificar que se analizan los signos de retinopatía diabética que deben ser analizados, sino también poder asistir a los médicos en la toma de decisiones al dar información explicativa.

Esta necesidad de poder interpretar la clasificación realizada toma especial interés en los casos de retinopatía leve, pues, los signos de este nivel de severidad son, en su mayoría, muy sutiles y fáciles de pasar por alto incluso para un experto.

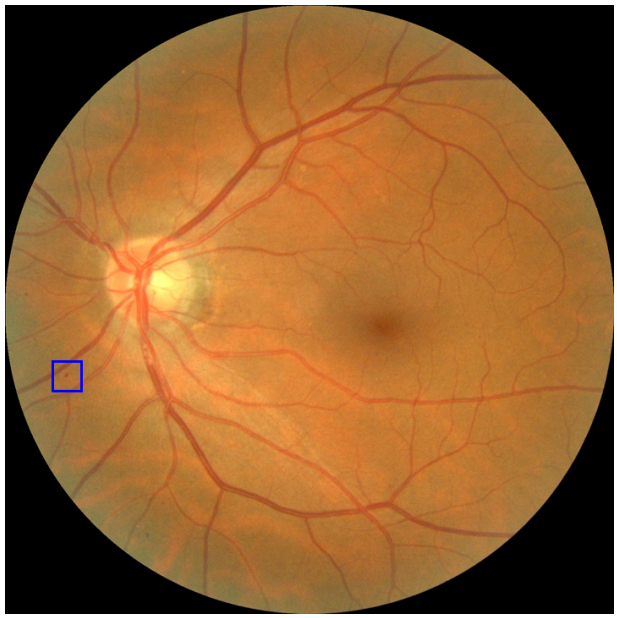


FIGURA 3.6: Imagen de fondo de ojo con "RDNP leve". Se aprecia un pequeño signo de retinopatía diabética en la parte izquierda remarcado con un rectángulo azul.

En la figura 3.6 se muestra un ejemplo de un caso de retinopatía diabética leve. El único signo de retinopatía diabética que justifica la clasificación de esta muestra como "RDNP leve" es el pequeño microaneurisma de la izquierda marcado en azul. De lo contrario, esta retinografía sería considerada "No patológica". Dada la sutileza de los signos en estos casos, un sistema de clasificación automático que clasifique correctamente este caso como "RDNP leve" pero que no ofrezca ningún otro tipo de información podría entrar en discrepancia con un médico que haya pasado por alto este pequeño signo. En este caso habría una discrepancia pero, al no existir mayor indicación, el sistema automático resultaría de poca utilidad.

Por tanto, se estudia la forma en que se pueda lograr una justificación por parte del sistema. En concreto, se analizan diferentes métodos que permitan resaltar qué zonas de la imagen dan pistas que esta es patológica.

3.4.1. Métodos

Tal y como se analizó en el capítulo 2.3, a grandes rasgos, existen tres métodos para obtener los mapas de las zonas que apoyan la decisión de clasificar de un modo u otro una imagen: análisis mediante oclusiones (2.3.1), análisis de gradientes (2.3.2) y análisis de mapas de activación (2.3.3).

Dadas las características de las imágenes que se trabajan, se decide descartar el análisis mediante oclusiones para enfocar el esfuerzo en explorar las otras dos ramas.

Análisis de gradientes

La idea detrás del análisis de gradientes consiste, *grosso modo*, en calcular el gradiente de los *logits* de la clase deseada, clase patológica en este caso, respecto a la imagen de entrada. Esto da como resultado, para cada píxel de la imagen de entrada, cuánto aumenta o disminuye la certeza de que esta imagen es patológica si se cambiara la intensidad de uno de estos píxeles en 1. En otras palabras, se obtiene la importancia de cada píxel de la imagen para la clasificación de esta imagen como patológica.

En la práctica, el cálculo del gradiente de los *logits* respecto a la entrada no da resultados fácilmente interpretables. En este sentido, con el fin de mejorar la interpretabilidad, existen diversos trabajos en la literatura, específicamente, la retro propagación guiada [24] y los gradientes integrados [25].

Ambos tienen requisitos específicos para poder aplicarse. En el caso de la retropropagación guiada necesita que las activaciones del modelo sean rectificaciones lineales y, en el caso de los gradientes integrados, requiere establecer una imagen considerada línea de base. Dado que el modelo utilizado sí utiliza rectificaciones lineales como activación, se decide optar por el cálculo de gradientes mediante retropropagación guiada.

En la figura 3.7 se muestra un ejemplo de retropropagación guiada aplicada a una imagen etiquetada como “RDNP leve”. Este ejemplo concreto es clasificado como “RDNP leve” debido a un microaneurisma en la parte izquierda de la imagen. Se puede apreciar en el mapa de gradientes que la zona donde aparece el microaneurisma queda resaltada mientras que el resto de la imagen permanece en gris. Para facilitar la apreciación, se remarca en azul.

Esto significa que, un cambio de intensidad en los píxeles pertenecientes al microaneurisma contribuyen en una medida mucho mayor a la clasificación

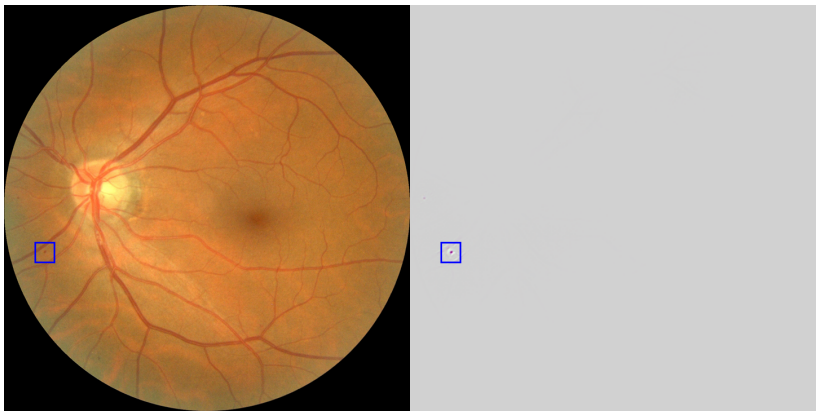


FIGURA 3.7: Retinografía etiquetada como "RDNP leve" con un pequeño signo de retinopatía diabética junto a su mapa normalizado de gradientes de la imagen de entrada. En ambas imágenes se resalta en un recuadro azul la zona donde se aprecia la lesión.

de si esta imagen es patológica o no que un cambio en intensidad de un píxel que no pertenece a esta zona. En otras palabras, se puede decir que, el microaneurisma contribuyó en mayor medida a la clasificación de esta imagen como patológica que el resto de la imagen.

Si bien cualitativamente es un resultado positivo, aún no es posible utilizar esta información para mostrar el nivel de influencia de cada píxel en la clasificación final, pues, al tener una entrada de tres canales (RGB), se tiene un gradiente de tres canales también. Por tanto, será necesario un método que agregue la información de cada uno de los canales.

Frente a la necesidad de agregar los tres canales en uno, se plantean los siguientes métodos:

- Suma del valor absoluto de cada canal.
- Distancia euclídea de cada punto respecto a un gradiente plano.
- Desviación de cada punto respecto la media de los tres canales.

Al representar cada pixel la medida en que cambia la decisión final frente a un aumento de 1 en la intensidad de cada canal, en caso de tener un pixel con valor de gradiente negativo significará que este contribuye negativamente a la consideración de la imagen como patológica. Tanto si los gradientes son positivos como negativos estos contribuyen a la "importancia" del pixel para

la clasificación, como primera operación de agregación se considera la suma en valor absoluto de todos los canales.

Similar a la suma en valor absoluto, se considera como segunda opción la agregación mediante distancia euclídea, para así dar más importancia a los puntos más dispersos.

Por último, al hablar únicamente de intensidades, el hecho de que un pixel tenga el mismo gradiente en los tres canales indica que, o bien este no es importante para la clasificación (en caso de ser los tres cercanos a cero), o bien simplemente hay que oscurecer o aumentar el brillo de ese pixel para inclinar la clasificación. Dado que está trabajando con imágenes de retina y no es normal encontrar puntos que sean monocromos, es de esperar que los puntos realmente importantes para la clasificación tengan gradientes distintos de cero y a su vez, distintos entre cada canal. Por tanto, como tercera operación de agregación se propone utilizar la desviación de cada punto respecto la media de sus tres canales.

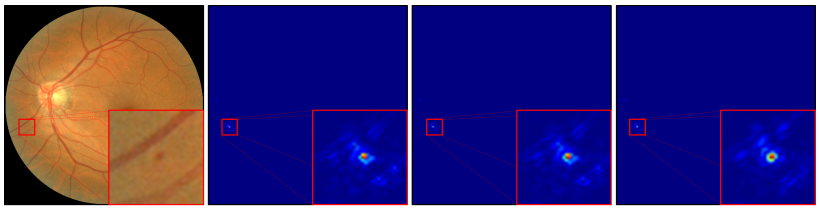


FIGURA 3.8: Imagen de gradientes de la figura 3.7 agregados con suma de valor absoluto, distancia euclídea y desviación respectivamente. Para mejor visualización, se muestra un recorte con aumento en la zona de interés de cada imagen.

En la figura 3.8 se muestra un ejemplo de la aplicación de los tres métodos de agregación sobre la imagen 3.6. Si bien todos ofrecen resultados similares, el método basado en desviaciones consigue acotar mejor la zona a resaltar. En el contexto del problema, desde el momento en que se detecta un único signo de retinopatía ya se puede considerar patológico, por tanto, en este caso sería más deseable contar con el método basado en desviación al ser el que más acota el área a resaltar. En el caso de ejemplo mostrado en la figura, a pesar de haber una diferencia de apenas unos pocos píxeles, estos representan una buena parte de la lesión a detectar.

Análisis de mapas de activación

Tal y como se introdujo en la sección 2.3.3, los mapas de activación se pueden extraer de la penúltima capa de la red neuronal para así obtener una medida de la importancia de cada zona de la imagen para la clasificación final. Este método, a diferencia de los basados en gradiente, en lugar de inferir la importancia de cada pixel a través de un gradiente modificado de los *logits*, la obtienen directamente de la entrada a los mismos a costas de obtener un resultado en mucha menor resolución.

Para poder extraer estos resultados, primero se modificó la arquitectura de la red neuronal utilizada para que pudiera dar, además de una clasificación, los mapas de activación para la clase de "patológico".

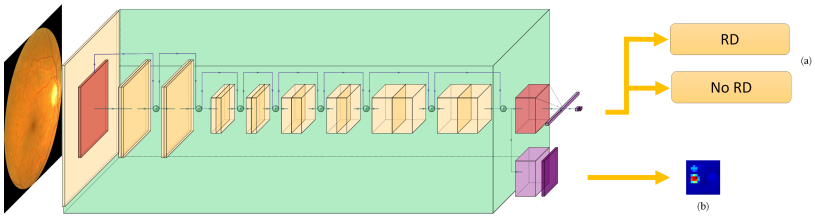


FIGURA 3.9: Arquitectura modificada para la extracción de mapas de activación. Obtiene, por un lado, una clasificación en "RD" y "No RD" (a) y, por otro, los mapas de activación (b).

En la figura 3.9 se muestra la arquitectura modificada para obtener los mapas de activación.

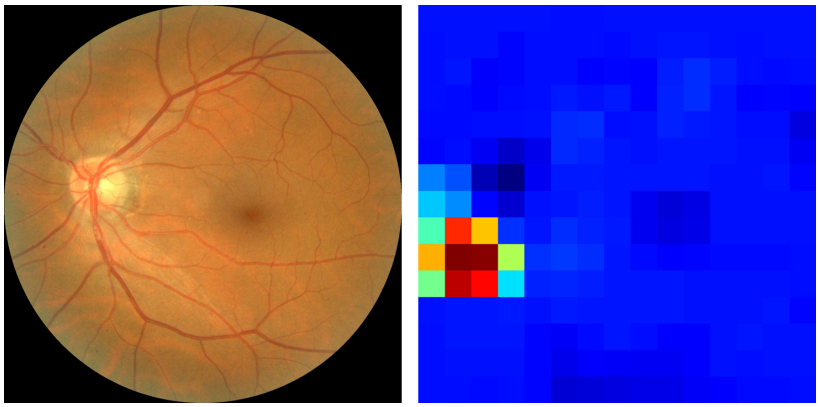


FIGURA 3.10: Mapas de activación para la imagen 3.6.

En la figura 3.10 se muestra el resultado de extraer los mapas de activación de la imagen de la figura 3.6. Se puede apreciar que es capaz de mostrar las zonas de la imagen a la que pertenece la lesión de interés. Sin embargo, debido a la baja resolución, esta detección es imprecisa.

Para afrontar este problema y poder detectar de forma más acotada, se propone una aproximación multiescala: extraer los mapas de activación a diferentes escalas y mezclarlos para así retener la información de lesiones de diferentes tamaños a la vez que se consigue refinar la detección.

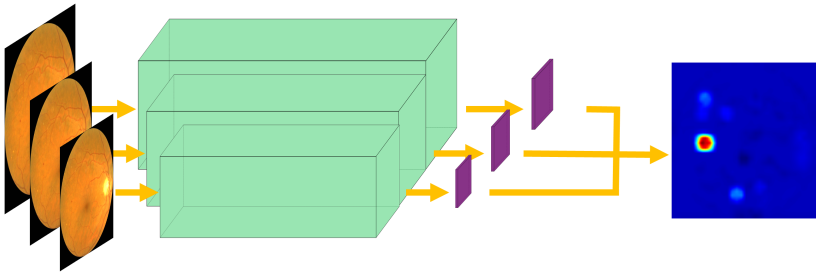


FIGURA 3.11: Extracción multiescala del mapa de activación.

En la figura 3.11 se muestra la arquitectura aplicada a múltiples escalas sobre la imagen 3.6. De dicha imagen, se extraen tres mapas de activación en diferentes escalas mostrados en la figura 3.12.

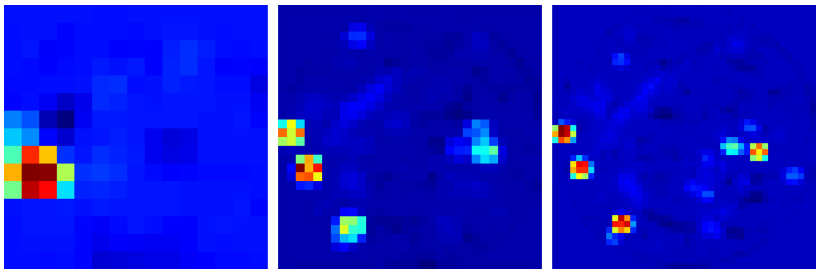


FIGURA 3.12: Mapas de activación de la imagen 3.6 a diferentes escalas.

Estas tres escalas se combinan para así formar el resultado de la figura 3.13.

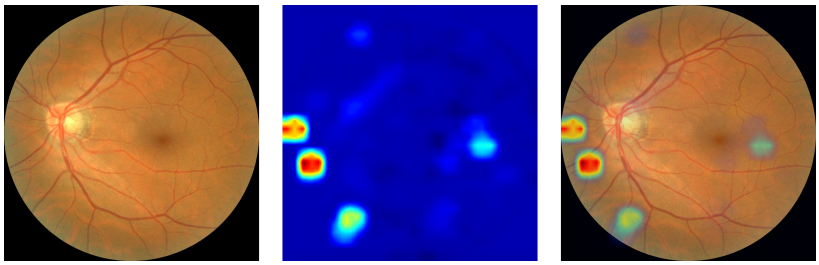


FIGURA 3.13: Mapas de activación agregados de la imagen 3.6. Se muestra la retinografía de entrada, el resultado de la agregación de los mapas de activación a múltiples escalas y la superposición del mapa con la entrada.

3.4.2. Resultados

Los métodos explorados en las secciones anteriores se aplicaron a una serie de imágenes para así verificar su desempeño. En la figura 3.14 se muestran algunos ejemplos.

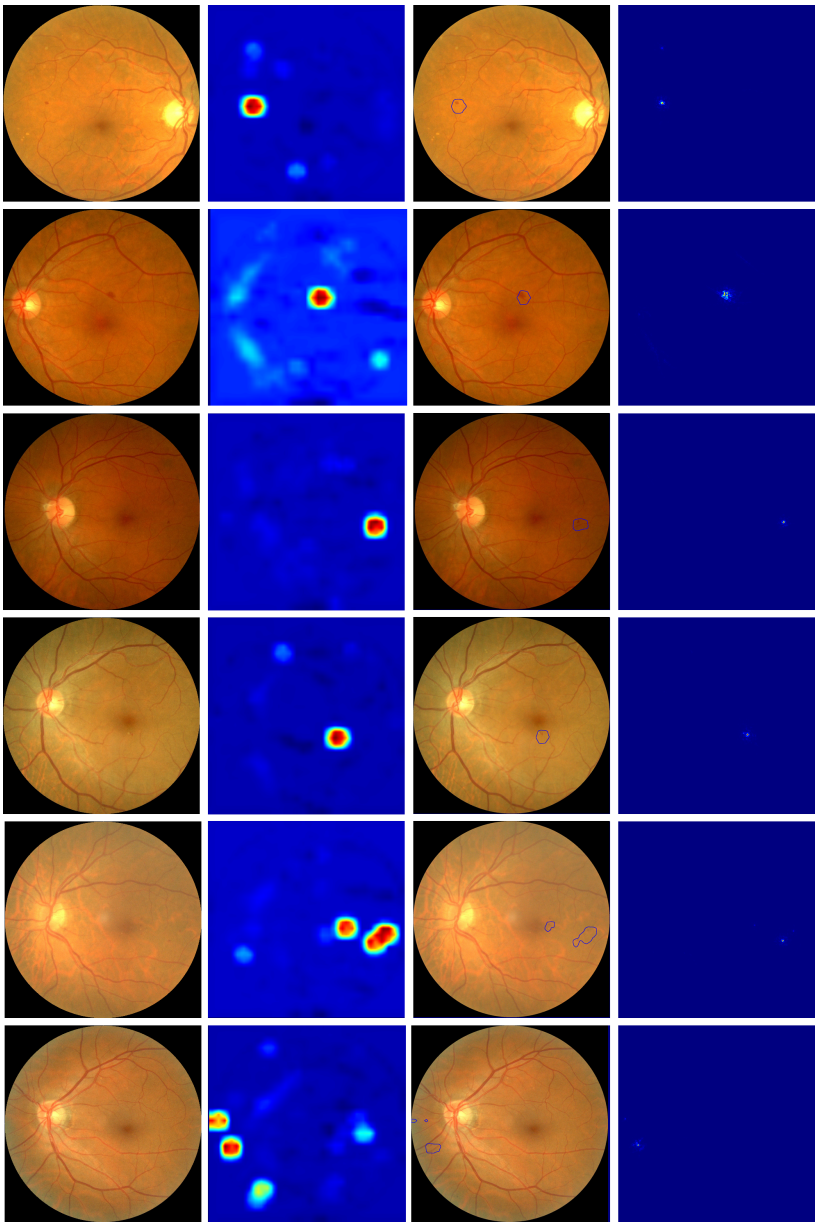


FIGURA 3.14: Aplicación de los métodos de interpretación. Primera columna: retinografía de entrada. Segunda columna: mapas de activación multiescala. Tercera columna: selección automática de zonas de interés en base a los mapas de activación. Última columna: gradientes.

En esta figura se muestra también una detección automática de las zonas de interés en base a los mapas de activación (tercera columna). Cabe destacar que, a pesar de obtener resultados visualmente positivos, existe un gran inconveniente a la hora de analizarlos con un método automático que dificulta en gran medida la obtención de un resultado de detección/segmentación automático de todos los signos de retinopatía diabética. Tanto para el caso de gradientes como para el caso de mapas de activación, se obtiene un mapa de la medida de "importancia" de cada pixel para la clasificación. Este valor de "importancia" no está acotado a ningún rango específico, por lo que, la decisión de que valor de "importancia" tiene que tener un pixel para que éste sea importante viene dado por el rango de valores obtenido para cada imagen individual. Es justo esta falta de "referencia" la principal dificultad, pues, esto hace necesario el uso de un algoritmo adicional para segmentar o agrupar la imagen de gradientes o activaciones resultantes para poder detectar automáticamente las zonas de interés. El algoritmo utilizado en la figura de resultados 3.14 es un algoritmo de detección de contornos de las zonas que tengan píxeles con un valor de "importancia" de al menos el 75 % del rango de la imagen. Se aprecia que, si bien es capaz de detectar zonas de interés en todas las imágenes, falla al obtener todos los signos de retinopatía a pesar de apreciarse de forma cualitativa en la imagen de activaciones como es el caso de la imagen de la última fila.

3.5. Conclusiones

El programa Retisalud demostró un gran éxito cribando la población diabética de la Comunidad Autónoma de Canarias. Gracias a este programa, hasta 2015, el 32 % de la población diabética había sido cribada para retinopatía diabética.

En este trabajo se demuestra que es posible desarrollar un algoritmo con una precisión superior al 95 % a la hora de detectar casos de "RDNP moderada" o superior y una capacidad de clasificación de casos sanos al menos por encima del 70 %. También, se sugiere que este método automático fue capaz de encontrar signos de retinopatía diabética en 1.258 pacientes antes de que fueran detectados en una fecha posterior por parte de los médicos de familia, representando el 7.9 % del total de pacientes detectados por ellos.

Estos resultados respaldan la posibilidad de permitir dar una respuesta automática tras la captura de imágenes en lugar de esperar los 22 días de media

que se necesita para que AP evalúe la retinografía, permitiendo un reenvío más rápido al especialista en caso de ser necesario.

A pesar de estos resultados positivos, sigue existiendo un gran obstáculo, sobre todo a la hora de utilizar este método como herramienta de apoyo, que es la poca capacidad de explicación del algoritmo. Un algoritmo que responda "Patológico/No patológico"frente a un par de imágenes de una retinografía sin que explique, de algún modo, los motivos que le han llevado a tal conclusión tiene una utilidad limitada. Podrá utilizarse, tal vez, como herramienta de detección precoz frente a posibles casos patológicos, pero difícilmente pueda utilizarse como herramienta para asistir a un médico si este no otorga más información.

En este sentido, se estudiaron además, distintos métodos de interpretabilidad de modelos para la obtención de una explicación de las razones que llevan al sistema a dar una determinada clasificación. Específicamente, se estudió la posibilidad de utilizar la técnica de retro-propagación guiada y una versión multiescala del análisis de mapas de activaciones con el fin de resaltar aquellas zonas de la imagen que llevaron al sistema a dar con el resultado.

Se verifica que sí es posible dar una respuesta más informativa, dando como resultado, además de la clasifiación en "Patológico/No patológico", un análisis de qué zonas de la imagen son más reelevantes para esta decisión. No obstante, si bien los resultados son positivos en el sentido de que se consigue detectar correctamente signos de retinopatía diabética incluso en imágenes de casos leves donde las lesiones son más sutiles, debido a la propia naturaleza de estos métodos, estos no pueden utilizarse como métodos de detección/segmentación automática. A pesar de este inconveniente, esta herramienta resulta de especial utilidad, pues, permite explicar y justificar las decisiones que toma el método, permitiendo también así, servir de apoyo al médico.

Como resultado de la investigación realizada, se obtuvo una publicación [69] y otra que, a fecha de depósito de la presente tesis, se encuentra en revisión.

Capítulo 4

Reconstrucción de fase de frente de onda

4.1. Introducción

El frente de onda se puede interpretar como la distorsión que experimenta un haz de luz debido tras su interacción con un medio turbulento. Si el frente de onda presenta diferencias con respecto a una superficie de referencia (por ejemplo, un plano o una esfera) se dice que está aberrado o que presenta aberraciones.

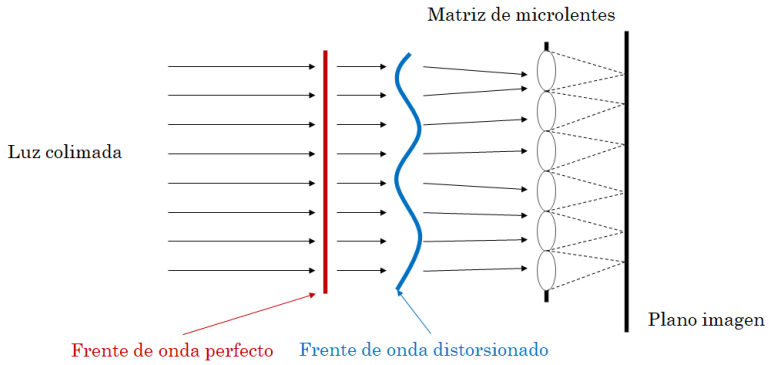


FIGURA 4.1: Luz colimada atravesando un medio distorsionado y siendo recogida por un sensor tras una matriz de microlentes.

En la figura 4.1 se presenta un diagrama explicativo donde un haz de luz colimada, que por definición es un haz de luz donde todos los rayos viajan paralelos entre sí e inciden a la vez sobre cualquier superficie perpendicular

a los mismos, atraviesa un medio turbulento que lo distorsiona (medio azul) y, tras atravesar este medio, los rayos de luz continúan con una dirección diferente a la original. Tener una medida del frente de onda una vez atravesado el medio proporcionaría una valiosa información, permitiendo no solo entender mejor el medio en cuestión, sino también, conociendo la dirección que tomarán los rayos de luz una vez hayan atravesado el medio, permitiendo una corrección de la imagen recogida en una cámara aplicando medios de corrección específicos.

De especial interés es la correcta medida del frente de onda en ámbitos como la astronomía o la oftalmología. En astronomía se utiliza el conocimiento del frente de onda en el ámbito de la óptica adaptativa. Ésta consiste en agregar un espejo deformable en el camino óptico del telescopio que corrige en tiempo real las aberraciones de la atmosfera que han sido previamente estimadas mediante un sensor de frente de onda. Por tanto, a mejor estimación del frente de onda de la atmósfera, mejor calidad de imagen se logrará. En oftalmología se utiliza en el campo de la aberrometría ocular, donde se estima el frente de onda ocular para corregir la visión en diferentes grados (por ejemplo, lentes de contacto o cirugía), para la observación de la retina *in vivo* o con propósitos diagnósticos o experimentales.

Quizás una de las principales dificultades a la hora de utilizar técnicas que requieran una correcta medición del frente de onda es la propia medida pues, actualmente, solo existen métodos indirectos para realizar su medición, por ejemplo a partir de sus derivadas. De los sistemas actuales de medición, el más extendido y considerado estándar *de facto* es el sensor *Shack-Hartmann* [70, 71]. Este sensor consiste en una matriz de microlentes delante del sensor de una cámara, donde cada micro-lente captura los rayos de luz de una determinada porción del frente de onda. En ausencia de aberración, cada micro-lente focalizaría la luz en un punto específico del sensor, cuya anchura viene determinada por el disco de Airy. Cuando existen aberraciones, cada microlente ya no focaliza la luz en un punto sino que creará una mancha más o menos extensa dependiendo de las aberraciones presentes. De manera simplificada, calculando la diferencia entre los centroides de las imágenes de cada microlente en el caso aberrado y sin aberraciones se puede obtener la derivada del frente de onda responsable de ese desplazamiento para cada microlente. En la figura 4.1 se muestra el esquema del funcionamiento del sensor.

Como resultado de la medición se obtendrán, por tanto, las derivadas parciales en ambas direcciones del frente de onda a medir. No obstante, aún es necesario realizar la reconstrucción de la fase de frente de onda a partir de sus derivadas, siendo este un problema bien conocido en óptica [72, 73, 74].Una forma de resolver este problema sería buscar aquella reconstrucción que minimice la diferencia entre las derivadas de la reconstrucción y las derivadas medidas. No obstante, esta aproximación requiere la realización de ciertas hipótesis sobre la fase a reconstruir, por lo que, en muchos casos, tanto la precisión como coste computacional de este método dependen de estas hipótesis iniciales. Dada las dificultades que plante este problema, numerosos métodos de resolución han sido propuestos en la literatura [75, 76, 77, 78, 79, 80].

Quizás uno de los métodos más utilizados para afrontar el problema de minimización es el de plantearlo como uno de Mínimos Cuadrados. Southwell propuso un método de mínimos cuadrados utilizando un modelo basado en diferencias centrales [75]. Este modelo asume que la relación de los puntos de la fase a reconstruir es siempre cuadrática. A pesar de ser cierta esta relación en muchos casos, particularmente cuando el frente de onda solo contiene bajas frecuencias, este modelo puede no funcionar correctamente en presencia de altas frecuencias. Huang y Asundi [81] propusieron una implementación iterativa del algoritmo de Southwell mejorando la precisión de la reconstrucción en fases con altas frecuencias a costa de un aumento del coste computacional. Li et al [82] propusieron una metodología basada en la utilización del teorema de Taylor y el error de truncado para deducir la relación entre los puntos discretos de la fase y las derivadas de la geometría de Southwell. Más tarde, Ren et al. [83] propusieron una implementación modificada de la misma metodología para reconstruir fases a partir de gradientes incompletos en dominios arbitrarios. Más recientemente, Huang et al. [84] propusieron un método basado en un modelo que asume que la relación entre cada punto de la geometría a reconstruir puede describirse con una curva diferenciable definida en porciones mediante polinomios, conocida como *spline*, mejorando los métodos de Southwell y de Li en los problemas sobre los que se evalúa.

Todos estos métodos nombrados comparten la necesidad de asumir una relación entre los puntos discretos de la geometría a reconstruir. La correcta selección de esta hipótesis condicionará, en muchos casos, no solo la precisión de la reconstrucción sino también el tiempo de cómputo. Es aquí donde el método propuesto en la presente tesis doctoral marca una diferencia: en lugar de asumir una geometría subyacente constante de forma independiente

al problema a resolver, se propone aprender dicha geometría para así lograr un modelo capaz de reconstruir de manera más exacta la fase de frente de onda a partir de sus derivadas.

4.2. Estado del arte

Como se mencionó anteriormente, uno de los métodos más extendidos para la obtención de la solución al problema de reconstrucción de fases a partir de sus derivadas parciales (o como parte de la solución) es el de Mínimos Cuadrados. Los métodos de reconstrucción de fases de diferencias finitas buscan, mediante Mínimos Cuadrados, minimizar el error entre las derivadas medidas y la que generaría la fase reconstruida siguiendo una hipótesis de modelo que explique la relación entre cada punto de la geometría que se reconstruye. La elección de esta relación subyacente condiciona la precisión de la reconstrucción de la fase a medir. Formalmente, los métodos de diferencias finitas basados en Mínimos Cuadrados que reconstruyen una fase $\phi \in \mathbb{R}^{M,N}$ con M y N como las resoluciones espaciales, se describen en la ecuación 4.1.

$$DZ = G \tag{4.1}$$

Con la matriz dispersa $D \in \mathbb{R}^{2MN,MN}$ como la matriz que define de que forma se relacionan los puntos de la fase a reconstruir. En otras palabras, la matriz D representa el modelo de geometría. La matriz $G \in \mathbb{R}^{2MN,1}$ determina como se relacionan los puntos de cada fila de D. $Z \in \mathbb{R}^{MN,1}$ representa la fase a reconstruir mediante Mínimos Cuadrados redimensionada a como se describe en la ecuación 4.2.

$$Z = \begin{bmatrix} \phi_{0,0} \\ \phi_{0,1} \\ \vdots \\ \phi_{M-1,N-1} \end{bmatrix} \tag{4.2}$$

4.2.1. Southwell

El método de Southwell asume una relación bicuadrática entre cada punto de la geometría a reconstruir. Esta relación entre la fase a reconstruir y la medida de gradientes se describe en la ecuación 4.3.

$$\begin{cases} \frac{S^x_{i,j}+S^x_{i,j+1}}{2} = \frac{\phi_{i,j+1}-\phi_{i,j}}{h} \\ \frac{S^y_{i,j}+S^y_{i+1,j}}{2} = \frac{\phi_{i+1,j}-\phi_{i,j}}{h} \end{cases} \tag{4.3}$$

donde $\phi \in \mathbb{R}^{M,N}$ es la fase a reconstruir, (i,j) son los índices de cada punto tal que $(i,j) \in \{(0,0)\dots(M-1,N-1)\}$, $h \in \mathbb{R}$ es el intervalo de muestreo y $S^x \in \mathbb{R}^{M,N}$ y $S^y \in \mathbb{R}^{M,N}$ son las derivadas parciales medidas. Dada esta relación, la matriz G de la ecuación 4.1 se construye siguiendo la ecuación 4.4.

$$G = \frac{1}{2}h \begin{bmatrix} S^x_{0,0}+S^x_{0,1} \\ S^x_{0,1}+S^x_{0,2} \\ \vdots \\ S^x_{M-1,N-2}+S^x_{M-1,N-1} \\ S^y_{0,0}+S^y_{1,0} \\ S^y_{0,1}+S^y_{1,1} \\ \vdots \\ S^y_{M-2,N-1}+S^y_{M-1,N-1} \end{bmatrix} \tag{4.4}$$

4.2.2. *Higher order finite difference* (Li et al.)

Li et al. [82] propusieron un método directo al considerar los términos de mayor orden de la expansión de Taylor. Este método es muy similar al método de Southwell, pues sigue la misma ecuación 4.1 que Southwell manteniendo igual la matriz D y modificando la matriz G siguiendo la ecuación 4.5.

$$G = \frac{1}{24}h \begin{bmatrix} 12(S^x_{0,0}+S^x_{0,1}) \\ -S^x_{0,3}+13S^x_{0,2}+13S^x_{0,1}-S^x_{0,0} \\ -S^x_{0,4}+13S^x_{0,3}+13S^x_{0,2}-S^x_{0,1} \\ \vdots \\ 12(S^x_{M-1,N-2}+S^x_{M-1,N-1}) \\ 12(S^y_{0,0}+S^y_{1,0}) \\ -S^y_{3,0}+13S^y_{2,0}+13S^y_{1,0}-S^y_{0,0} \\ -S^y_{4,0}+13S^y_{3,0}+13S^y_{2,0}-S^y_{1,0} \\ \vdots \\ 12(S^y_{M-2,N-1}+S^y_{M-1,N-1}) \end{bmatrix} \tag{4.5}$$

4.2.3. *Splines* (Huang)

Al igual que los métodos anteriores, en el método propuesto por Huang [84], la ecuación para resolver el problema se mantiene (ecuación 4.1) pero cambia la hipótesis de geometría que se asume. Específicamente, Huang propone una relación local basada en un spline. Esta relación se describe en la ecuación 4.6.

$$\begin{cases} \sum_{k=0}^{3} \frac{1}{k+1} c^{x}_{i,j,k} \Delta x^{k+1}_{i,j} = \phi_{i,j+1} - \phi_{i,j} \\ \sum_{k=0}^{3} \frac{1}{k+1} c^{y}_{i,j,k} \Delta y^{k+1}_{i,j} = \phi_{i+1,j} - \phi_{i,j} \end{cases} \tag{4.6}$$

donde $\Delta x_{i,j}$ y $\Delta y_{i,j}$ son los tamaños de salto en y y en x en la posición (i,j). $c^{x}_{i,j,k}$ y $c^{y}_{i,j,k}$ son los coeficientes del polinomio de k-ésimo orden empezando en (i,j). Estos coeficientes se determinan mediante un ajuste a un spline cúbico.

Con esta relación establecida, este método sigue la ecuación 4.1, cambiando la matriz G y manteniendo la matriz D. Siguiendo la relación zonal de la ecuación 4.6, la matriz G se describe en la ecuación 4.7.

$$G = \begin{bmatrix} \sum_{k=0}^{3} \frac{1}{k+1} c^{x}_{0,0,k} \Delta x^{k+1}_{0,0} \\ \sum_{k=0}^{3} \frac{1}{k+1} c^{x}_{0,1,k} \Delta x^{k+1}_{0,1} \\ \vdots \\ \sum_{k=0}^{3} \frac{1}{k+1} c^{x}_{M-1,N-2,k} \Delta x^{k+1}_{M-1,N-2} \\ \sum_{k=0}^{3} \frac{1}{k+1} c^{y}_{0,0,k} \Delta y^{k+1}_{0,0} \\ \sum_{k=0}^{3} \frac{1}{k+1} c^{y}_{0,1,k} \Delta y^{k+1}_{0,1} \\ \vdots \\ \sum_{k=0}^{3} \frac{1}{k+1} c^{y}_{M-2,N-1,k} \Delta y^{k+1}_{M-2,N-1} \end{bmatrix} \tag{4.7}$$

4.3. Método propuesto

La principal diferencia entre los métodos citados anteriormente es la forma en la que se estima la matriz G a partir de los gradientes medidos S^x y S^y. Estimar de forma fija esta matriz implica asumir una geometría subyacente a la fase que se pretende reconstruir. El hecho de utilizar una hipótesis fija de manera independiente al problema es una fuente de errores, pues, la relación entre una fase determinada y sus derivadas dependerá de la propia fase.

El método propuesto, en lugar de asumir una relación fija e independiente al problema a resolver, consiste en reconstruir la superficie deseada aprendiendo de un conjunto de muestras representativo del problema en cuestión. Específicamente, este método consiste en una red neuronal entrenada con datos sintéticos que siguen la distribución del problema objetivo, permitiendo así especializar la red neuronal en la reconstrucción de fases de frente de onda de un determinado tipo de problema.

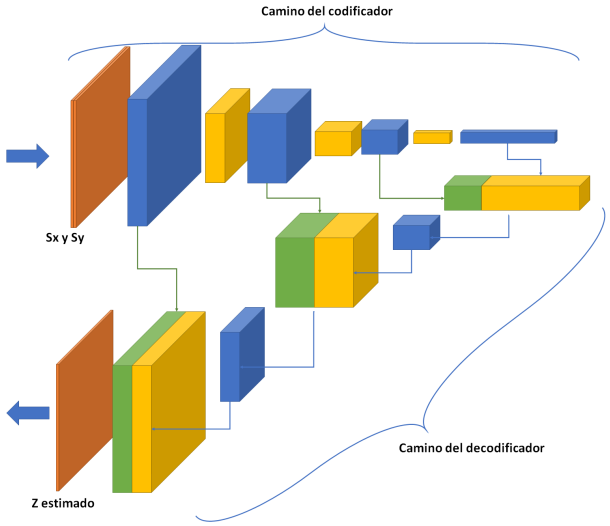


FIGURA 4.2: Arquitectura de red neuronal para la reconstrucción de fases a partir de los gradientes medidos.

En la figura 4.2 se presenta la arquitectura de la red neuronal propuesta. Está compuesta por dos partes: un codificador que extrae las características principales y un decodificador conectado con cada una de las resoluciones diferentes del codificador que reconstruye la fase. El codificador consiste en 3 capas convolucionales de tamaño 7x7 y profundidad 32, 64 y 64 respectivamente. Entre cada capa convoulucional se aplica una capa de *average-pooling* para así aumentar el campo receptivo de las capas posteriores. Tras estas 3 capas, se aplica una última capa convolucional con un kernel de 1x1 y profundidad 1024 antes de empezar la fase de decodificación. Este decodificador actúa a modo de espejo del codificador, con 3 capas convolucionales del mismo tamaño que las correspondientes del codificador, 64, 64 y 32 respectivamente. Las entradas a cada capa del decodificador son la salida de la capa anterior

redimensionadas con un factor 2 utilizando interpolación bilineal y concatenadas con la salida homóloga de la fase de codificación. Tras la tercera capa convolucional, se aplica una última convolución de 1x1 que transforma los 32 canales de la salida del codificador en la geometría estimada.

4.4. Simulaciones

Para el análisis del desempeño del método propuesto en comparación con los otros métodos del estado del arte primero se definió una metodología de generación de fases sintéticas junto con sus correspondientes derivadas, y luego, se diseñaron 3 análisis diferentes.

Respecto a la metodología de generación de fases, se implementó un sistema que genera una fase sintética como combinaciones de modos de Zernike siguiendo una distribución proporcionada. De cada modo de Zernike que contribuye a la fase a generar, se calcula sus derivadas parciales analíticas, para poder así obtener, junto con la fase sintética, sus derivadas parciales analíticas.

Para la validación de los tres análisis se generaron fases sintéticas de tamaño 21x21 utilizando combinaciones aleatorias de los primeros 30, 120, o de los 90 a 120 términos de Zernike, cada uno ponderado con un peso aleatorio entre -1 y 1. La figura 4.3 muestra un ejemplo visual de una fase generada sintéticamente utilizando los 120 primeros términos de Zernike. Adicionalmente, para poner la técnica en contexto, se generaron turbulencias atmosféricas asumiendo que estas siguen el espectro de Kolmogorov [85], imitando las condiciones de observación del Gran Telescopio de Canarias (GTC) con un diámetro de 10.4 m y un parámetro de Fried de 20 cm.

En el primer análisis se entrena la red neuronal con un conjunto de entrenamiento generado de la misma forma que los datos de validación.

El segundo análisis se diseña para simular la situación en la que los datos del conjunto de validación pertenecen a una distribución diferente a la utilizada para generar los datos de entrenamiento. Para ello, se entrena la red neuronal con datos generados con los primeros 120 términos de Zernike y se realiza la evaluación sobre un conjunto compuesto por los primeros 30 modos y por otro compuesto por los modos del 90 al 120.

El tercer análisis se diseña para simular una situación un poco más realista en la que la medida que se realiza no es perfecta, por ejemplo, el caso en que

el sensor de fase presenta cierto ruido, añadiendo ruido Gaussiano de 30 dB a los gradientes analíticos. En este caso, la red neuronal se entrena sin ruido y se realiza la evaluación con gradientes ruidosos, simulando así una situación en la que se presenta un cierto nivel de ruido inesperado que no fue tenido en cuenta durante el desarrollo de la red neuronal.

Los datos se generan de forma aleatoria siguiendo cada una de las configuraciones (primeros 30 modos de Zernike, primeros 120, del 90 al 120 y GTC) y los gradientes horizontales y verticales se calculan analíticamente. El conjunto de entrenamiento está formado con 30.000 instancias para cada configuración y el conjunto de validación con 1000 fases nunca antes vista por la red neuronal. En todos los casos, el entrenamiento y validación se realiza utilizando fases de tamaño 21x21.

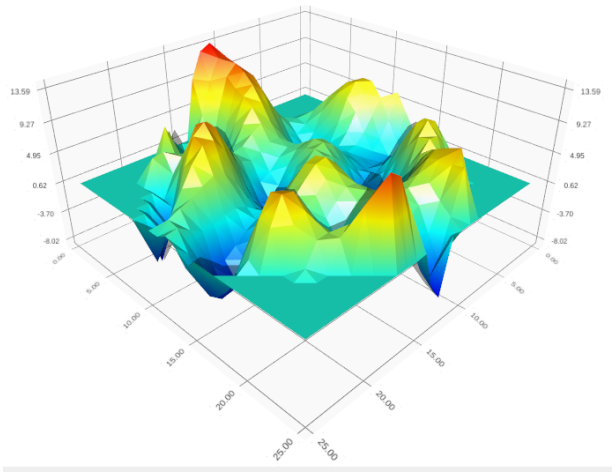
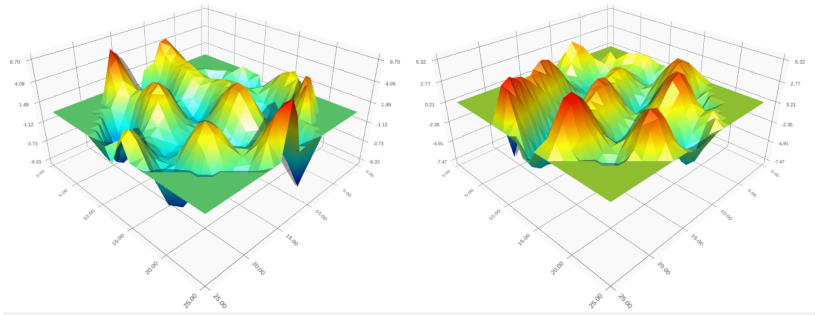


FIGURA 4.3: Ejemplo de fase sintética de tamaño 21x21 generada con una combinación aleatoria de los 120 primeros modos de Zernike.

Teniendo en cuenta que cada fase se encuentra en un rango diferente, se utiliza como medida de mérito para la validación el *root relative squared error* (RRSE) para poder comparar la calidad de la reconstrucción entre distintos métodos. Esta medida se define en la ecuación 4.8.



(A) Gradiente en x. (B) Gradiente en y.

FIGURA 4.4: Gradientes en x e y del fase de la figura 4.3.

$$RRSE = 100 \sqrt{\frac{\sum_{i,j} \left(\phi_{i,j} - \phi^*_{i,j} - (\overline{\phi} - \overline{\phi^*}) \right)^2}{\sum_{i,j} \left(\phi^*_{i,j} - \overline{\phi^*} \right)^2}} \tag{4.8}$$

con ϕ y ϕ^* como la fase estimada y el *ground-truth* respectivamente, y $\overline{\phi}$ y $\overline{\phi^*}$ como la media de la fase reconstruida y la media del *ground-truth* respectivamente.

4.5. Resultados

TABLA 4.1: Primer análisis: la red neuronal se entrena con la misma configuración que el conjunto de validación.

Zernike terms	Southwell mean/std	Li mean/std	Huang mean/std	Ours mean/std
1 to 30	8.66/2.02	4.55/1.25	**0.15/0.05**	0.63/0.14
90 to 120	41.53/8.37	35.19/8.2	10.13/3.2	**0.03/0.01**
1 to 120	40.42/8.18	34.16/8.0	9.73/3.1	**0.26/0.04**
GTC	33.63/8.48	28.66/7.7	8.57/2.82	**1.73/0.73**

La tabla 4.1 resume los resultados en términos de RRSE siguiendo la ecuación 4.8 donde, para cada caso, el conjunto de entrenamiento está compuesto por datos generados aleatoriamente y siguiendo la misma distribución que los datos utilizados para la validación. El método propuesto mejora los resultados de los demás métodos con los que se compara excepto para el caso de 1 a 30 modos de Zernike, donde el método de Huang muestra mejor desempeño. En caso de que la distribución del conjunto de validación no se corresponda

con la utilizada para entrenar, es de esperar que los resultados empeoren. En este sentido, se verificó que los resultados empeoran levemente al utilizar como datos de entrenamiento los primeros 120 modos de Zernike y como validación los conjuntos compuestos por los primeros 30 y los modos del 90 a 120. Específicamente, para los casos mencionados se obtiene un error de 0.98 $\pm$ 0.28 y 0.24 $\pm$ 0.03 respectivamente, aunque siguen siendo mejores resultados que el resto de métodos estudiados.

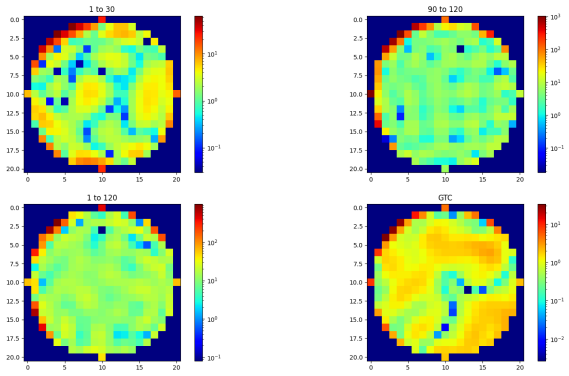


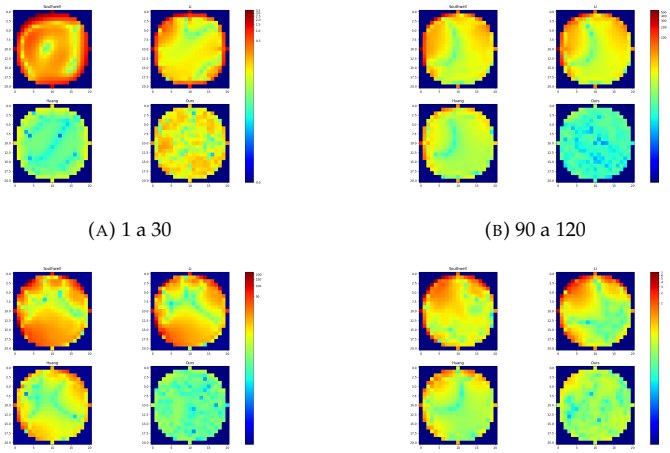
FIGURA 4.5: Ejemplos de fases a reconstruir representativos de cada experimento de la tabla 4.1. Representado en escala logarítmica.

La figura 4.5 muestra una fase representativa de cada experimento de la tabla 4.1 y la figura 4.6 muestra el error de reconstrucción de cada método para las fases de la figura 4.5. Para lograr una mejor visualización, la figura 4.5 se muestra en escala logarítmica mientras que la figura 4.6 se muestra con una normalización exponencial siguiendo la ecuación 4.9.

$$z_{i,j} = \theta_{i,j}^{\gamma} \tag{4.9}$$

Donde $\theta_{i,j}$ es el valor a normalizar, $z_{i,j}$ es el valor normalizado y γ es el factor de normalización.

Para simular una situación en la que se utiliza una cámara real, se realiza el experimento de añadir ruido a las derivadas que se utilizan como entrada a la red neuronal. La tabla 4.2 resume los resultados para el caso de tener un ruido Gaussiano de 30 dB en la señal de entrada. Se verifica que el método



(A) 1 a 30 (B) 90 a 120

(C) 1 a 120 (D) GTC

FIGURA 4.6: Error de reconstrucción de cada ejemplo de la figura 4.5 para cada método. Representado con normalización exponencial siguiendo la ecuación 4.9 con $\gamma = 0{,}1$.

TABLA 4.2: Segundo *test*: La red neuronal se entrena con datos generados con los primeros 120 modos de Zernike sin ruido y validada sobre fases con ruido Gaussiano de 30 dB.

Zernike terms	Southwell mean/std	Li mean/std	Huang mean/std	Ours mean/std
1 to 30	8.68/1.96	4.83/1.22	**2.00/0.37**	2.66/0.53
90 to 120	40.93/8.17	34.69/8.01	10.89/3.02	**5.53/0.88**
1 to 120	40.69/8.65	34.57/8.49	10.99/3.07	**5.48/0.79**

propuesto, si bien empeora respecto al caso de no tener ruido, mejora al resto de métodos excepto para el caso del conjunto de validación compuesto por los primeros 30 modos de Zernike donde el método de Huang sigue teniendo mejores resultados.

4.6. Conclusiones

En la bibliografía, existen grandes diferencias en la metodología utilizada para testear los diferentes métodos de integración de derivadas de fase de frente de onda. En muchos casos, se encuentra que se testean únicamente sobre un conjunto muy limitado (generalmente 1 o 2) de superficies sintéticas bien conocidas y caracterizadas. Al realizar pruebas tan limitadas, en muchos casos

se presentan métodos que se comportan muy diferentemente dependiendo de la naturaleza del problema a resolver.

En relación a este problema, en el presente trabajo se propuso una metodología de análisis que consiste en validar cada método sobre un conjunto de superficies generadas a partir de combinaciones de modos de Zernike. Esta metodología permite validar diferentes métodos sobre un conjunto arbitrario de superficies generadas siguiendo una distribución deseada, desde una distribución que simule turbulencias atmosféricas a una completamente aleatoria. Además de generar superficies sintéticas, teniendo en cuenta que el resultado del método de integración podría estar fuertemente correlacionado al método utilizado para obtener las derivadas de la fase original (por ejemplo, un método de integración que asuma una geometría basada en diferencias centrales se beneficiaría de unas derivadas generadas a partir de la fase original utilizando diferencias centrales), se generan las derivadas analíticas en ambas direcciones junto con la superficie a reconstruir para, así, tener una simulación más objetiva.

Esta metodología, si bien es más completa que simplemente verificar con un número finito de superficies específicas, puede fallar a representar los fenómenos físicos en los que es necesario reconstruir la fase a partir de sus derivadas, pues, no todos están necesariamente bien representados con un número finito de polinomios de Zernike. A pesar de que, teóricamente, cualquier fase puede ser representada como una combinación de polinomios de Zernike, en la práctica, los términos de muy alto orden son computacionalmente costosos. Incluso así, esta metodología puede ser representativa de algunos campos relacionados con la óptica, como por ejemplo oftalmología o astronomía, donde se acepta que 120 polinomios de Zernike son suficientes para representar la naturaleza del problema [86, 87].

Adicionalmente, se demostró que el método de integración propuesto consigue mejores resultados incluso cuando la distribución de los datos de validación no se corresponde con la de los datos de entrenamiento. Al ser un método basado en *deep learning*, la calidad de los resultados está directamente relacionada con la calidad de los datos y qué tan representativos estos son del problema objetivo a resolver. Por tanto, el desempeño del método empeorará si un modelo entrenado en un contexto específico se utiliza en uno diferente. No obstante, el método propuesto es escalable en el sentido que el conjunto de entrenamiento puede modificarse para ajustarlo a la naturaleza del problema a resolver, esté descrito o no en términos de polinomios de

Zernike.

Además de las metodologías con las que se compara el método propuesto, existen otras que no solo se basan en Mínimos Cuadrados. Por ejemplo, existen los algoritmos basados en la Transformada Discreta de Fourier (DFT) para el cálculo de valores de frente de onda en el dominio de frecuencia. En este sentido, Poyneer et al [88] propusieron una metodología basada en el uso iterativo de transformadas de Fourier extendiendo las derivadas al borde de la pupila para, así, poder aplicar algoritmos de integración basados en la DFT para el caso de pupilas circulares. El principal problema de los métodos basados en DFT es que no logran buenos resultados sobre pupilas circulares, y para incrementar la precisión de estos métodos es necesario aumentar el *padding* de la matriz inicial, incrementando así el coste computacional. Por estas razones se han elegido los métodos con los que se compara el método propuesto al ser representativos del estado del arte. Southwell es, probablemente, el más utilizado, mientras que el de Li y el de Huang pueden ser considerados como nuevas y recientes variaciones del algoritmo de Southwell.

Respecto al tiempo de cómputo, si bien es necesario entrenar la red neuronal, una vez entrenada, la inferencia requiere menos de 10 ms. sobre una GPU Nvidia 1050 Ti, permitiendo, así, utilizar este método en sistemas de tiempo real.

Teniendo en cuenta lo expuesto anteriormente, se demuestra que el método propuesto es capaz de reconstruir superficies a partir de sus derivadas una vez ha convergido mejorando a los otros métodos con los que se compara en las simulaciones sintéticas presentadas, incluso bajo la presencia de ruido.

Como resultado de la investigación realizada en esta línea, se obtuvo una publicación [89].

Capítulo 5

Depth from focus

5.1. Introducción

La información de distancia de cada punto de la escena a una referencia (la propia cámara, preferiblemente) es, quizás, una de las piezas claves para entender mejor una escena real [90]. Por ejemplo, esta información puede ser útil para localizar objetos con mayor precisión [91, 92, 93], para construir una malla tridimensional [94, 95, 96, 97], o para propósitos artísticos como puede ser la cinematografía [98].

Para obtener esta información existen varios métodos en la literatura. Pueden clasificarse como "activos" aquellos que requieren de *hardware* adicional que, por ejemplo, emite luz para obtener así las distancias [99, 100, 98, 101, 102]; o, como "pasivos" aquellos que estiman las distancias utilizando únicamente la luz recibida.

Algunos de estos métodos pasivos utilizan dos cámaras para crear una geometría binocular (estéreo) para estimar distancias [103, 104, 105]. Otros se basan en el movimiento de la cámara, capturando múltiples imágenes de una misma escena desde diferentes puntos de vista para luego, mediante algoritmos de *structure from motion* (SFM), estimar la posición tridimensional de cada objeto de la escena [106, 107, 108]. Otros, tratan de extraer la información de distancias a partir de imágenes de una única cámara (estimación de distancias monocular). El presente capítulo se centra específicamente en la estimación de distancias monocular, marcando como restricción el uso de una única cámara para la realización de dicha estimación.

La estimación de distancias monocular es una técnica realmente desafiante puesto que, necesita detectar "pistas" visuales, sutiles en muchos casos, y

explotar conocimiento previo para poder dar una inferencia robusta. Recientes avances en *deep learning* abren una serie de alternativas capaces de captar estas pistas y este conocimiento previo necesario [109, 110, 111, 112]. No obstante, los métodos basados en *deep learning* siguen siendo muy dependientes del conocimiento previo del algoritmo. Por tanto, en muchos casos es necesario utilizar modelos de redes neuronales relativamente pesados para lograr buenos resultados. Además, al basarse en conocimiento previo, cuando se presentan escenas poco comunes a las vistas durante el proceso de entrenamiento o con pocas pistas visuales se produciría una estimación pobre. Por último, al no contar estos métodos con ninguna referencia física no podrán proveer en ningún caso un valor de distancias cuantitativo y absoluto, sino únicamente podrán inferir un valor relativo.

Otro método para la obtención de distancias a partir de una cámara estática es *depth from focus* [113]. Este método se basa en la captura de múltiples imágenes de la misma escena variando la posición de enfoque de la cámara, obteniendo así lo que se denomina "*focal stack*", con tantos "planos" como imágenes fueron capturadas. Tras la obtención de esta información, se analiza el enfoque de cada píxel para así estimar el plano de pertenencia, y, en consecuencia, las distancias. Uno de los principales inconvenientes de esta técnica es su dependencia de la óptica del sistema. Por ejemplo, si la cámara tiene demasiada profundidad de campo, todos los planos del *focal stack* estarían siempre enfocados, imposibilitando estimación alguna de distancias; por el contrario, si la profundidad de campo es limitada, se requerirían demasiados planos para barrer todo el rango de enfoque, volviendo al algoritmo computacionalmente inviable.

La presente tesis aborda el problema de la estimación de distancias con *depth from focus* utilizando *deep learning*. Concretamente, se pretende diseñar un método que sea capaz de resolver los dos problemas principales de esta técnica: la estricta dependencia de los métodos clásicos de *depth from focus* con la óptica del sistema y el movimiento entre imágenes de un mismo *focal stack* inherente al propio método de captura.

5.2. Estado del arte

Los algoritmos convencionales de *depth from focus* estiman el valor de distancia de cada pixel analizando qué tan enfocado está dicho pixel en cada uno

de los planos del *focal stack*, estableciendo como medida de distancia la que le corresponde al plano en el que dicho pixel se encuentra mejor enfocado.

Dicho esto, un algoritmo de *depth from focus* consiste, *grosso modo*, en dos partes: extracción de información de enfoque mediante un operador y cálculo de distancia de cada píxel en función de su medida de enfoque.

Si bien es un método simple a alto nivel, tiene una serie de particularidades prácticas que hacen este tipo de algoritmos particularmente difíciles:

- Existe una gran variedad de métodos de extracción de medida de desenfoque, dando cada uno de ellos un resultado diferente.
- Los operadores de desenfoque son, en general, muy sensibles al ruido.
- Las zonas en las que no es posible extraer información de enfoque serán difícilmente bien estimadas.

En esencia, un operador de desenfoque es un operador que mide las altas frecuencias de la imagen. Los más utilizados son los basados en primeras y segundas derivadas que miden en la práctica el nivel de cambio de un pixel respecto a su vecindario.

En general, al medir las diferencias de un pixel respecto a sus vecinos, se tiene especial sensibilidad al ruido, dando lugar a situaciones en las que el propio ruido del sensor de la cámara es el responsable de dar falsos máximos de enfoque. Este efecto se magnifica en imágenes con zonas en las que no es posible extraer ningún tipo de información de distancias, por ejemplo, una imagen con una zona de color uniforme; en este caso, se obtendría únicamente la información de ruido como información de enfoque, dando lugar a un resultado claramente erróneo.

A pesar de estos inconvenientes, existen métodos en la literatura que afrontan este problema mediante diferentes técnicas. [114] propone un método basado en la variación total utilizando varios mapas de distancias iniciales obtenidos a partir de diferentes operadores de desenfoque. De forma similar, [115] propone el método "Variational Depth From Focus" (VDFF) que aborda el problema utilizando un término de "fidelidad" no convexo y un regularizador convexo para obtener mapas de distancias robustos.

A pesar de obtener mejores resultados con estos métodos citados, estos han de ser fuertemente parametrizados para cada situación, imagen, cámara y/o

contexto, pues, muchos se basan en conocimiento previo de la escena a estimar y parámetros ajustables como pueden ser los tamaños de *kernels*, operador de desenfoque o regularizadores. Por lo que, si bien se pudiera resolver algún conjunto específico de situaciones mediante estos algoritmos, han de ser correctamente ajustados para obtener resultados positivos, impidiendo así la generalización a distintos tipos de escenas, cámaras o resoluciones.

Para afrontar el problema de la generalización, las nuevas técnicas basadas en *deep learning* son especialmente útiles, pues, pueden aprender a extraer características generalizables para lograr una estimación de distancias robusta. En esta línea, el primer trabajo en seguir esta aproximación fue [116], mostrando resultados interesantes sobre imágenes sintéticas. Con los recientes avances del *deep learning*, aparecieron nuevos métodos basados en arquitecturas más complejas, por ejemplo, el método de Hazirbas et al [117].

No obstante, los métodos del estado del arte basados en *deep learning* siguen teniendo problemas a la hora de generalizar. En el caso del método de Hazirbas, por un lado, está restringido a un número fijo de planos por *focal stacks*, obligando a entrenar modelos específicos para situaciones diferentes. Por otro, arroja como resultado valores absolutos, lo que provoca que la precisión de la estimación de distancias sea muy dependiente de la similitud que tenga la distribución del conjunto de entrenamiento con la distribución de las imágenes que se generan para la inferencia.

5.3. El método

5.3.1. Conjunto de datos

Los algoritmos de *depth from focus* en general se ven muy afectados por la óptica de la cámara. Por tanto, si el conjunto de datos de entrenamiento estuviera formado únicamente por imágenes tomadas de una misma cámara, el modelo resultante fallaría a la hora de generalizar a distintas cámaras. Más aún, cámaras con características ópticas diferentes tendrán una profundidad de campo diferente por lo que, dependiendo de la óptica de cada cámara se necesitará un número diferente de planos y posiciones de enfoque para barrer todo el rango deseado.

Dicho esto, el conjunto de datos ideal será aquel que cuente, no solo con *focal stacks* de escenas reales de suficiente resolución y calidad de imagen y *ground truth*, sino también con variedad de escenas, condiciones de luz, parámetros

de cámara, posiciones de enfoque y número de planos. En la literatura no se ha encontrado ningún conjunto de datos público que cumpla con estas características.

El principal problema encontrado es la falta de conjuntos de datos compuestos por *focal stacks* de escenas reales y sus correspondientes distancias. Este problema se debe principalmente a la dificultad que supone extraer distancias de una escena real. En este sentido, el único conjunto de datos encontrado formado por este tipo de instancias fue *mobile Depth From Focus* (mDFF) [117]. Sin embargo, la resolución espacial es baja y la información de distancias es muy dispersa y poco precisa, imposibilitando su uso para entrenamiento.

Al verificar la falta de este tipo de conjuntos de datos, se procede a buscar otros que estén formado por imagen e información de distancias (RGB-D); para así, generar sintéticamente los desenfoques y formar un *focal stack* sintético. Como punto de partida, se exploraron múltiples conjuntos de datos públicos resumidos en la tabla 5.1.

	Real	Resolución	Precisión	Secuencias	Nº de instancias
DDFF[117]	Si	552x383	Media	No	720
Diode[118]	Si	1024x768	Media	No	27858
FlyingThings[119]	No	960x540	Perfecto	Si	22390
DIML[120]	Si	1344x756	Media	Si	1500*
NYuv2[121]	Si	640x480	Media	No	1449
RedWebV1[122]	Si	<500x500	Media	No	3600
Sintel[123]	No	1024x436	Perfecto	Si	1064
Middlebury 2005[124]	Si	∼1300x1100	Alta	No	81**
Middlebury 2006[125]	Si	∼1300x1100	Alta	No	243**
Middlebury 2014[126]	Si	∼2900x2000	Alta	No	69***
Mobile DFF[117]	Si	552x309	Baja	No	181

TABLA 5.1: Resumen delos conjuntos de datos RGB-D públicos analizados. (*) 1500 imágenes es el número que contiene el conjunto"limpio" de entrenamiento, el conjunto de entrenamiento "crudo" es de órdenes de magnitud superior pero no está igual de depurado. (**) Middlebury 2005 y 2006 cuenta con 3 exposiciones diferentes para 3 condiciones de iluminación diferentes para cada imagen, resultando en 9 instancias por cada imagen. (***) Middlebury 2014 cuenta con 3 condiciones de iluminación por cada imagen.

De los explorados, se seleccionaron Middlebury[124, 125, 126], DIML[120] y FlyingThings[119] por tener características claramente diferentes: Middlebury está compuesto por imágenes de escenas reales en alta resolución (desde 1.5 MP hasta 6 MP) con un información de distancia de alta calidad, pero cuenta con un número limitado de instancias. DIML cuenta con un gran número de muestras de imágenes de escenas reales de al menos 1.5 MP, sin embargo, la información de distancias posee ciertos errores, principalmente en los bordes. Por último, FlyingThings, a diferencia de los anteriores, es un conjunto de datos completamente sintético y, por tanto, con información de distancias exacta.

Generación de distancias

En el conjunto de datos de Middlebury la información de distancias se da en disparidad. Para traducir de información de disparidad en píxeles a metros se utiliza la ecuación:

$$D_i = \frac{fb}{Disp_i} \tag{5.1}$$

Con b la línea de base (distancia entre las cámaras), f la longitud focal de la cámara, $Disp \in \mathbb{R}^{W,H}$ el mapa de disparidades y $D \in \mathbb{R}^{W,H}$ el mapa de distancias, con W y H como la resolución espacial en ancho y alto respectivamente.

Puesto que las oclusiones existen de forma inherente a un sistema estéreo, la información de disparidades está incompleta en aquellas zonas donde se produce una oclusión. Si no se rellenara estas zonas, aparecerían artefactos en las imágenes desenfocadas con estas distancias. Por tanto, para poder evitar estos artefactos, se rellenan previamente las zonas sin información del mapa de distancias. Para ello se utiliza el algoritmo "*fast bilateral solver*" [127]. Este algoritmo está diseñado para rellenar zonas de una imagen utilizando una primera estimación, una imagen de guía y un mapa de confianza que indica cuanta certidumbre hay en la imagen de primera estimación para cada píxel. Este algoritmo se aplica utilizando como estimación inicial el mapa de distancias, como guía la propia imagen a color, y, como mapa de confianza, la máscara de oclusiones.

Para los conjuntos de datos de DIML y FlyingThings este procedimiento no es necesario porque ya proveen estimaciones de distancias densas en metros.

Generación de *focal stacks*

Con las distancias correspondiente a cada imagen a color se tiene la información necesaria para poder generar sintéticamente imágenes desenfocadas a diferentes distancias y así generar *focal stacks*. Para generar cada *focal stack*, primero se genera el vector ordenado $s \in \mathbb{R}^N$ que contiene las diferentes posiciones de enfoque de cada plano del *focal stack* a generar, siendo N el número de planos. Idealmente, para minimizar la incertidumbre de la predicción de un algoritmo de *depth from focus*, la profundidad de campo de cada plano del *focal stack* no debería de solaparse con la de ningún otro plano. Sin embargo, esta situación ideal es difícilmente alcanzable en escenarios reales a causa de posibles incertidumbres provocadas por partes mecánicas, eléctricas o de cualquier otra naturaleza. Por tanto, para emular este tipo de incertidumbre, en lugar de calcular las posiciones de enfoque ideales que evitan este solape, el vector s es calculado de forma aleatoria; permitiendo así, generar *focal stacks* que solapen planos o que nunca enfoquen objetos muy cercanos, lejanos o en posiciones intermedias.

Para generar un *focal stack* a partir de una imagen a color ($img \in \mathbb{R}^{W,H}$) y un mapa de distancias ($D \in \mathbb{R}^{W,H}$), se utiliza el modelo de lente delgada [128] con el objetivo de emular una cámara sintética. Los parámetros necesarios para definir este tipo de cámara son: la longitud focal (f), número f ($f\#$), el diámetro de apertura ($D = \frac{f}{f\#}$) y el tamaño del píxel de la cámara (pxs). El círculo de confusión ($CoC(x)$) para una distancia x y una distancia de enfoque u se define como:

$$CoC(x,u) = D\frac{1}{\frac{1}{f}-\frac{1}{u}}\left|\frac{1}{u}-\frac{1}{x}\right|\frac{1}{pxs} \tag{5.2}$$

Para generar el desenfoque de cada plano, primero se descompone el rango de distancias en capas discretas $l \in \mathbb{R}^L$, utilizando en nuestro caso $L = 100$. Luego, para capa l, se aplica un desenfoque Gaussiano a la imagen a color utilizando como desviación típica $\sigma(x,u) = CoC(x,u)0{,}3$.

$$layer^v = G(img, \sigma(l_v, s_p)) \tag{5.3}$$

Con $G(img, \sigma(l_v, s_p))$ como el desenfoque Gaussiano aplicado a la imagen a color con desviación típica $\sigma(l_v, s_p)$ y, $layer^v \in \mathbb{R}^{W,H}$ siendo la capa que

simula el desenfoque de un objeto que se encuentra a una distancia l_v, con una cámara enfocando a una distancia s_p.

Finalmente, se mezclan mediante interpolación lineal las capas generadas, obteniendo $P^p \in \mathbb{R}^{W,H}$ como el plano desenfocado a la distancia de enfoque s_p:

$$P_i^p = layer_i^{j_i} \frac{|D_i - l_{k_i}|}{|l_{j_i} - l_{k_i}|} + layer_i^{k_i} \frac{|D_i - l_{j_i}|}{|l_{j_i} - l_{k_i}|} \tag{5.4}$$

Donde $j_i \in \mathbb{R}$ y $k_i \in \mathbb{R}$ son los índices de las dos capas más cercanas al píxel D_i calculados siguiendo las ecuaciones:

$$j_i = \{p \mid |l_p - D_i| = \min_{p'} |l_{p'} - D_i|\} \tag{5.5}$$

$$k_i = \{p \mid |l_p - D_i| = \min_{p' \neq j_i} |l_{p'} - D_i|\} \tag{5.6}$$

Esta aproximación por capas es similar al trabajo de Kraus et al [129], diferenciándose en la mezcla, ya que aquí se utiliza interpolación lineal.

En la figura 5.1 se muestra un ejemplo de un *focal stack* del conjunto de datos de Middlebury generado con este método.

FIGURA 5.1: *Focal stack* generado a partir de una imagen de Middlebury [124], desde el enfoque más cercano (arriba-izquierda) al más lejano (abajo-derecha).

Generación del mapa de índices de planos

Como se ha mencionado anteriormente, la precisión de los algoritmos de *depth from focus* depende fuertemente de la óptica del sistema. Por tanto, se introduce el término "mapa de índices de planos" $I \in \mathbb{R}^{W,H}$ que contiene la información del plano al que pertenece cada píxel, tomando I valores en el rango $[0, N-1]$. Este mapa de índices de planos permite representar las distancias en función de la configuración del *focal stack*, logrando así una independencia de la óptica del sistema.

Para generar estos mapas de índices de planos a partir del mapa de distancias se interpola linealmente los índices del vector s, de forma similar a como se hizo para P^p:

$$I_i = j_i \frac{|D_i - s_{k_i}|}{|s_{j_i} - s_{k_i}|} + k_i \frac{|D_i - s_{j_i}|}{|s_{j_i} - s_{k_i}|} \tag{5.7}$$

En este caso, los índices j_i y k_i son los dos índices de planos con la distancia de enfoque más cercana a D_i.

5.3.2. Arquitectura

Codificador 2D - Decodificador 3D en multiescala

Se propone una arquitectura consistente en dos partes: un extractor de características 2D siamés y un decodificador 3D. En la figura 5.2 se ilustra la arquitectura de la red neuronal.

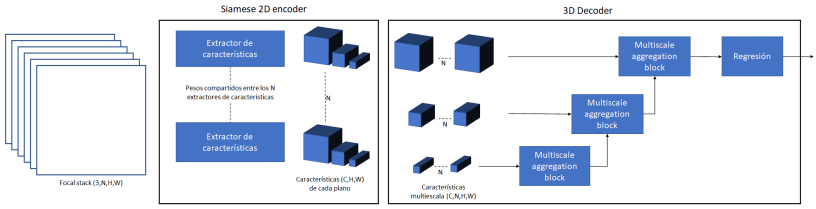


FIGURA 5.2: Arquitectura de red para *depth from focus*.

Codificador 2D siamés

La primera parte del modelo recibe el nombre de "Codificador 2D siamés" ("*Siamese 2D encoder*") puesto que es un extractor de características de imágenes 2D que se aplica a cada imagen de entrada del *focal stack*, utilizando los mismos pesos para cada plano. Este extractor de características obtendrá

la información de desenfoque de cada plano del mismo modo en que los métodos de estimación de distancias monoculares del estado del arte extraen la información de distancias de una única imagen. La salida serán las características multiescala antes de la capa de producto interno que se ocupa de traducir los mapas de características en información de distancia en una red neuronal de estimación de distancias monocular [112]. Obteniendo así, para cada plano del *focal stack*, un conjunto de mapas de características multiescala.

Una de las principales ideas de esta arquitectura es la característica "siamésa" del codificador 2D; es decir, el hecho de que los pesos de la red neuronal que se utiliza para extraer los mapas de características de cada plano son compartidos entre sí. Al compartir los pesos de cada subred neuronal aplicada a cada plano del *focal stack*, se puede utilizar la misma arquitectura de red para diferentes números de planos, evitando así utilizar configuraciones específicas para diferentes cámaras o situaciones. De lo contrario, cabría de entrenar específicamente una nueva red neuronal para cada número diferente de planos que se quiera utilizar.

Por otro lado, al utilizar los mismos pesos para cada plano del *focal stack* se evitan posibles sobre-ajustes para posiciones de enfoque específicas.

Decodificador 3D

La segunda parte del modelo recibe el nombre de "Decodificador 3D" ("*3D decoder*") puesto que es el encargado de decodificar las características obtenidas por el codificador 2D, traduciéndolas a un mapa de índice de planos mediante convoluciones 3D. Este decodificador apila las características obtenidas por el codificador, obteniendo así tensores con la forma (C, N, H, W) para cada escala, con C como el número de canales. Luego, mezcla secuencialmente cada volumen de características utilizando el módulo *Multiscale aggregation block* (figura 5.3).

Este bloque se encarga de mezclar las características de diferentes resoluciones (excepto por el primer bloque que solo se ejecuta sobre una única resolución). Inspirado en el método de *sharp mask* [130], primero escala el volumen de característica de menor resolución, adapta los canales del de mayor resolución y concatena a lo largo de la primera dimensión para luego aplicar una serie de bloques residuales como los de ResNet [20] (mostrado en la figura 5.3), de forma similar a la implementación de RefineNet [131] para la

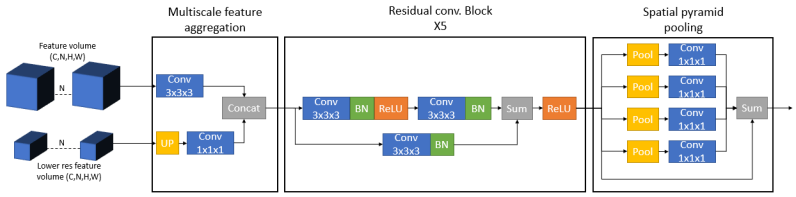


FIGURA 5.3: *Multiscale aggregation block.*

segmentación de imágenes. Finalmente, al final de cada bloque, para capturar la información global, se aplica el módulo de *spatial pyramid pooling* [34], similar al que se utiliza en la red neuronal de segmentación PSP-Net [132] presentado en la sección 2.4.4.

Las convoluciones del decodificador son todas 3D, para así analizar no solo en XY, sino también a lo largo de los distintos enfoques.

Regresión

El resultado final del decodificador 3D es un tensor con forma (C, N, H, W). En lugar de obtener el mapa de distancias utilizando capas de producto interno, se propone una aproximación diferente: primero reducir el número de canales a 1 para luego aplicar el módulo de regresión [133] sobre la dimensión de profundidad. La principal ventaja de esta aproximación es la capacidad de poder construir un decodificador que sea independiente del número de planos del *focal stack*, del mismo modo que el codificador 2D lo es. Esta independencia se consigue al tratar los planos en la dimensión de profundidad en lugar de tenerlos en la dimensión de canales, lo que requeriría capas de producto interno de tamaño fijo como sucede en [117].

Como consecuencia de esta arquitectura, se obtiene una independencia total del número de planos del *focal stack*; permitiendo así entrenar con un número de planos variable en cada entrada, abriendo la posibilidad de aprender mejor las características necesarias para extraer la información de distancias a partir del enfoque.

5.3.3. Implementación

Datos

Como se ha dicho anteriormente, el conjunto de datos de Middlebury cuenta con imágenes de alta resolución y estimaciones de distancias de alta calidad; sin embargo, este conjunto de datos cuenta con un número limitado de muestras, un total de apenas 50 imágenes de escenas diferentes. Por tanto, será necesario la aplicación de técnicas de aumento artificial de datos para poder entrenar con este conjunto de datos.

En primer, lugar, se simula una cámara sintética eligiendo aleatoriamente la longitud focal, apertura y tamaño de píxel. Luego, se genera el vector s calculando las posiciones de enfoque ideales que permitirían un barrido de todo el rango de enfoque sin solapar ningún plano. De los planos generados, se seleccionan aleatoriamente entre 4 y 10 posiciones diferentes, que serán las utilizadas para generar el *focal stack*. Aplicando este procedimiento se logra generar muestras con cámaras diferentes con un número de planos variable, entre 4 y 10 planos.

Cabe destacar que el hecho de que se restrinja el número de planos luego de haber definido una cámara puede imposibilitar la realización de un barrido completo del rango de enfoque, dando lugar a situaciones en que se tengan ciertas partes de la escena nunca enfocadas. Como se ha dicho anteriormente, este tipo de situaciones son posibles en escenarios reales, por tanto, los datos son generados con esta característica. Para acercar más los datos generados a situaciones reales, se añade ruido Gaussiano a las posiciones de enfoque de los planos seleccionados (vector s) simulando así una situación aún más realista en la que no se enfoca exactamente donde se pretende. De este mismo modo, se consigue aumentar también la variedad del conjunto de datos generado al incrementar la posibilidad de generar instancias diferentes.

Tras aplicar este proceso a cada escena y cada configuración de iluminación del conjunto de datos, se consigue convertir un conjunto de datos de 50 muestras a uno de 2100, pues, se generan *focal stacks* de 7 números de planos diferentes. Cada imagen tiene al menos 3 condiciones de iluminación diferentes y cada una se espeja horizontalmente aleatoriamente.

Una vez generado el conjunto de datos, se aplican técnicas de aumento de datos tradicionales como el añadido de ruido de *Poisson*, modificación de saturación y contraste, escalado y recorte.

Además de utilizar el conjunto de datos de Middlebury para entrenar, se realizaran experimentos con los conjuntos de datos de DIML y FlyingThings, puesto que, cada uno de ellos cuenta con importante diferencias que pueden potencialmente afectar el desempeño del método. FlyingThings es exacto pero formado por escenas completamente sintéticas y poco realistas. DIML está formado por imágenes reales pero los mapas de distancias tienen errores notables, sobre todo en los bordes.

Función de costes

Sea M el conjunto de píxeles válidos en el *ground-truth*. Sea $d \in \mathbb{R}^{W,H}$ la predicción del modelo y $I \in \mathbb{R}^{W,H}$ el mapa de índices de planos definido en la sección 5.3.1. Se define la función de costes de índices de planos $\mathcal{L}_{idx}$ como sigue:

$$\mathcal{L}_{idx}(d, I) = \frac{1}{|M|} \sum_{i \in M} |d_i - I_i| \tag{5.8}$$

Tal y como se denota en otros trabajos de la literatura [112], si se agregara una función de costes adicional que sea consciente de los bordes mejoraría la calidad del resultado. En consecuencia, tal y como se hace en [111, 112], se añade el "*gradient matching term*":

$$\mathcal{L}_g(d, I) = \frac{1}{|M|} \frac{1}{K} \sum_{k}^{K} \sum_{i \in M} |\nabla_x R_i^k| + |\nabla_y R_i^k| \tag{5.9}$$

Donde R^k es $|d_i - I_i|$ para la escala k, K el número máximo de escalas diferentes a evaluar (4 en este caso) y, ∇_x y ∇_y los gradientes horizontales y verticales respectivamente. Finalmente, la función de costes final es la composición de la función de costes de índices de planos y el *gradient matching term*:

$$\mathcal{L}(d, I) = \mathcal{L}_{idx}(d, I) + \gamma \mathcal{L}_g(d, I) \tag{5.10}$$

Con γ a 0,5. Con esta función de costes, el modelo se entrena utilizando el optimizador ADAM [68], un optimizador basado en gradiente descendiente que utiliza la información de momentos de primer y segundo orden, con $\alpha = 0{,}001$, $\beta 1 = 0{,}9$ y $\beta 2 = 0{,}999$.

5.4. Experimentos cuantitativos

Los experimentos se centran en:

- Analizar los resultados variando el conjunto de datos de entrenamiento.
- Analizar del efecto del los módulos de *spatial pyramid pooling* y resolución en z del decodificador.
- Analizar el efecto del *gradient matching term* y la aleatoriedad de la selección de planos.
- Comparar cuantitativamente el método propuesto con aportaciones similares de la literatura.
- Analizar y comparar cualitativamente sobre escenarios reales.

5.4.1. Métrica de evaluación

Como se ha mencionado en la sección 5.3.1, no se ha podido encontrar un conjunto de datos compuesto por *focal stacks* reales y una información de distancias precisa; por tanto, para realizar una evaluación cuantitativa se utiliza el conjunto de validación de Middlebury y DIML, puesto que estos dos conjuntos de datos, al estar compuestos por escenas reales, representan mejor un escenario real.

Puesto que no es posible comparar directamente el método propuesto con otros trabajos de la literatura, pues, cada uno tiene como resultado una información diferente (distancias absolutas, invariantes a la traslación y escala, índices de planos, etc.) se diseñan dos métricas: una absoluta y una normalizada. La absoluta es el RMS de la siguiente ecuación:

$$RMS(\hat{d}, d^*) = \sqrt{\frac{1}{|M|} \sum_{i \in M} (\hat{d}_i - d_i^*)^2} \tag{5.11}$$

Con M como el conjunto de píxeles válidos en el *ground truth*, $d^* \in \mathbb{R}^{W,H}$ el *ground truth* en metros y $\hat{d} \in \mathbb{R}^{W,H}$ la predicción de cada método correctamente escalada y trasladada a la misma escala y origen que el *ground truth*. En el caso de nuestro método, $\hat{d}$ es, sabiendo la posición de enfoque de cada plano, la traducción de índices de planos a distancias mediante interpolación lineal. Para los métodos invariantes a escala y traslación, $\hat{d}$ se define de la siguiente forma:

$$\hat{d} = ds + b \tag{5.12}$$

Con d como la salida invariante a la escala y traslación, s y b definidos como:

$$(s, b) = arg\,min_{s,b} \sum_{i \in M} (sd_i + b - d_i^*)^2 \tag{5.13}$$

Como segunda métrica, primero se normaliza la predicción y el *ground truth* en el rango $[0, 1]$ y luego se calcula la similitud entre ambas utilizando como métrica el SSIM [134]. El SSIM se calcula a nivel de ventanas. Para cada píxel de ambas imágenes a comparar, se toma una ventana (de tamaño 11 en este caso) y se calcula su similitud siguiendo la fórmula:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{5.14}$$

Con x e y como las dos ventanas a comparar, μ_x y μ_y como la media de x e y respectivamente, σ_x^2 y σ_y^2 como la varianza de x e y respectivamente, σ_{xy} como la covarianza de x e y, $c_1 = (0{,}01L)^2$, $c_2 = (0{,}02L)^2$, y L como el rango dinámico de los valores de los píxeles. El valor final de similitud SSIM se obtiene con la media de la similitud de todas las ventanas.

La métrica de la ecuación 5.11 da como resultado el error en metros de cada método comparado con el *ground truth*. Esta métrica en particular favorecerá a los métodos invariantes a escala y traslación puesto que el resultado será escalado y trasladado buscando el mejor ajuste con el *ground truth*, por tanto, el resultado que se compara se podrá decir que no tiene error ni en escala ni en traslación. Mientras que, en el caso del método propuesto, como este da toda la información necesaria para obtener distancias absolutas, no se le aplica modificación alguna de escala o traslación, por tanto, el resultado final comparado en la ecuación 5.11 podría tener errores de escala y/o traslación.

5.4.2. Análisis del método

Selección del conjunto de datos

En primer lugar, se analiza el desempeño del método entrenado con los conjuntos de datos: Middlebury, DIML y FlyingThings.

La tabla 5.2 muestra una comparación cuantitativa del método entrenado con estos tres conjuntos de datos y evaluado sobre el conjunto de validación de Middlebury 2005-2006 [124, 125], Middlebury 2014 [126] y DIML [120].

El conjunto de validación de Middlebury se divide en dos grupos: Middlebury 2005-2006 y Middlebury 2014. Se hace esta división puesto que cada subconjunto tiene resolución diferente y sus *ground truths* fueron tomados con métodos diferentes: Middlebury 2005-2006 tiene una resolución de 1.5MP y sus *ground truth* obtenidos con el método [125], mientras que Middlebury 2014 tiene una resolución de 5MP y sus *ground truth* obtenidos con el método [126].

TABLA 5.2: Comparación cuantitativa del desempeño del método entrenado con diferentes conjuntos de datos.

	Entrenado con DIML		Entrenado con Middlebury		Entrenado con FlyingThings	
	RMS	SSIM	RMS	SSIM	RMS	SSIM
Middlebury 2005-2006	0.09	70 %	**0.09**	**73 %**	0.18	53 %
Middlebury 2014	0.13	80 %	**0.12**	**82 %**	0.31	60 %
DIML	**0.04**	**92 %**	0.13	79 %	0.27	76 %

Como es de esperar, se logra mejor desempeño sobre el conjunto de validación del conjunto de datos sobre el que fue entrenado. No obstante, es importante analizar los resultados del desempeño para todos los conjuntos de datos de validación. De este análisis, se puede extraer que el utilizar únicamente FlyingThings para el entrenamiento no da buenos resultados sobre escenas reales, mientras que los modelos entrenados con DIML y Middlebury fueron capaces de lograr un resultado similar sobre los distintos conjuntos de validación. Esto podría explicarse dada la propia naturaleza de FlyingThings, pues, a diferencia de Middlebury y DIML, este está formado por escenas completamente sintéticas y poco realistas.

A pesar de haber dado una métrica de calidad cuantitativa, no es suficiente para entender el efecto que esta diferencia supone. De hecho, sería de esperar que entrenar con DIML o Middlebury diera resultados diferentes debido a la gran diferencia en cuanto a calidad de los *ground truths* de ambos conjuntos;

sin embargo, según los resultados de la tabla 5.2 se podría entender que entrenando con DIML o Middlebury se obtienen resultados similares, entrando en contradicción con lo que cabría de esperar.

Configuraciones de arquitectura

Se realizan experimentos para analizar el efecto de configuraciones diferentes de la arquitectura. Específicamente, se analizan diferentes configuraciones del decodificador 3D:

- El efecto del módulo de *spatial pyramid pooling* al final de cada *multiscale aggregation block*.
- La resolución del modelo por plano:
  - Misma resolución en z que número de planos
  - Doble de resolución en z que número de planos
  - Cuádruple de resolución en z que número de planos

TABLA 5.3: Desempeño de diferentes configuraciones de decodificador 3D.

	Middlebury 2005-2006		Middlebury 2014		DIML	
	RMS	SSIM	RMS	SSIM	RMS	SSIM
Sin pooling	0.14	65 %	0.18	76 %	0.17	77 %
Con Pooling	0.09	73 %	0.12	82 %	0.13	79 %
Doble planos + pooling	**0.08**	**78 %**	**0.10**	**85 %**	0.11	81 %
Cuádruple planos + pooling	0.08	76 %	0.11	83 %	**0.09**	**82 %**

De la tabla 5.3 se puede concluir que el módulo de *spatial pyramid pooling* sí mejora el desempeño del modelo. Es importante notar que, en este caso, el *spatial pyramid pooling* no solo se aplica a nivel espacial, sino a nivel de planos, lo que permite comparar cada plano con todos los demás, por lo que, de no utilizar este módulo, se podría dar el caso de tener planos que nunca se comparen con algún otro (por ejemplo, el primer con el último plano de un *focal stack* con un número elevado de planos).

Habiendo concluido que el uso del *spatial pyramid pooling* mejora los resultados, se experimenta variando la resolución del decodificador por cada plano.

Lo que se pretende con la variación de la resolución en profundidad es es verificar si, virtualmente incrementando el número de planos del decodificador (por ejemplo, un *focal stack* de 6 planos de entrada y un decodificador con funcionando con forma $[B, C, 12, H, W]$), se pudieran mejorar los resultados. En otras palabras, se pretende analizar el efecto de aumentar la resolución de cada plano en el decodificador. La tabla 5.3 muestra los resultados para experimentos ejecutados con el doble y el cuádruple de resolución. Estos resultados muestran que doblando el número de planos los resultados mejoran, pero se estancan más allá de este valor.

De acuerdo con estos resultados, los siguientes experimentos se realizan con el modelo utilizando *spatial pyramid pooling* y doblando el número de planos.

Análisis del entrenamiento

Se analiza el efecto de modificar el protocolo de entrenamiento explicado en la sección 5.3.3. Específicamente, se analiza el efecto del *gradient matching term* y de la utilización de planos variables en el conjunto de datos de entrenamiento.

TABLA 5.4: Análisis de diferentes tipos de entrenamiento

	Middlebury 2005-2006		Middlebury 2014		DIML	
	RMS	SSIM	RMS	SSIM	RMS	SSIM
Sin gradiente + planos variables	0.08	76 %	0.11	84 %	0.12	80 %
Con gradiente + planos fijos	0.10	70 %	0.15	79 %	0.13	78 %
Con gradiente + planos variables	**0.08**	**78 %**	**0.10**	**85 %**	**0.11**	**81 %**

En la tabla 5.4 se muestran los resultados de entrenar con y sin el *gradient matching term* 5.9 y de entrenar con un número de planos fijo o variable en el conjunto de datos de entrenamiento.

5.4.3. Comparación de métodos

En la tabla 5.5 se muestra una comparativa de nuestro método con otros algoritmos de la literatura. Específicamente, se compara con un algoritmo de *depth from focus* tradicional [115], uno basado en *deep learning* [117] y con uno de estimación de distancias monoculares a partir de una única imagen también basado en *deep learning* [112].

Para esta comparativa, el método propuesto se entrenó y configuró con las configuraciones que arrojaron mejor resultados en las secciones anteriores:

- Entrenado con el conjunto de datos de Middlebury generado con número de planos variable.
- Duplicando el la resolución de cada plano.
- Utilizando el módulo de *spatial pyramid pooling*.
- Entrenado utilizando como función de costes la ecuación 5.10.

TABLA 5.5: Comparativa con otros métodos.

	Middlebury 2005-2006		Middlebury 2014		DIML	
	RMS	SSIM	RMS	SSIM	RMS	SSIM
VDFF[115]	0.10	41 %	0.18	60 %	0.16	76 %
DDFF[117]	0.14	72 %	0.24	79 %	0.35	72 %
MiDaS[112]	0.09	73 %	0.17	83 %	0.17	80 %
Nuestro	**0.08**	**78 %**	**0.10**	**85 %**	**0.11**	**81 %**

Según los resultados de la tabla 5.5, se extrae que el método propuesto es el que mejor desempeño tiene sobre los datos de validación de todos los conjuntos de datos sobre los que se evaluó. A este, le sigue, el algoritmo de MiDaS, luego VDFF, y por último DDFF. Es de notar que el método MiDaS, que es un algoritmo de estimación de distancias monocular, supere a los métodos VDFF y DDFF, que son algoritmos específicos de *depth from focus*. Esto se explica al tener en cuenta las características específicas de los métodos VDFF y DDFF. VDFF es un algoritmo de *depth from focus* que no utiliza técnicas de *deep learning* y que se basa únicamente en la información de enfoque para dar un mapa de distancias inicial que luego rellena buscando que la transición entre los valores de la estimación inicial y los que tiene que rellenar sea suave. Por otro lado, DDFF es un algoritmo basado en *deep learning* entrenado con un conjunto de datos específico. En este sentido, frente a MiDaS,

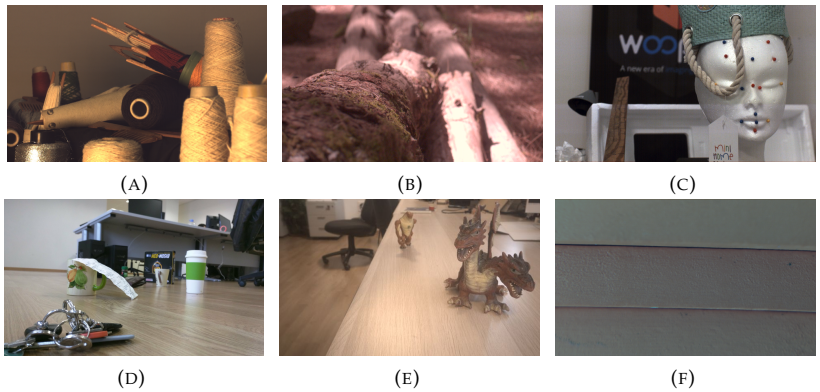
VDFF no aprovecha la información semántica mientras que DDFF se especializa en un conjunto de datos específico, imposibilitando su generalización. El método propuesto por otro lado, consigue los mejores resultados pues, es está diseñado para poder extraer información semántica y de enfoque para estimar distancias a la vez que se entrenó para ser capaz de generalizar a diferentes escenas.

Estas diferencias se aprecian mejor al realizar comparaciones cualitativas. Estas se realizan en la sección 5.5.

5.5. Experimentos cualitativos

Como se ha visto en la sección 5.4, una evaluación cuantitativa no es capaz de expresar por sí sola toda la información que se podría dar con una evaluación cualitativa. Para ello, se prepara un conjunto de *test* formado por imágenes de escenas reales tomadas con diferentes dispositivos y cámaras:

- 3 *focal stacks* de 6 planos tomados con una cámara Flare 2MP [135] usando una lente líquida de enfoque variable controlada electrónicamente [136].
- 1 *focal stack* de 6 planos con la misma Flare 2MP pero utilizando un lentes de diferente focal (Thorlabs LSC01-A-Ø2"N-BK7 [137]).
- 2 *focal stacks* de 10 planos tomados con un teléfono móvil (Motorola Nexus 6).



(A) (B) (C)

(D) (E) (F)

FIGURA 5.4: Muestras de los *focal stacks* del conjunto de test cualitativo. 5.4a, 5.4b y 5.4c son *focal stacks* de 6 planos tomados con la cámara Flare 2MP con la lente líquida con una exposición de 6ms. 5.4a es una escena de interior con luz artificial, 5.4b es una escena de exterior con luz natural y 5.4c es una escena de interior sin luz artificial. Las muestras 5.4d y 5.4e son de *focal stacks* de 10 planos tomados con el móvil con exposición automática. Finalmente, la muestra 5.4f es un *focal stack* de 6 planos utilizando la cámara Flare 2MP y la lente macro [137].

La figura 5.4 muestra uno de los planos de cada *focal stack* del conjunto de *test* cualitativo.

5.5.1. Comparación de conjuntos de datos

Siguiendo el experimento de la tabla 5.2, se comparan entrenamientos con diferentes conjuntos de datos: Middlebury, DIML y FlyingThings.

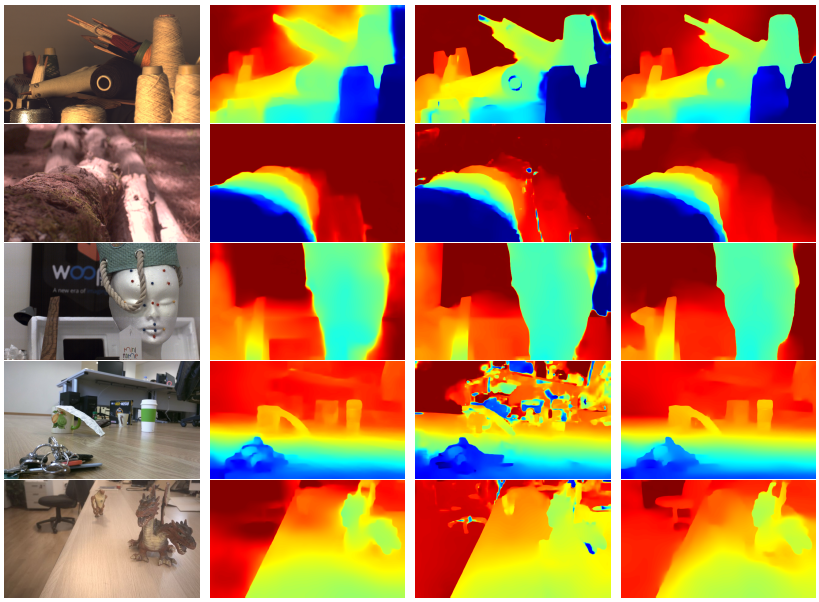


FIGURA 5.5: Primera columna: una imagen RGB del *focal stack*, segunda a cuarta columna: mapas de distancia obtenidos con el modelo entrenado con DIML, FlyingThings and Middlebury respectivamente.

Como puede verse en la figura 5.5, al entrenar con el conjunto de datos FlyingThings (tercera columna), no se logra generalizar bien; probablemente debido a que este es completamente sintético. Este resultado es coherente con los experimentos cuantitativos de la tabla 5.2. Al entrenar con el conjunto de datos DIML (segunda columna) se lograron mejores resultados que con FlyingThings, pero presentando errores al detectar los bordes, probablemente causados por los *ground truth* del conjunto de datos. Por último, el modelo entrenado con el conjunto de datos Middlebury fue el que mejores resultados mostró a pesar de contar con un número limitado de instancias.

Estos resultados cualitativos contrastan con los cuantitativos de la tabla 5.2. Coinciden en demostrar el bajo desempeño al utilizar FlyingThings pero, mientras en la tabla 5.2 no se puede apreciar diferencia de calidad de resultados entre utilizar DIML o Middlebury, los resultados cualitativos de la figura 5.5 muestran una clara diferencia entre ellos. Esta diferencia se hace

patente sobre todo en los bordes de los objetos, dónde más errores presentan los *ground truth* del conjunto de datos DIML.

Este efecto observado demuestra la importancia del análisis cualitativo para poder entender el efecto de una diferencia cuantitativa.

5.5.2. Comparación de métodos

En esta sección se comparará con otros métodos descritos en la literatura utilizando el mismo conjunto de *test* y comparando con los mismos métodos con los que se comparó en la sección 5.4.3. El objetivo de esta comparación es obtener información cualitativa de cada método para así realizar una comparativa similar a la realizada en la figura 5.5.

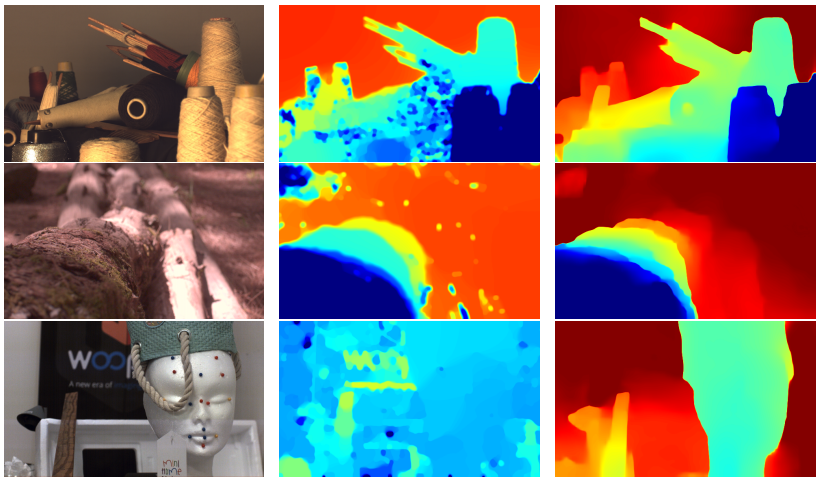


FIGURA 5.6: Columna izquierda: una imagen RGB del *focal stack*, columna central: resultados de VDFF, columna derecha: resultados del métodos propuesto.

En la figura 5.6 se compara al método propuesto con el algoritmo VDFF [115]. En este caso, los 3 *focal stacks* son de 6 planos y tomados con la misma cámara y el algoritmo de VDFF está parametrizado con los mismos parámetros que dieron lugar a los resultados de la tabla 5.5. El método propuesto consigue extraer distancias de las 3 escenas, mientras que VDFF logra dar ciertos resultados, aunque con artefactos, en los primeros dos casos pero falla completamente en la tercera escena.

Los artefactos de las primeras dos escenas pueden deberse perfectamente al ruido de la imagen. Los algoritmos clásicos de *depth from focus* se basan en analizar el gradiente de la imagen y luego extraen distancias; por lo que, son muy sensibles al ruido, dando esos resultados con claros errores repartidos en puntos dispersos. Por otro lado, en el caso en que VDFF falla completamente, cabe destacar que este caso es una escena de interior sin luz artificial, por tanto, una escena oscura y con mayor ruido que las demás, razón que podría explicar el fallo.

Si bien se podrían mejorar los resultados de VDFF con una mejor parametrización, es importante remarcar que estas 3 instancias están tomadas con la misma cámara y óptica y aún así en 1 de los 3 casos falla completamente.Lo que demuestra que, para obtener resultados coherentes con este algoritmo, no es suficiente con parametrizar para una cámara y óptica específica.

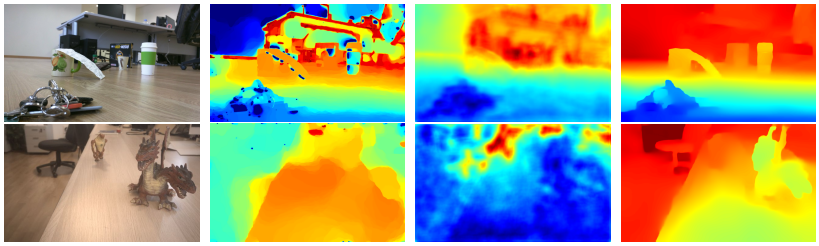


FIGURA 5.7: Primera columna: una imagen RGB del *focal stack*, segunda a cuarta columna: resultados de VDFF, DDFF y el método propuesto respectivamente.

En la figura 5.7 se compara con VDFF and "Deep depth from focus" (DDFF) [117]. En este caso los *focal stacks* utilizados fueron los dos de 10 planos tomados con móvil (Motorola Nexus 6). En este caso, el método propuesto también consigue dar resultados coherentes con la escena capturada. En cuanto a VDFF, este consigue dar resultados coherentes asimismo, aunque con bastantes artefactos al igual que en la comparativa anterior. Por otro lado, DDFF da resultados completamente diferentes en ambos casos: en el primero se puede apreciar cierta coherencia con la escena en cuestión (aunque con claros errores), mientras que en el segundo caso no es posible distinguir relación alguna entre el mapa de distancias y la escena.

Este desempeño se hace patente también en la tabla 5.5, donde se muestra la clara diferencia en las métricas de DDFF respecto al resto. Estos resultados

sobre escenas reales se deben a la propia naturaleza de DDFF, ya que fue entrenado sobre un conjunto específico de datos fijando parámetros clave como el número de planos del *focal stack*, impidiendo así generalizar bien a escenarios reales.

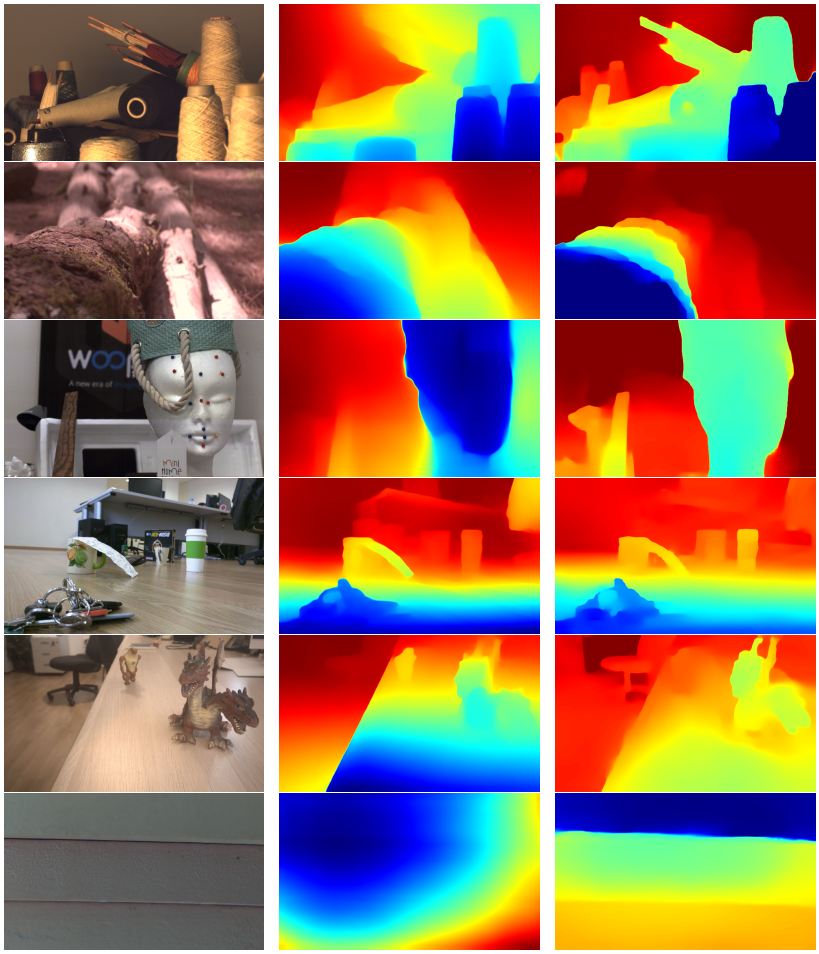


FIGURA 5.8: Columna izquierda: una imagen RGB del *focal stack*, Columna central: resultados de MiDaS, columna derecha: resultados del método propuesto.

Finalmente, en la figura 5.8 se compara con el algoritmo de estimación monocular de distancias MiDaS [112]. Visualmente, ambos métodos consiguen obtener resultados de alta calidad. Sin embargo, al contar con la información

adicional del desenfoque, el método propuesto (tercera columna) consigue mejores resultados, coincidiendo así con los resultados cuantitativos de la tabla 5.5. Por ejemplo, en las filas 1 y 3 MiDaS confunde el fondo con el frente, mientras que en el caso de la columna 6 falla claramente al ser imposible obtener información alguna de distancias utilizando únicamente una imagen. Esta última muestra es un caso extremo: una escena compuesta por 3 tablas de madera colocadas a diferentes distancias de la cámara capturadas utilizando una lente macro. En este último caso se demuestra la utilidad de la información de enfoque al lograr un resultado coherente con la escena con el método propuesto, mientras que MiDaS falla como es de esperar.

Otro punto importante es la escala en la que se obtiene el resultado; MiDas da una salida invariante a la escala y traslación, mientras que el método propuesto da como salida un mapa de índices de planos que, en caso de contar con la información de enfoque de cada plano, puede traducirse a distancias absolutas.

Respecto al tiempo de ejecución, el método propuesto es considerablemente más rápido: 25 ms. para un *focal stack* de 6 planos en una Nvidia 2080 Rtx mientras que MiDaS se ejecuta en 260 ms. para una imagen de la misma resolución sobre la misma tarjeta gráfica.

5.5.3. Análisis *focal stacks* dinámicos

Uno de los principales inconvenientes de los métodos de *depth from focus* es la imposibilidad de tratar con *focal stacks* que presenten movimiento entre planos. El movimiento puede darse principalmente por dos razones:

- Movimiento de cámara.
- Movimiento de la escena.

En ambos casos, los resultados que pueda obtener cualquier método que no contemple corrección de movimiento serán negativos.

En el caso del primer tipo de movimiento, movimiento de cámara, este podrá ser corregido con algoritmos de registro rígido, pues, al mantener la escena estática, se podrá corregir mediante transformaciones de perspectiva entre un plano y otro para así alinear cada elemento.

En el caso del movimiento de la escena, este podrá compensarse con algoritmos de registro no rígido (por ejemplo, *optical flow*), sin embargo, será una compensación más que una corrección efectiva. Por otro lado, al tener que

registrar imágenes con diferente enfoque, un registro no rígido alteraría el nivel de desenfoque de cada plano, afectando así a la información esencial utilizada en los métodos de *depth from focus*.

FIGURA 5.9: Ejemplo de *focal stack* con movimiento entre planos. Se marca un rectángulo rojo en cada plano en la misma posición para facilitar la apreciación del movimiento.

En la figura 5.9 se muestra un ejemplo de un *focal stack* con movimiento. Este ejemplo presenta un movimiento únicamente de la escena y no de cámara.

El método propuesto no contempla ningún tipo de corrección de movimiento ni tampoco aplica un registro previo a las imágenes, y, aunque lo contemplara, existen situaciones que son imposibles de corregir como la que se muestra en la figura 5.10. En dicha figura se muestra un ejemplo de un *focal stack* capturado con una cámara relativamente lenta, a 25 imágenes por segundo (FPS).

FIGURA 5.10: Ejemplo de *focal stack* con movimiento agresivo entre planos. *Focal stack* capturado a 25 FPS.

Si bien es de esperar que el método falle, se ejecuta sobre un conjunto de imágenes tomado con una cámara estática.

FIGURA 5.11: Resultados de aplicar el método sobre *focal stacks* con movimiento. Primera columna: primer plano del *focal stack*, segunda: último plano del *focal stacks*, tercera: mapa de distancias estimado.

En la figura 5.11 se muestran algunos resultados de aplicar el método sobre *focal stacks* con movimiento. Claramente, el algoritmo falla al no haber tenido en cuenta este tipo de situaciones en ningún momento.

5.6. *Depth from focus* dinámico

Como se ha visto en la sección 5.5.3, el método falla al encontrarse en situaciones que presentan movimiento dentro de un mismo *focal stack*. Este tipo de

casos resulta particularmente habitual en condiciones en las que las imágenes se toman a mano alzada o sobre escenas reales en lugar de sobre situaciones controladas.

Para dar solución al problema, primero ha de definirse qué tipo de salida se espera de un *focal stack* con movimiento. Al representar cada plano momentos diferentes de la escena, se decide por dar como solución esperada de un *focal stack* con movimiento a un mapa de distancias que se corresponda con un plano específico (de referencia) del *focal stack*. Así, en el caso de ejemplo de la figura 5.10, si se elige el último plano como referencia, el mapa de distancias esperado deberá contemplar la mano, mientras que, si se elige el primero como referencia, el mapa de distancias no debería de contemplarla puesto que en el primer plano esta no es visible.

Habiendo establecido el objetivo a perseguir, el método se divide en tres partes:

- Búsqueda y procesamiento de datos.
- Arquitectura del modelo.
- Implementación del protocolo de entrenamiento.

5.6.1. Conjunto de datos

Como se ha visto en la sección 5.3.1, encontrar datos de *focal stacks* con su respectiva información de distancia que sean de calidad resulta todo un reto. Más aún si se requiere que estos contengan movimiento dentro del propio *focal stack*.

Al no encontrar este tipo de datos, se prosigue, de forma similar a como se hizo para el caso en que no hay movimiento, a generar sintéticamente los datos necesarios. Para ello, será necesario partir de datos que sean imágenes junto con sus respectivos mapas de distancias y que a su vez estén tomados en forma de secuencia, para así poder generar secuencias variando el enfoque y de esta manera simular una captura de un *focal stack* con movimiento.

Como se aprecia en la tabla 5.1, apenas 3 de los 11 conjuntos de datos explorados contienen secuencias de imágenes. Dos de ellos son conjuntos de datos sintéticos y el tercero es DIML. Según lo analizado en las secciones 5.4 y 5.5, los datos sintéticos no logran buen desempeño y el conjunto de datos DIML contiene muchos errores que sesgan el resultado final.

Debido a estas dificultades, se decide seguir por una vía diferente: generar los mapas de distancias a partir de información estéreo. El trabajo de Ranftl et al [112] demuestra que es posible generar datos de distancias a partir de escenas de películas 3D. Para ello, se seleccionan películas que hayan sido grabadas en estéreo y, a partir del par de imágenes se extraen distancias.

Selección de datos

En primer lugar, es necesario elegir las películas a utilizar. Se seleccionan las que se proponen en el trabajo de Ranftl et al [112]. Una vez seleccionadas, estas son cortadas por escenas. Para ello, se utiliza la herramienta *ffmpeg* [138] que contiene una utilidad para la extracción automática en escenas. Adicionalmente, se descartan todas las escenas que duren menos de 1 segundo. Al contrario que en [112], como se pretende utilizar secuencias y no imágenes individuales, no se hace el submuestreo a 4 FPS, y, para compensar, en lugar de muestrear 24 imágenes cada cuatro segundos, se extraen 22 cada 12.

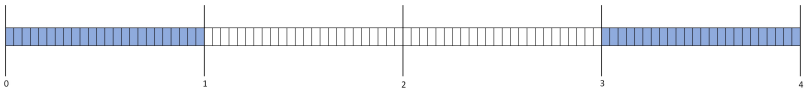


FIGURA 5.12: Ejemplo de selección de 24 imágenes cada 3 segundos de una escena extraída a 24 FPS.

A modo de ilustración, en la figura 5.12 se muestra un ejemplo de selección de 24 imágenes cada 3 segundos de una escena extraída a 24 FPS.

El resultado de esta selección resulta en un conjunto de secuencias de 22 imágenes, cada uno de los imágenes dentro de una misma secuencia consecutivos entre sí, manteniendo así la coherencia temporal.

Extracción de distancias

Tal y como [112] denota, extraer distancias de un par estéreo resulta especialmente difícil al no contar con la información de calibración. Por tanto, los métodos de estéreo dan resultados pobres a la hora de estimar distancias.

No obstante, sí es posible extraer esta información a partir de un *optical flow*. Para ello, se utiliza el algoritmo de *optical flow* RAFT [139] para extraer el movimiento de cada pixel de una imagen a la otra y manteniendo únicamente la información de desplazamiento horizontal. Este algoritmo consiste en una

red neuronal que realiza una estimación inicial de desplazamiento que luego refina iterativamente. En la figura 5.13 se muestra el diagrama del algoritmo.

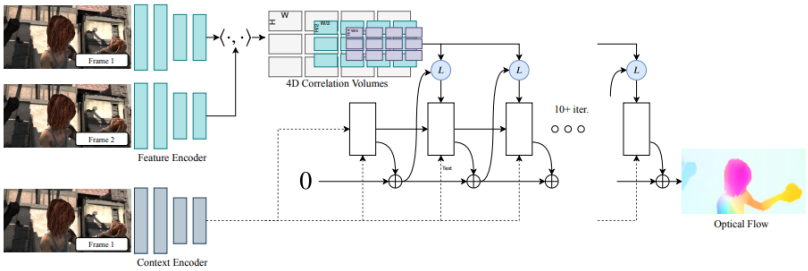


FIGURA 5.13: Diagrama del algoritmo RAFT. Imagen extraída de [139].

Al tratarse de un algoritmo de *optical flow* y no de estéreo, para obtener el equivalente a las oclusiones se calcula el *optical flow* en ambas direcciones y se remapean de forma directa e inversa para verificar su consistencia. Cualquier punto con un error de remapeo mayor a un pixel se considerará inválido, generando así una máscara similar a la máscara de oclusiones de un algoritmo estéreo.

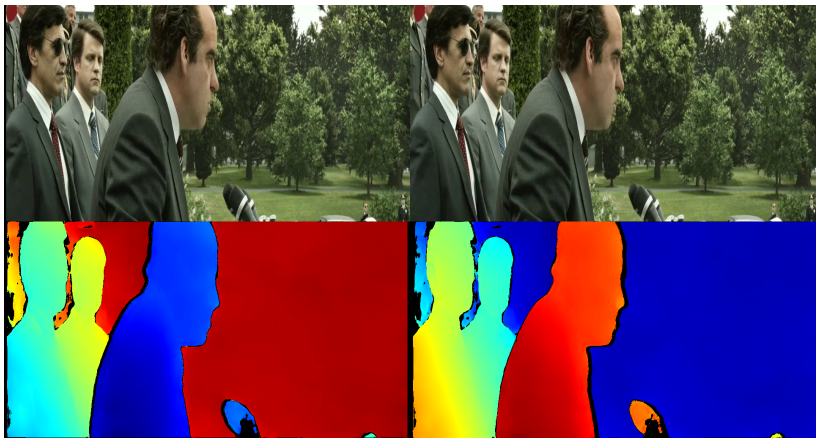


FIGURA 5.14: Ejemplo de un par estéreo de una imagen y el desplazamiento en X de los mapas de *optial flows* obtenidos en ambas direcciones (desplazamiento de la imagen de la izquierda a la derecha y viceversa).

En la figura 5.14 se muestra un ejemplo de par de imágenes de una escena con sus mapas de desplazamiento en X tras aplicar la máscara de oclusiones.

A modo de filtro de detección de imágenes inválidas, se utiliza la máscara de oclusiones. Si esta contiene más de un 30 % de píxeles inválidos, se considerará el par de imágenes inválido así como toda la secuencia a la que pertenezca. Del mismo modo, se considerará inválido a cualquier par de imágenes que, tras aplicar la máscara de oclusiones, más del 10 % de los píxeles tengan una disparidad vertical superior a 2 píxeles.

Proceso completo

Habiendo definido como se seleccionan los datos y extraen distancias, el proceso completo de extracción datos se presenta en la figura 5.15.

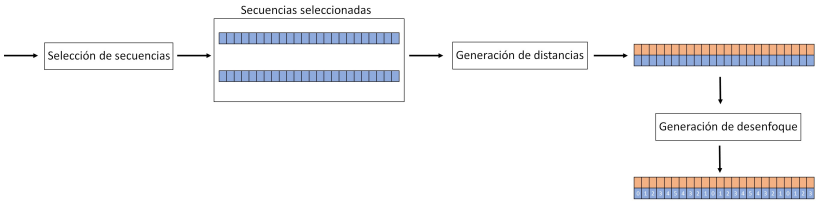


FIGURA 5.15: Ilustración del proceso completo para extraer de un vídeo secuencias desenfocadas en distintos puntos.

En ella se muestra como se utiliza la selección explicada en la sección 5.6.1, la extracción de distancias explicada en 5.6.1 y la generación de imágenes desenfocadas de la sección 5.3.1. Cabe destacar que, al tener secuencias más largas que tamaño de *focal stack*, se repetirán posiciones de enfoque, por lo que, se define además un ciclo de enfoques donde se indica el orden en que se recorre cada posición de enfoque. En el caso de la figura 5.15 se muestra un ejemplo en el que se recorre de forma simétrica simulando una curva triangular.

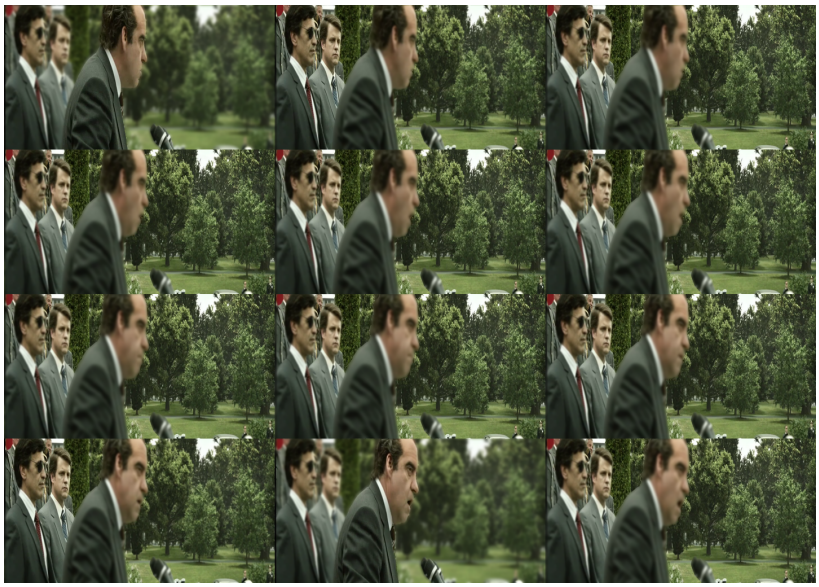


FIGURA 5.16: Ejemplo de una secuencia de imágenes desenfocadas sintéticamente. Ordenadas de izquierda a derecha y de arriba abajo.

En la figura 5.16 se muestra un ejemplo de una secuencia de imágenes desenfocadas sintéticamente siguiendo el proceso de la figura 5.15.

5.6.2. Arquitectura

En la literatura, la solución propuesta para solventar el problema del movimiento es registrar las imágenes antes de aplicar el método de *depth from focus*. Siguiendo en esta linea, se propone realizar un alineamiento de cada plano al plano de referencia escogido, con la diferencia que, en lugar de registrar las imágenes de entrada, se realiza un registro diferenciable a nivel de características de imagen. Al ser diferenciable, se incluye en la propia red neuronal, permitiendo así un entrenamiento punto a punto de todo el modelo.

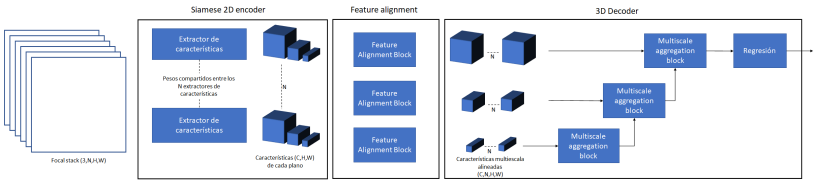


FIGURA 5.17: Arquitectura de red para *depth from focus* contemplando movimiento entre planos de un mismo *focal stack*.

En la figura 5.17 se muestra la nueva arquitectura del modelo. En ella, se agrega el módulo *Feature alignment*, donde se registran las características de cada imagen a la imagen escogida como referencia para cada una de las escalas que el codificador 2D tiene como salida.

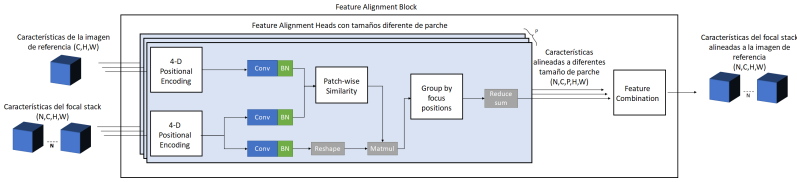


FIGURA 5.18: Diagrama del *Feature Alignment Block*.

En la figura 5.18 se muestra el diagrama del bloque encargado de alinear las características. Este se basa en el mecanismo de la estructura de *transformer* [140], comúnmente aplicado a problemas de traducción de texto. Un bloque *transformer* es un mecanismo que pondera la influencia de distintas partes de los datos de entrada.

Este bloque puede entenderse desde el punto de vista del funcionamiento de los sistemas de búsqueda y recopilación de información siguiendo el modelo "*Query, (Key, Value)*". En este sentido, se tiene una serie de valores, los "*Value*", que están representados por sus correspondientes claves, los "*Key*". En el momento en que se realiza una petición dando un "*Query*", se busca la "*key*" que más se parezca a la petición realizada (se busca la *key* que se parezca más al *query* en cuestión) y se da como resultado el *value* asociado a dicha *key*. Los sistemas de búsqueda y recopilación de información modernos son más sofisticados, pero el modelo de *transformer* sigue esta filosofía. En este sentido, un *transformer* puede entenderse como una función que, dada la tripleta *query, (key, value)*, produce un resultado que combinación de los distintos *values*, ponderados según la similitud de sus correspondientes *keys*

con la *query*. En este contexto, las entradas *query*, *key* y *value* son tensores. Las dimensiones de cada tensor depende de la implementación específica.

Como se mencionó anteriormente, el modelo de *transformer* se aplica comúnmente a problemas de traducción de texto. No obstante, otros trabajos lo han aplicado a imágenes [141, 142]. Sin embargo, el método propuesto se diferencia en varios aspectos:

- No se aplica a nivel de píxel, sino a nivel de parches de tamaño variable.
- No se utiliza como método para la extracción de la medida de importancia de un tensor consigo mismo, sino que se utiliza como método para la obtención de importancia entre imágenes diferentes para así registrar respecto a una referencia.
- Se aplica a un contexto de cuatro dimensiones: espacial (ancho y alto), de enfoque y temporal en lugar de uno bidimensional.

Feature Alignment Block

El *Feature Alignment Block* toma como entrada, por un lado el tensor de características de la imagen de referencia escogido (la última imagen vista en términos cronológicos por ejemplo) $RF \in \mathbb{R}^{C,H,W}$, y por otro, el tensor $SF \in \mathbb{R}^{N,C,H,W}$ que representa el conjunto de tensores de características de las imágenes del *focal stack*, incluyendo el de referencia. En este contexto, N representa el número de *frames* que tiene la secuencia a analizar, C, H y W el número de características, ancho y alto del tensor de características extraído por el codificador 2D a una escala específica.

Feature Alignment Head

El *Feature Alignment Block* está compuesto por P *Feature Alignment Head*, que son los módulos que alinean los tensores al de referencia siguiendo un tamaño de parche diferente.

Cada *Feature Alignment Head*, primero incrusta la información temporal, espacial y de enfoque a cada tensor de entrada. Esta información se codifica siguiendo la ecuación 5.16.

$$E_{2i,y,x} = \sin\left(e^{2i(-\log\alpha/C)}\sqrt{x^2+y^2+t^2+d^2}\right) \tag{5.15}$$

$$E_{2i+1,y,x} = \cos\left(e^{2i(-\log\alpha/C)}\sqrt{x^2+y^2+t^2+d^2}\right) \tag{5.16}$$

Con $E \in \mathbb{R}^{C,H/h_p,W/w_p}$ como la codificación a incrustar para unas características con unas dimensiones de $[C,H,W]$, w_p y h_p como el ancho y alto del parche respectivamente, $t \in [0,N-1]$ como la posición temporal del tensor de características, siendo 0 para el correspondiente con la última imagen en orden cronológico, $d \in [0,N-1]$ como la posición de enfoque del plano, y, $i \in [0,C/2]$.

Una vez generada la codificación, incrusta en los tensores de entrada siguiendo la ecuación 5.18.

$$ERF_{c,y,x} = RF_{c,y,x} + E_{c,y/h_p,x/w_p} \tag{5.17}$$

$$ESF_{n,c,y,x} = SF_{n,c,y,x} + E^n_{c,y/h_p,x/w_p} \tag{5.18}$$

Con $E^n \in \mathbb{R}^{C,H/h_p,W/w_p}$ como la codificación generada con la información temporal y de enfoque de la imagen n del *focal stack*.

Luego, se aplica una capa convolucional al tensor de características de la imagen de referencia, para así obtener $ERF^q \in \mathbb{R}^{C,H,W}$ como el tensor que actuará de *query* en el modelo de *transformer*. Paralelamente, se aplican dos capas convolucionales al tensor ESF, obteniendo así $ESF^k \in \mathbb{R}^{N,C,H,W}$ y $ESF^v \in \mathbb{R}^{N,C,H,W}$ como los tensores de *key* y *value* del modelo de *transformer* respectivamente.

Los tensores ERF^q y ESF^k sirven de entrada al módulo *patch-wise similarity*. Este primero re-dimensiona ERF^q en $ERF^{q'} \in \mathbb{R}^{\frac{HW}{w_p h_p}, w_p h_p C}$ y ESF^k en $ESF^{k'} \in \mathbb{R}^{N,\frac{HW}{w_p h_p}, w_p h_p C}$. Luego, calcula la similitud entre el *query* $ERF^{q'}$ y la *key* $ESF^{k'}$ siguiendo la ecuación 5.19.

$$Sim_{n,i,i'} = -\sqrt{\sum_j \left(ERF^{q'}_{i,j} - ESF^{k'}_{n,i',j}\right)^2} \tag{5.19}$$

Con $Sim \in \mathbb{R}^{N,\frac{HW}{w_p h_p},\frac{HW}{w_p h_p}}$ como la medida de similitud a nivel de parche entre las características de la imagen de referencia y cada uno de los imágenes del *focal stack*. Específicamente, $Sim_{n,i,i'}$ representa la medida de similitud entre el parche i de las características de la imagen de referencia con el parche i' de la imagen n. En el caso de que n sea el índice correspondiente a la imagen de referencia y i sea igual a i', entonces $Sim_{n,i,i'}$ debería de ser cercano a 0.

Esta medida de similitud se normaliza siguiendo la función *softmax*, siguiendo la ecuación 5.20, para obtener una medida de probabilidad.

$$Sim'_{n,i,j} = \frac{e^{Sim_{n,i,j}}}{\sum_{j'} Sim_{n,i,j'}} \tag{5.20}$$

Con $Sim' \in \mathbb{R}^{N,\frac{HW}{w_p h_p},\frac{HW}{w_p h_p}}$ como la medida de similitud normalizada en el rango $[0,1]$. Con la medida de similitud, se redimensiona el *value* ESF^v en $ESF^{v'} \in \mathbb{R}^{N,\frac{HW}{w_p h_p},w_p h_p C}$ para así poder operar con el tensor de similitudes siguiendo al ecuación 5.21.

$$ASF'_{n,i,i'} = \sum_j Sim'_{n,i,j} ESF^{v'}_{n,j,i'} \tag{5.21}$$

Con $ASF' \in \mathbb{R}^{N,\frac{HW}{w_p h_p},w_p h_p C}$ como el tensor de características de las imágenes del *focal stack* alineados a la imagen de referencia. Este tensor luego se redimensiona a $ASF \in \mathbb{R}^{N,C,H,W}$.

En caso de que en la secuencia de entrada se tengan imágenes con una posición de enfoque repetida, se procede a agruparlos y reducirlos mediante suma para así dar como salida un tensor que representa las características de las imágenes del *focal stack* alineados con la imagen de referencia.

Feature Combination

La salida de las P *Feature Alignment Heads* da como resultado P tensores $ASF \in \mathbb{R}^{N,C,H,W}$, o, lo que es lo mismo, un tensor $PASF \in \mathbb{R}^{N,C,P,H,W}$. Este se re-dimensiona a $[N*C,P,H,W]$ para así, mediante una convolución 2D poder reducir los P tamaños de parches a 1, dando como resultado un tensor $[N*C,1,H,W]$ que, a su vez, se re-dimensiona a $[N,C,H,W]$. Finalmente, se trasponen las primeras dos dimensiones para así tener el tensor en el formato que el decodificador 3D espera.

5.6.3. Implementación

La implementación realizada para generar los datos de entrenamiento del nuevo modelo es similar a la explicada en la sección 5.3.3 pero con algunas diferencias fundamentales:

- Se añade el concepto de "ciclo de enfoques", es decir el orden en que se capturan las posiciones de enfoque.
- Cada instancia individual consiste en una secuencia y no en un *focal stack*.
- Al trabajar a nivel de secuencia y no a nivel de *focal stack* individual, el tamaño del tensor de entrada se mantiene constante, variando el ciclo de enfoque para lograr una entrada de tamaño de *focal stack* variable.
- Por cada instancia de entrenamiento se obtienen tantos mapas de distancia como elementos tenga la secuencia menos $(N-1)$, siendo N el tamaño del *focal stack* de dicha secuencia. Esto se debe a que para extraer un mapa de distancias hace falta tener procesado al menos una imagen de cada posición de enfoque, las que se procesen pasado ese número mínimo generarán un mapa de distancia al contar con la información de todas las anteriores.

Por restricciones de memoria, se fija el tamaño de secuencia a 11. Por lo que, por cada secuencia de 22 imágenes extraídas según el proceso explicado en 5.6.1, se extraen dos instancias de entrenamiento de tamaño 11. Para cada instancia de entrenamiento, se define una cámara de forma aleatoria siguiendo el mismo método que en 5.3.3 y un tamaño de *focal stack* aleatorio variando en el rango $[4, 11]$. Una vez fijada la cámara y tamaño de *focal stack*, se define las posiciones de enfoque siguiendo el mismo método que en 5.3.3.

Por lo que respecta al ciclo de enfoques, este se define siguiendo una onda triangular para así simular el comportamiento que pueda seguir una lente que barre varias posiciones de enfoque repetidamente.

Dada una secuencia de entrada representada por el tensor $t \in \mathbb{R}^{S,3,H_i,W_i}$, con S como el tamaño de la secuencia (11 en este caso), y, H_i y W_i como la resolución espacial de la imagen de entrada, se calculan las características de cada imagen de la secuencia de entrenamiento ejecutando únicamente el codificador 2D, y obteniendo $f \in \mathbb{R}^{S,C,H_f,W_f}$, con C como el número de canales, y, H_f y W_f como la resolución espacial de los mapas de características. Una vez obtenidas las características de cada imagen, para cada imagen i de la

secuencia, con $i \in [N, S-1]$, siendo N el tamaño de *focal stack*, ejecuta la siguiente parte del modelo, el módulo *Feature alignment* utilizando el tensor $f^i \in \mathbb{R}^{i,C,H_f,W_f}$, con $f^i_{s,c,h,w} = f_{s,c,h,w} \forall s \in [0,i]$ y utilizando como referencia las características de la imagen i, obteniendo así $af^i \in \mathbb{R}^{C,N,H_f,W_f}$ como el tensor de características del *focal stack* alineadas con la imagen i. Luego, se ejecuta el decodificador 3D con af^i como entrada obteniendo un mapa de índices de plano como salida.

Por tanto, para el tensor t de entrada se obtienen $S-N+1$ mapas de índices de plano como salida, representados por el tensor $d \in \mathbb{R}^{S-N+1,H,W}$. La función de costes 5.10 se modifica para contemplar cada uno de los mapas de índices de plano generados por la secuencia de entrenamiento:

$$\mathcal{L}'_g = \frac{1}{S-N+1} \sum_{i}^{S-N+1} \mathcal{L}_g(d^i, I^i) \tag{5.22}$$

Con $d^i \in \mathbb{R}^{H,W}$ y $I^i \in \mathbb{R}^{H,W}$ como el mapa de índices de planos estimado y el mapa esperado para la imagen i, con $i \in [N, S-1]$.

Asimismo, por lo que respecta al módulo *Feature alignment*, se fija el número de tamaños de parches (P) a 5, variando desde $\frac{1}{2}$ el tamaño de resolución de entrada a $\frac{1}{32}$.

Con esta función de costes y modelo, este se entrena utilizando el optimizador ADAM [68] con $\alpha = 0{,}001$, $\beta 1 = 0{,}9$ y $\beta 2 = 0{,}999$.

5.7. Resultados finales

Una vez entrenado el modelo, se evalúa nuevamente con el conjunto de datos sobre el el que el algoritmo anterior falla.

FIGURA 5.19: Comparativa de resultados del nuevo método con los *focal stacks* de la figura 5.11. Primera columna: primer plano del *focal stack*, segunda: último plano del *focal stack*, tercera: mapa de distancias estimado con el primer método, cuarta: mapa de distancias estimado con el método nuevo.

En la figura 5.19 se muestran los resultados del nuevo algoritmo frente al anterior sobre las imágenes de la figura 5.11. Los mapas de distancia del nuevo algoritmo se hicieron tomando como referencia la última imagen del *focal stack* (segunda columna). Se aprecia claramente que ya no se obtienen artefactos originados debido al movimiento, como se aprecia en los resultados del algoritmo anterior. Por otro lado, se demuestra que este nuevo método no solo es capaz de lidiar con movimientos leves como los que aparecen en los *focal stacks* de las primeras tres filas, sino que también es capaz de tratar con movimientos tan agresivos como para tener *focal stacks* con planos con información totalmente diferente entre sí como es el caso del último *focal stack*.

Adicionalmente, se prueba el método sobre un nuevo conjunto de datos generado utilizando la misma cámara Flare [135] pero aumentando la velocidad de captura a 180 fotogramas por segundo (FPS).

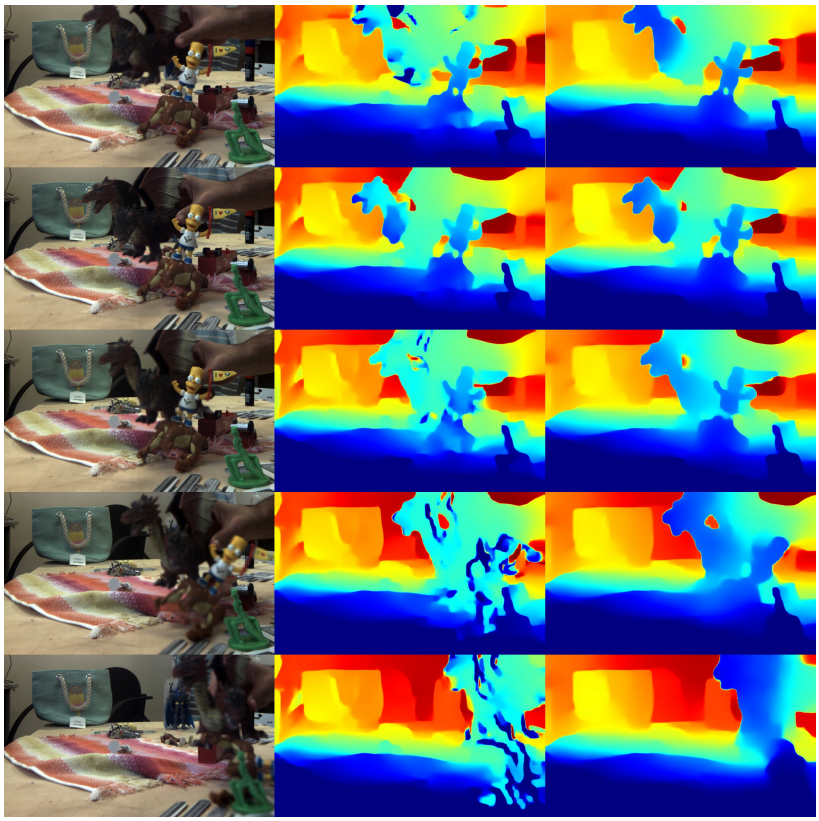


FIGURA 5.20: Comparativa entre el método anterior y el nuevo. Primera columna: imagen toda enfocada, segunda: mapa de distancias estimado con el primer método, tercera: mapa de distancias estimado con el método nuevo

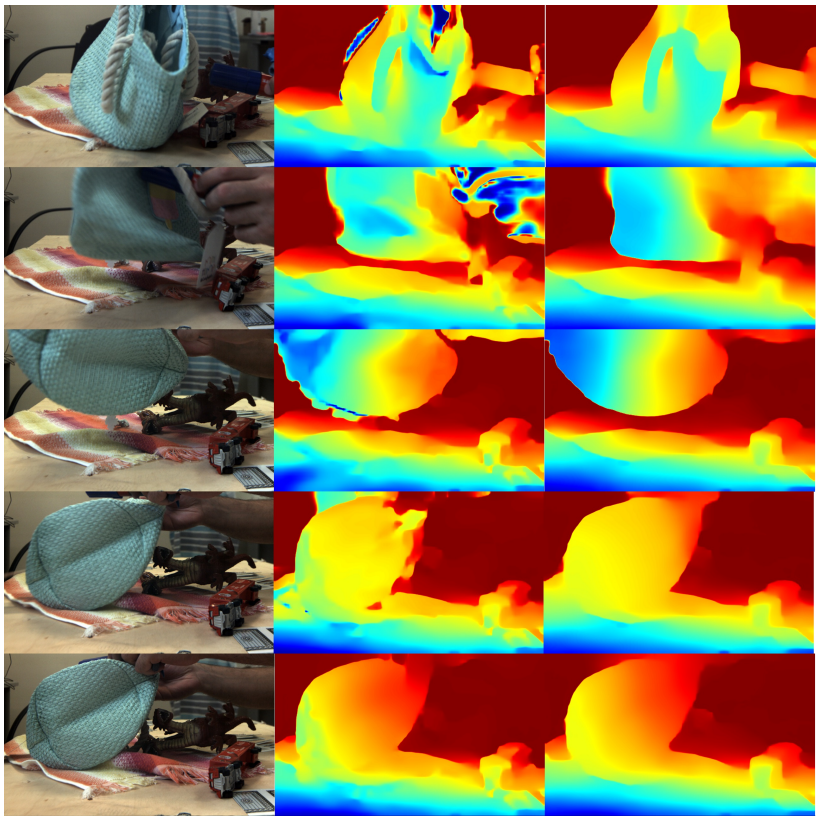


FIGURA 5.21: Comparativa adicional entre el método anterior y el nuevo. Primera columna: imagen toda enfocada, segunda: mapa de distancias estimado con el primer método, tercera: mapa de distancias estimado con el método nuevo

Las figuras 5.20 y 5.21 muestran dos secuencias de ejemplo del conjunto de datos tomado a 180 FPS. En este caso, se puede apreciar que en ambas secuencias el nivel de artefactos del algoritmo anterior es menor en comparación con lo visto en la primera comparativa. Esto puede deberse en gran parte a la cantidad de fotogramas por segundo con que se capturaron, 156 en el primer caso y 180 en el segundo. A pesar de partir de un primer algoritmo con menos fallos, el método nuevo sigue superando el desafío de mejorar los resultados del primer método.

Para mejor visualización, en lugar de mostrar una imagen de la secuencia, se muestra la imagen toda enfocada obtenida a partir de la secuencia de entrada

y el mapa de distancias. Esta imagen se logra utilizando el mapa de distancias para elegir de qué plano del *focal stack* de entrada tomar cada píxel para generar esta imagen toda enfocada.

5.8. Conclusiones

El problema de extracción de distancias de una escena es un problema ampliamente estudiado en la literatura dado el valor que tiene este tipo de información en multitud de contextos. Uno de los métodos de extracción de distancias es conocido como *depth from focus*, y que consiste en capturar múltiples imágenes de una misma escena variando la posición de enfoque para luego, mediante el análisis del desenfoque, estimar distancias.

En la bibliografía existen múltiples métodos de *depth from focus* que buscan sortear problemas inherentes a este método, como puede ser el ruido de la imagen, estimación de distancias en zonas sin textura o compensación del movimiento entre imágenes dentro de un mismo *focal stack*.

En el presente trabajo se propone un método novedoso para la resolución del problema de *depth from focus*. Este se basa en los recientes avances en redes neuronales, diseñando una arquitectura capaz de extraer las características necesarias de la imagen para luego obtener la información de distancias, sin necesidad de depender de una cámara, óptica o configuración de posiciones de enfoque especificas.

Este método propuesto se compara con algoritmos del estado del arte de *depth from focus*, tanto basados en métodos clásicos como basados en redes neuronales modernas, así como con un método de extracción de distancias monoculares. En todos los casos presenta importantes mejoras, no solo en términos cualitativos, sino también en términos de coste computacional, pudiendo ejecutar una estimación de distancias sobre un *focal stack* de 6 planos en menos de 30 ms. sobre una GPU Nvidia Quadro RTX 8000. Además de la comparativa cuantitativa, se realiza una comparativa cualitativa sobre escenas reales tomadas con una variedad de dispositivos, condiciones de luz y ópticas diferentes, demostrando una vez más la mejora en términos de calidad respecto al estado del arte.

Adicionalmente, se analiza el problema de contar con movimiento dentro de un mismo *focal stack*, verificando que, como es de esperar, el método diseñado falla en presencia de esta situación. Para este problema, se propone como

solución un registro diferenciable de características que se incorpora dentro de la propia arquitectura de red neuronal. Con este módulo adicional se demuestra que se resuelve el problema de contar con movimiento, verificando sobre una variedad de *focal stacks* de validación capturados intencionalmente con diferentes tipos de movimiento.

Con estos resultados se concluye que se aporta un nuevo método para la resolución del problema de *depth from focus* que, no solo es capaz de proveer mejor calidad de resultados y resiliencia al movimiento, sino que también es capaz de ejecutar de forma eficiente, abriendo la posibilidad a ser utilizado en dispositivos de captura de vídeo en lugar de imagen estática.

Como resultado de la investigación realizada en esta línea, se logró una patente [143] y un trabajo que se encuentra, a fecha de depósito de la presente tesis, en revisión.

Capítulo 6

Discusiones

En la presente tesis se analizaron distintos ámbitos en los que fuera posible aplicar técnicas de *machine learning* y *computer vision* para resolver o dar mejores soluciones a problemas reales. Concretamente se identificaron tres casos: detección de signos de retinopatía diabética, reconstrucción de fases de frente de onda y estimación de distancias.

A simple vista, estos tres problemas parecen diferentes, tanto en contexto como en métodos de resolución. No obstante, existe un hilo conductor que los une.

Todos los problemas surgen de un mismo tipo de dato, la imagen. En el caso del cribado de la retinopatía diabética se parte de retinografías, es decir imágenes de fondo de ojo. En el caso de la reconstrucción de fase de frente de onda se inicia del conjunto de derivadas parciales de la fase que se pretende reconstruir. Estas derivadas parciales se representan a su vez como imágenes de igual resolución que la fase a reconstruir. Por último, caso del problema de estimación de distancias, se parte de un conjunto de imágenes enfocadas a distintas distancias.

Los problemas a su vez, se afrontan actualmente utilizando los datos de partida y conocimiento específico del contexto.

En el caso de retinopatía diabética, éste se resuelve mediante la participación de un médico especializado en el cribado de retinopatía diabética (RD) que, mediante el análisis de una retinografía, establece la existencia o no de signos de RD. En el caso de la reconstrucción de fase de frente de onda se utiliza el conocimiento previo de la geometría subyacente a la fase a reconstruir para luego, con esta hipótesis de geometría, reconstruir la fase. Con respecto a la estimación de distancias, se utiliza el conocimiento previo de la óptica

y del sistema para así analizar la información de enfoque y poder estimar distancias.

Este conocimiento específico mencionado, para los tres problemas que se afrontan en la presente tesis, puede ser aprendido a partir de los datos. En el caso del cribado de la retinopatía diabética, puede extraerse el conocimiento del médico experto al imitar su comportamiento utilizando como datos imágenes de fondo de ojo etiquetadas apropiadamente bajo los estándares establecidos. En el caso de la reconstrucción de fase, el conocimiento necesario de la geometría subyacente puede aprenderse a partir de un conjunto de fases sintéticas generadas siguiendo la distribución específica del contexto deseado. En el caso de la estimación de distancias, el conocimiento necesario para poder analizar correctamente la información de enfoque puede obtenerse a partir de un conjunto de datos compuesto de imágenes de distintas escenas y cámaras y sus correspondientes mapas de distancias.

Dadas las características de estos problemas, las técnicas de *machine learning* y visión artificial resultan especialmente apropiadas para su resolución. *Machine learning* para lograr el aprendizaje del conocimiento necesario y visión artificial para su aplicación al tipo de dato de interés, la imagen.

6.1. Retinopatía diabética

Los estadíos avanzados de la retinopatía diabética y el edema macular diabético constituyen dos de las principales causas de disminución irreversible de la visión en los países desarrollados [43, 44] y son además una de las principales causas de ceguera [45]. De acuerdo con la Asociación Americana de Diabetes y la Academia Americana de Oftalmología, los pacientes con diabetes mellitus necesitan un examen de fondo de ojo, al menos una vez al año, para identificar posibles lesiones [46, 47]. Esta recomendación se basa en el hecho de que, según la Organización Mundial de la Salud, el tratamiento precoz de la retinopatía diabética puede reducir el riesgo de pérdida visual severa en más de un 90 % [48].

Teniendo en cuenta la creciente cantidad de pacientes diabéticos y el escaso número de oftalmólogos que hay en proporción, los sistemas modernos de salud, se enfrentan a un reto asistencial.

Para dar respuesta a esta situación en 2006, en Canarias, surge el programa

de teleoftalmología Retisalud. Este programa consiste en un protocolo de cribado que involucra al médico de familia (MF). Éste se encarga de analizar retinografías de sus pacientes de forma anual o bienal y, las que considere patológicas, las remite de forma telemática al especialista. De este modo se consigue, utilizando la infraestructura ya existente, detectar pacientes con signos de retinopatía diabética de forma precoz. Para ello, se formó a los MF con un curso teórico de 4 horas seguido de lectura de retinografías durante dos meses con examen posterior. Sin embargo, los MF no son especialistas en el análisis de retinografías, por lo que, es posible que fallen en un número considerable de casos. Por otro lado, a pesar del éxito del programa, el hecho de añadir una tarea más al ya gran número de tareas que tienen que realizar los MF supone un reto

Con el ya existente sistema de cribado de la población diabética, se hace evidente la utilidad de un método automático de clasificación de imágenes de fondo de ojo. De modo que, se pueda contar con una evaluación desde el momento en que se toma la retinografía, sin necesidad de esperar a la evaluación por parte del médico de familia en caso de que se vean signos evidentes de retinopatía y sea conveniente remitir directamente al oftalmólogo. Logrando así, descargar al médico de familia de la evaluación de aquellos casos claramente patológicos, y agilizar el proceso de remisión al especialista.

La presente tesis se ocupa del desarrollo de este sistema automático. De forma específica, el objetivo es lograr un sistema capaz de clasificar una imagen de fondo de ojo en función de si presenta o no algún signo de retinopatía diabética.

Este sistema se desarrolló utilizando una red neuronal de clasificación modificada específicamente para el análisis de retinografías. Consiste en un sistema que extrae las características fundamentales que definen cada imagen seguido de un clasificador que produce una probabilidad de haber encontrado algún signo de retinopatía diabética en la imagen analizada.

Para el aprendizaje de este sistema, se partió de la base de datos de retinografías tomadas entre los años 2007 y 2017 bajo el programa Retisalud. De esta base de datos se utilizó un pequeño porcentaje de los datos totales para el desarrollo, manteniendo el resto para la evaluación del modelo.

Una vez implementado el sistema, se realizó un *backtesting* sobre todos los datos que no fueron utilizados para el desarrollo del algoritmo para así analizar su desempeño y estudiar qué hubiera pasado si hubiera funcionado un

algoritmo de cribado automático evaluando cada imagen desde el momento de la captura.

Como resultado de este análisis se obtuvo una sensibilidad superior al 95 % en lo que respecta casos de grado moderado o superior y una especificidad del 85 %. Más aún, se encontró que, potencialmente, hubiera sido posible detectar 1.258 casos de retinopatía de forma prematura, al lograr dar con un diagnóstico patológico antes incluso que el médico de familia. En contraste con estos resultados positivos, se obtuvo una sensibilidad especialmente baja en el caso de las retinopatías leves (70 %). Es decir, 3 de cada 10 pacientes con signos leves de retinopatía no son detectados.Para dar explicación a los errores más significativos, se realizaron una serie de reevaluaciones por un tercer especialista, arrojando como resultados que, de los casos en que el algoritmo automático discrepa con la clasificación del oftalmólogo, el tercer especialista concuerda con el algoritmo automático, de media, en el 33 % de las veces.

Habiendo probado el posible desempeño de un sistema automático de cribado, resulta inevitable plantear un nuevo protocolo de cribado que contemple la utilización de esta nueva herramienta.

6.1.1. Posible línea futura

Como posible línea futura se propone la aplicación del algoritmo automático como herramienta en dos fases. Una primera fase orientada a la verificación del desempeño del sistema en un escenario completamente real y, una segunda fase de aplicación para su utilización.

Si bien en la presente tesis se realizó un *backtesting* para analizar el posible desempeño que hubiera tenido un algoritmo automático, este análisis no deja de ser una simulación de la realidad. En este sentido, implementar un protocolo que contemple la utilización del algoritmo automático en la práctica clínica real resultaría de gran interés, pues, se obtendrían datos relevantes de su auténtico valor.

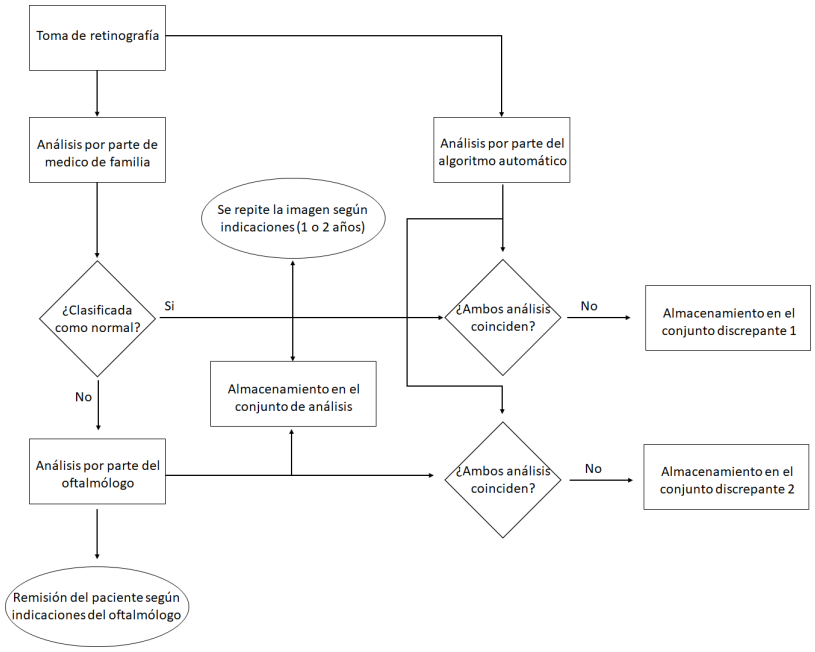


FIGURA 6.1: Posible protocolo de Retisalud extendido para la verificación del algoritmo automático. Sigue los mismos pasos que el protocolo de Retisalud de la figura 3.1 [66] con el añadido de la ejecución en paralelo y de forma independiente del algoritmo de cribado automático.

En la figura 6.1 se presenta un posible protocolo para la verificación. Este nuevo protocolo sigue los mismos pasos que el protocolo original de Retisalud [66] (algunos pasos del diagrama original se simplificaron para dar énfasis en el añadido nuevo) y ejecuta en paralelo y de forma transparente a los médicos el algoritmo de cribado automático al momento de la captura de la retinografía.

Este nueva propuesta de protocolo añade el análisis de la retinografía en el momento de la captura y el almacenamiento de las imágenes tomadas en tres conjuntos diferenciados. El primero, el conjunto de análisis, contiene todas las imágenes tomadas junto con su evaluación por parte del algoritmo de cribado, el médico de familia y el oftalmólogo en caso de que proceda. El segundo, el conjunto discrepante 1, contiene todas aquellas imágenes que el

médico de familia consideró "normales" y que fueron evaluadas por el algoritmo automático como "patológicas". Y el tercero, el conjunto discrepante 2, contiene todas aquellas imágenes donde la evaluación del oftalmólogo y la del algoritmo automático discrepan.

Con este nuevo protocolo se obtiene la posibilidad de recibir datos estadísticos del desempeño del sistema en tiempo real así como nuevos conjuntos de imágenes ya clasificadas.

Con el conjunto de análisis se podrán extraer resultados estadísticos al poder comparar la evaluación de cada uno de los agentes y poder así obtener las métricas del desempeño.

Con el conjunto discrepante 1 sin embargo, no se podrá establecer una medida directa de desempeño al no contar con una evaluación por parte del oftalmólogo en este conjunto. No obstante, este conjunto es de utilidad para poder realizar un análisis en retrospectiva a nivel de paciente, similar a como se realizó en la sección 3.3.4.

Por último, con el conjunto discrepante 2 (conjunto con evaluación por parte del oftalmólogo) que contiene aquellas imágenes en las que el algoritmo falla, se tiene una muestra de la distribución de errores del algoritmo. Con estos datos de errores se pueden diseñar futuros protocolos de reajuste y reentrenamiento para así corregir anomalías y mantener la calidad del modelo a lo largo del tiempo.

Una vez implementado este nuevo protocolo y verificado el desempeño en una situación real podría demostrarse que la precisión se mantiene a lo largo del tiempo. De esta manera, se podría proponer una modificación del protocolo que cuente con la asistencia de esta nueva herramienta de cribado automático.

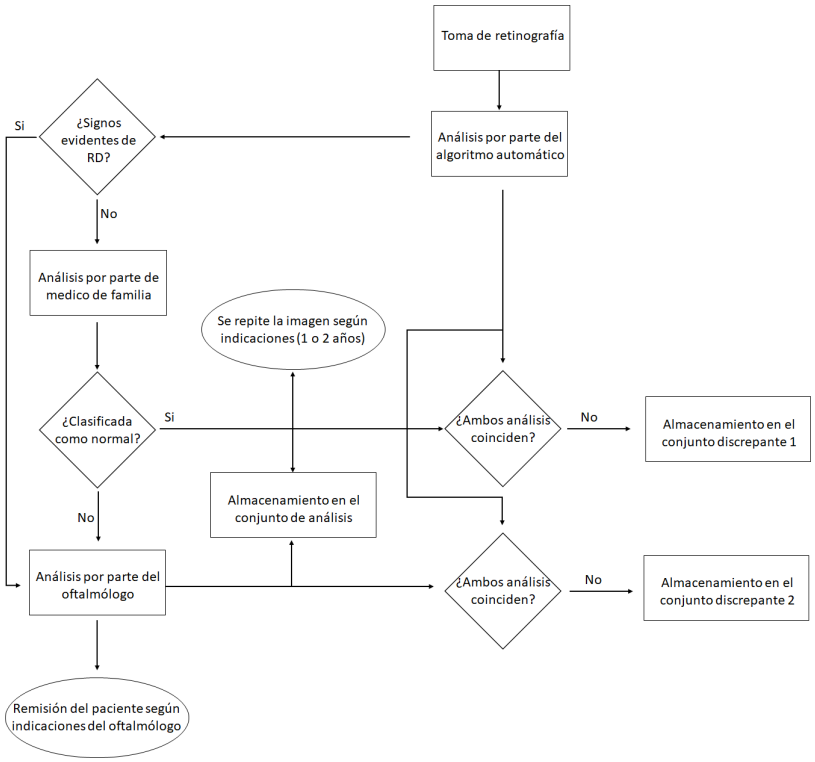


FIGURA 6.2: Posible protocolo de Retisalud extendido para la utilización del algoritmo automático. Sigue los mismos pasos que el protocolo de Retisalud de la figura 6.1 con la diferencia que en este caso, el médico de familia recibe la imagen con un "consejo" de valoración y además, añade la posibilidad de remitir al paciente directamente al especialista en caso de detectar signos de RD evidentes.

En la figura 6.2 se presenta una propuesta de protocolo para la utilización de un algoritmo de cribado automático. Este nuevo proyecto de protocolo mantiene lo añadido en el de la figura 6.1 con la diferencia de que el médico de familia recibe la imagen con un "consejo" de valoración. Además, abre la posibilidad de remitir al paciente directamente al especialista en caso de detectar signos de RD evidentes.

Con este nuevo protocolo, el médico de familia recibiría una imagen de fondo de ojo junto con un "consejo" de valoración para así facilitar su trabajo. No obstante, una simple recomendación no sería suficiente para dar utilidad

al sistema, pues, carecería de una adecuada explicación. En esta tesis, en la sección 3.4 se estudiaron diversos métodos para permitir explicar la clasificación obtenida por el algoritmo automático, logrando dar con dos métodos capaces de informar qué zonas de la imagen fueron relevantes a la hora de definir una retinografía como patológica. El protocolo de la figura 6.2 propone proveer a los médicos de familia de una evaluación junto con cada imagen que resalte qué zonas de la imagen fueron relevantes para la clasificación.

Además, este nuevo protocolo propone la posibilidad de obviar la evaluación por parte del médico de familia en caso de detectar signos evidentes de RD, evitando así retrasar la valoración del oftalmólogo de dichos casos. Sin embargo, para lograr esto, sería necesario añadir un parámetro diferenciador de lo que se considera un signo "evidente" de RD. Este parámetro podría ser un valor mínimo a partir del cual se considera signo evidente, por ejemplo, una predicción de más del 80 % de probabilidad de ser patológica será considerada "signo evidente de RD".

Asimismo, se propone mantener los conjuntos de análisis y los discrepantes 1 y 2 para así seguir con una verificación continua para garantizar la calidad de la clasificación a lo largo del tiempo.

6.2. Reconstrucción de fase de frente de onda

Una de las principales dificultades de la medida de la fase de frente de onda es la dificultad de medirla de forma directa. Hasta el momento, solo existen métodos indirectos para realizar su medición;por ejemplo a partir de sus derivadas.

El estándar *de facto* para la medida de fase de frente de onda es el sensor *Shack-Hartmann* [70, 71]. Este consiste en una cámara con una matriz de micro-lentes interpuesta delante del sensor. Cada micro-lente captura los rayos de luz de una porción del frente de onda. En ausencia de aberración, el lugar del sensor donde cada microlente forma imagen es conocido. Con este conocimiento, es posible obtener las derivadas parciales de la fase de frente de onda al medir la diferencia entre la posición en ausencia de aberraciones y la posición en presencia de éstas.

El resultado de la medida con el sensor *Shack-Hartmann* comprende dos matrices de derivadas parciales en dos dimensiones del espacio. Para lograr obtener el frente de onda es necesario integrar estas derivadas. Es aquí donde

se puede introducir una mejora en la reconstrucción del frente de onda.

En la literatura se han propuesto diversos métodos matemáticos para obtener la fase a partir de sus derivadas. Quizás, los más extendidos son los basados en Mínimos Cuadrados. Estos se basan en asumir una hipótesis de geometría subyacente para luego, mediante la técnica de Mínimos Cuadrados, obtener la fase minimizando el error de reconstrucción. Es aquí, en el asumir una hipótesis fija de forma independiente al contexto del problema donde se encuentran muchos de los errores de reconstrucción. En la literatura, han surgido diversos métodos basados en Mínimos Cuadrados, pero todos cuentan con un mismo patrón: variar la geometría de hipótesis y manteniéndola fija de forma independiente al problema.

En este punto es donde se hace la propuesta en la presente tesis: diseñar un algoritmo que sea adaptable a la geometría subyacente al problema en cuestión, en lugar de uno que mantenga una hipótesis de geometría fija.

Para ello, se diseñó una arquitectura de red neuronal, basándose en la realizada para el problema de retinopatía diabética. Se utilizó un extractor de características similar al utilizado en el capítulo 3 y un decodificador inspirado en la arquitectura RefineNet [131] explicada en la sección 2.4.

Además, se diseño un método para la generación de fases sintéticas junto con sus derivadas analíticas a partir de combinaciones de modos de Zernike. De este modo, se presentó un método para la generación de grandes cantidades de muestras de fase de cualquier distribución deseada (de la atmósfera por ejemplo).

Con este método de generación de fases, se entrenó el modelo propuesto. Una vez finalizado el proceso de entrenamiento, se evaluó y comparó con otros métodos del estado del arte en reconstrucción de fases de frente de onda. Los resultados de estas evaluaciones fueron positivos, pues, el método propuesto logró superar a los algoritmos del estado del arte con los que se comparó en la mayoría de simulaciones, incluso bajo la presencia de ruido o en situaciones en las que el conjunto de *test* se generó con una distribución diferente a la de entrenamiento.

6.2.1. Posible línea futura

Si bien se demostró mediante simulaciones la capacidad del método propuesto, aún queda por demostrar su capacidad en situaciones reales. Una

posibilidad para la evaluación podría ser mediante un sistema óptico que utilice un espejo deformable para generar aberraciones controladas (o al menos lo más controladas posibles). Con este sistema se podrían establecer una serie de experimentos para validar el método.

El primero, podría ser definir una distribución para la generación de aberraciones y capturar una gran cantidad de muestras con este sistema óptico. Luego, utilizar una parte de esta muestra para entrenar un modelo nuevo y con el resto de la muestra validar su desempeño y comparar con otros algoritmos de reconstrucción. Este experimento daría como resultado el desempeño real del método en un entorno controlado.

Como segundo experimento se propone una situación similar al anterior, con la diferencia que, en lugar de entrenar el modelo con datos reales, entrenarlo con datos sintéticos generados con la distribución conocida. Este experimento daría como resultado el desempeño de un modelo generado sintéticamente en una situación real. Este resultado es de especial interés pues, existen casos reales en los que no es posible extraer datos de entrenamiento empíricos, pero si existe una distribución conocida (o al menos estimada o asumida) del tipo de aberraciones que se espera.

Además, como aplicación práctica, este método es especialmente adecuado para el caso de metrología. En esta área, el tipo de fases a medir puede ser de componentes que no siguen una distribución de Zernikes de bajo orden (como pueden ser bordes duros y bien definidos). En este caso en particular ya se demostró que algoritmos de reconstrucción clásica como puede ser el de *Southwell* provocan errores dada su hipótesis de geometría. Por lo que, el método propuesto resulta especialmente adecuado al poder también entrenarse utilizando los datos conocidos del componente a medir.

6.3. *Depth from focus*

La información de distancias de una escena resulta una pieza clave de información a la hora de entender mejor una escena real. En cámaras convencionales, esta información se pierde una vez se ha capturada la escena al traducir cada punto tridimensional una coordenada bidimensional, resultando imposible recuperar la tercera dimensión con certeza.

Dada la importancia de la información de distancias, existen numerosos trabajos en la literatura que afrontan el problema de extracción de distancias con

diferentes técnicas. Una de ellas, es la técnica de *depth from focus*. Esta consiste en, utilizando una única cámara convencional y lente, capturar múltiples imágenes variando la posición de enfoque para luego estimar la información de distancias analizando el enfoque de cada imagen. La presente tesis exploró esta técnica en concreto para proponer una solución que afronte dos de los principales inconvenientes de los métodos basados en esta técnica: a dificultad de generalizar a diferentes cámaras y/o escenas y la sensibilidad al movimiento entre imágenes del mismo *focal stack*.

La solución propuesta parte de una evolución y mezcla de las arquitecturas de red neuronal utilizadas para resolver los dos problemas anteriores. De la arquitectura utilizada para la clasificación de imágenes en el problema del cribado de la retinopatía diabética, se utilizó el extractor de características para diseñar una arquitectura siamesa para la extracción de características de enfoque de cada imagen del *focal stack*. De la arquitectura utilizada para la reconstrucción de fases de frente de onda, se utilizó la metodología de decodificación para diseñar un decodificador que obtenga un mapa de distancias a partir de las características de enfoque de cada imagen.

Además, se diseñó e implementó una metodología para la generación de datos sintéticos para poder entrenar el modelo, siendo estos datos sintéticos suficientemente representativos de situaciones reales.

Mediante experimentación tanto cuantitativa como cualitativa se demostró que el método propuesto es capaz de tratar con imágenes de diferentes cámaras y configuraciones ópticas y, comparado con otros métodos del estado del arte, ofrece un resultado mejor tanto en términos cuantitativos como cualitativos.

Además, se diseñó e implementó un sistema de registro diferenciable interno a la arquitectura de red neuronal capaz de tratar el problema del movimiento entre imágenes del *focal stack*. Este nuevo método se probó con imágenes capturadas de escenas reales, demostrando su capacidad para tratar situaciones con movimiento interno a un *focal stack*.

6.3.1. Posible línea futura

El método propuesto cuenta con una serie de características interesantes:

- Gracias a la información de enfoque, no requiere la utilización de *backbones* pesados, pudiendo utilizar arquitecturas más ligeras, permitiendo así una ejecución rápida. Por ejemplo, para *focal stacks* de 6 planos, en un PC con una Nvidia RTX 2080 tarda 25 ms.
- No requiere de *hardware* específico para obtener la información necesaria a utilizar de entrada. Tan solo hace falta una cámara convencional que cuente con la posibilidad de mover el enfoque manualmente (cámaras de móvil por ejemplo).
- El método es independiente de la óptica de la cámara, siempre que esta sea capaz de obtener imágenes enfocadas en distintas posiciones.

Dadas estas características, como posible línea futura se propone implementar el método propuesto en dispositivos móviles. Esta implementación permitiría capturar fotos y sus correspondientes mapas de distancia utilizando el *hardware* ya existente en el propio dispositivo.

En este sentido, quizás una de las principales limitaciones de una implementación en dispositivo móvil sea la velocidad de captura. Pues, si bien la cámara puede que sea capaz de capturar a una velocidad alta, el movimiento de la lente requiere tiempos de espera más largos. Con una cámara de móvil con lente convencional este tiempo puede llegar a ser de 1 segundo para la captura de 6 imágenes a diferente enfoque. Este largo intervalo entre capturas contribuiría a la aparición de movimiento entre planos del mismo *focal stack* (causados incluso por el propio movimiento de la mano), lo que supondría una barrera infranqueable para otros métodos de *depth from focus*. A pesar de esta lenta captura, el método propuesto demostró ser capaz de lidiar con este tipo de movimiento, mostrándose especialmente adecuado para la aplicación en este tipo de dispositivos.

Si bien se tiene un método capaz de extraer distancias de un *focal stack* aunque este tenga movimiento entre planos, un tiempo de espera de 1 segundo para la captura puede resultar tedioso desde el punto de vista de la experiencia de usuario, y prohibitivo desde el punto de vista de una posible captura de vídeo. Como alternativa para resolver este problema inherente al *hardware* de dispositivos móviles, existen diversas tecnologías de lentes que pueden ser controladas electrónicamente para mover la posición de enfoque a altas velocidades. Ejemplo de ello es la lente líquida de Varioptics C-C-39N0-250 [136]. Si bien este tipo de tecnología no está implantado en el mercado de

forma general, ya existen modelos que incorporan este tipo de lentes como el Xiaomi Mi Mix Fold.

Aunque existan móviles con una tecnología capaz de mover rápidamente el enfoque, difícilmente puedan ser utilizados para la captura a ritmo de vídeo dado el alto coste de recursos que supondría. No obstante, el hecho de que existan este tipo de lentes permitiría la construcción de sistemas de captura de *focal stacks* a ritmo de vídeo. Con esto se obtendría una cámara capaz de extraer información de distancias de secuencias de vídeo y no solo de imágenes estáticas.

6.4. Conclusiones generales

6.4.1. Retinopatía diabética

- Es posible obtener un algoritmo automático de cribado de la retinopatía diabética.
- La utilización de este tipo de algoritmos en los programas de cribado ya existentes ayudaría al sistema de salud a agilizar el protocolo y mejorar la cobertura.
- La información analizada por este algoritmo automático para decidir si la retinografía es patológica o no es la misma que utilizan los médicos. Por tanto, este sistema puede utilizarse como herramienta de apoyo en la evaluación de retinografías.

6.4.2. Reconstrucción de fase de frente de onda

- El asumir una geometría estática para cada tipo de fase a reconstruir provoca errores de estimación.
- Los métodos de la literatura, al evaluarse sobre un conjunto amplio de datos, comienzan a demostrar su sesgo frente a cierto tipo de geometrías.
- Es posible obtener un método de reconstrucción que aprenda la geometría subyacente para lograr mejores estimaciones.

6.4.3. *Depth from focus*

- Los métodos clásicos de *depth from focus* pierden calidad a la hora de generalizar a diferentes cámaras y situaciones.
- Los métodos basados en *deep learning* para *depth from focus* resultan difíciles de llevar a cabo por la falta de datos disponibles.
- Se demostró la posibilidad de generar datos sintéticamente para el entrenamiento de modelos.
- El método propuesto con su variante de registro diferenciable resuelve los problemas inherentes *depth from focus*, el movimiento y la generalización.

6.5. Resultados de la investigación

- S. Ceruso, S. Bonaque-González, A. Pareja-Ríos, J. M. Rodriguez-Ramos, J. G. Marichal-Hernández, D. Carmona-Ballester, and R. Oliva, "Artificial intelligence for the automatic detection of diabetic retinopathy with feedback from key areas", Investigative Ophthalmology &Visual Science, vol. 60, pp. 1435–1435, Jul 2019
- S. Ceruso, S. Bonaque-González, A. Pareja-Ríos, D. Carmona-Ballester, and J. Trujillo-Sevilla, "Reconstructing wavefront phase from measurements of its slope, an adaptive neural network based approach", Optics and Lasers in Engineering, vol. 126, p. 105906, 2020
- S. Ceruso, R. Oliva-Garcia, and J. M. Rodríguez-Ramos, "Method for depth estimation for a variable focus camera.", European Patent Application No EP21382458.4, Wooptix S.L., 2021

Bibliografía

[1] "Real academia española." https://www.rae.es/, 2020. Accessed: 2020-08-18.

[2] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[6] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov 1998.

[7] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010.

[8] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *Journal of Machine Learning Research - Proceedings Track*, vol. 9, pp. 249–256, 01 2010.

[9] E. Lindholm, J. Nickolls, S. Oberman, and J. Montrym, "Nvidia tesla: A unified graphics and computing architecture," *IEEE Micro*, vol. 28, no. 2, pp. 39–55, 2008.

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, p. 1929–1958, Jan. 2014.

[11] M. Ranzato, F. J. Huang, Y. Boureau, and Y. LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[13] H. Iqbal, "Harisiqbal88/plotneuralnet v1.0.0," Dec. 2018.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[15] M. Lin, Q. Chen, and S. Yan, "Network in network," *CoRR*, vol. abs/1312.4400, 2014.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015.

[17] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[18] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[19] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," *CoRR*, vol. abs/1602.07261, 2016.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[21] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," *arXiv preprint arXiv:1611.05431*, 2016.

[22] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *CoRR*, vol. abs/1709.01507, 2017.

[23] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *CoRR*, vol. abs/1311.2901, 2013.

[24] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," 12 2014.

[25] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," *CoRR*, vol. abs/1703.01365, 2017.

[26] B. Zhou, A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," *CoRR*, vol. abs/1512.04150, 2015.

[27] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization," *CoRR*, vol. abs/1610.02391, 2016.

[28] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *CoRR*, vol. abs/1411.4038, 2014.

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015.

[30] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *CoRR*, vol. abs/1511.00561, 2015.

[31] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," *CoRR*, vol. abs/1612.01105, 2016.

[32] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," 2010.

[33] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on*

Pattern Analysis and Machine Intelligence, vol. 37, no. 9, pp. 1904–1916, 2015.

[35] F. Yu, V. Koltun, and T. A. Funkhouser, "Dilated residual networks," *CoRR*, vol. abs/1705.09914, 2017.

[36] G. Lin, A. Milan, C. Shen, and I. D. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," *CoRR*, vol. abs/1611.06612, 2016.

[37] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *CoRR*, vol. abs/1606.00915, 2016.

[38] L. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *CoRR*, vol. abs/1706.05587, 2017.

[39] V. Harjutsalo, L. Sjöberg, and J. Tuomilehto, "Time trends in the incidence of type 1 diabetes in finnish children: a cohort study," *The Lancet*, vol. 371, no. 9626, pp. 1777 – 1782, 2008.

[40] J. Shaw, R. Sicree, and P. Zimmet, "Global estimates of the prevalence of diabetes for 2010 and 2030," vol. 87, pp. 4–14, 11 2009.

[41] I. D. Federation, *IDF Diabetes Atlas*. International Diabetes Federation, 2013.

[42] F. Soriguer, A. Goday, A. Bosch-Comas, E. Bordiú, A. Calle-Pascual, R. Carmena, R. Casamitjana, L. Castaño, C. Castell, M. Catalá, E. Delgado, J. Franch, S. Gaztambide, J. Girbés, R. Gomis, G. Gutiérrez, A. López-Alba, M. T. Martínez-Larrad, E. Menéndez, I. Mora-Peces, E. Ortega, G. Pascual-Manich, G. Rojo-Martínez, M. Serrano-Rios, S. Valdés, J. A. Vázquez, and J. Vendrell, "Prevalence of diabetes mellitus and impaired glucose regulation in spain: the di@bet.es study," *Diabetologia*, vol. 55, pp. 88–93, Jan 2012.

[43] N. Congdon, B. O'Colmain, C. Klaver, R. Klein, B. Muñoz, D. S Friedman, J. Kempen, H. R Taylor, and P. Mitchell, "Causes and prevalence of visual impairment among adults in the united states," vol. 122, pp. 477–85, 04 2004.

[44] C.-F. Chou, M. F. Cotch, S. Vitale, X. Zhang, R. Klein, D. S Friedman, B. E.K. Klein, and J. Saaddine, "Age-related eye diseases and visual impairment among u.s. adults," vol. 45, pp. 29–35, 07 2013.

[45] C. Disease Control and P. , *CDC, National Diabetes Fact Sheet: General information and National Estimates on Diabetes in the United States*. 01 2011.

[46] E. Summary, "Standards of medical care in diabetes-2010," vol. 33, pp. S4–S10, 01 2010.

[47] A. A. of Ophthalmology Retina Panel., "Preferred practice pattern guidelines: diabetic retinopathy ppp," 2008.

[48] N. Cheung, P. Mitchell, and T. Y. Wong, "Diabetic retinopathy," *The Lancet*, vol. 376, no. 9735, pp. 124 – 136, 2010.

[49] N. Hautala, R. Aikkila, J. Korpelainen, A. Keskitalo, A. Kurikka, A. Falck, R. Bloigu, and H. Alanko, "Marked reductions in visual impairment due to diabetic retinopathy achieved by efficient screening and timely treatment," vol. 92, 10 2013.

[50] N. Hautala, P. Hyytinen, V. Saarela, P. Hägg, A. Kurikka, M. Runtti, and A. Tuulonen, "A mobile eye unit for screening of diabetic retinopathy and follow-up of glaucoma in remote locations in northern finland," vol. 87, pp. 912–3, 07 2009.

[51] I. Jivraj, M. Ng, C. Rudnisky, B. Dimla, E. Tambe, N. Nathoo, and M. T.S. Tennant, "Prevalence and severity of diabetic retinopathy in northwest cameroon as identified by teleophthalmology," vol. 17, pp. 294–8, 04 2011.

[52] J. Peng, H. Zou, W. Wang, J. Fu, B. Shen, X. Bai, X. Xu, and X. Zhang, "Implementation and first-year screening results of an ocular telehealth system for diabetic retinopathy in china," *BMC Health Services Research*, vol. 11, p. 250, Oct 2011.

[53] M. José Sender Palacios, M. Vernet Vernet, M. Maseras Bové, A. Playà, L. Pascual Batlle, J. C. Ondategui-Parra, and E. Jovell Fernández, "Oftalmopatía en la diabetes mellitus: detección desde la atención primaria de salud," vol. 43, p. 41–48, 01 2011.

[54] N. Deb-Joardar, N. Germain, G. Thuret, P. Manoli, A.-F. Garcin, L. Millot, Y. Gavet, B. Estour, and P. Gain, "Screening for diabetic retinopathy by ophthalmologists and endocrinologists with pupillary dilation and a nonmydriatic digital camera," *American Journal of Ophthalmology*, vol. 140, no. 5, pp. 814 – 821.e2, 2005.

[55] J. Beynat, A. Charles, K. Astruc, P. Metral, L. Chirpaz, A.-M. Bron, and C. Creuzot-Garcher, "Screening for diabetic retinopathy in a rural french population with a mobile non-mydriatic camera," *Diabetes & Metabolism*, vol. 35, no. 1, pp. 49 – 56, 2009.

[56] T. S. Surendran and R. Raman, "Teleophthalmology in diabetic retinopathy," *Journal of Diabetes Science and Technology*, vol. 8, no. 2, pp. 262–266, 2014. PMID: 24876576.

[57] J. Lopez-Bastida, F. Cabrera-Lopez, and P. Serrano-Aguilar, "Sensitivity and specificity of digital retinal imaging for screening diabetic retinopathy," vol. 24, pp. 403–7, 05 2007.

[58] F. C. López, P. I. C. Guerra, J. L. Bastida, and J. D. Arriaga, "Evaluación de la efectividad y coste-efectividad de la imagen digital en el diagnóstico de la retinopatía diabética.," *Archivos de la Sociedad Canaria de Oftalmología*, vol. 15, pp. 21–31, 2004.

[59] J. E. Chasan, B. Delaune, A. Y. Maa, and M. G. Lynch, "Effect of a teleretinal screening program on eye care use and resources.," *JAMA ophthalmology*, vol. 132 9, pp. 1045–51, 2014.

[60] S. Jones and R. T Edwards, "Diabetic retinopathy screening: A systematic review of the economic evidence," vol. 27, pp. 249–56, 03 2010.

[61] J. D Whited, S. Datta, L. M Aiello, L. Aiello, J. Cavallerano, P. R Conlin, M. Horton, R. A Vigersky, R. Poropatich, P. Challa, A. Darkins, and S.-E. Bursell, "A modeled economic analysis of a digital teleophthalmology system as used by three federal healthcare agencies for detecting proliferative diabetic retinopathy," vol. 11, pp. 641–51, 01 2006.

[62] J. Javitt, L. Aiello, Y. Chiang, F. Ferris, J. Canner, and S. Greenfield, "Preventive eye care in people with diabetes is cost-saving to the federal government: Implications for health-care reform," vol. 17, pp. 909–17, 09 1994.

[63] W. G. Baxt, "Use of an artificial neural network for the diagnosis of myocardial infarction," vol. 115, pp. 843–8, 01 1992.

[64] M. Karabatak and M. C. Ince, "An expert system for detection of breast cancer based on association rules and neural network," *Expert Systems with Applications*, vol. 36, no. 2, Part 2, pp. 3465 – 3469, 2009.

[65] T. Williamson, G. Gardner, D. Keating, C. Kirkness, and A. Elliott, "Automatic detection of diabetic retinopathy using neural networks," vol. 37, p. S973, 02 1996.

[66] A. Ríos, S. Bonaque, M. Serrano-García, F. Cabrera-López, P. Abreu-Reyes, and M. Marrero-Saavedra, "Tele-ophthalmology for diabetic retinopathy screening: 8 years of experience," vol. 92, 10 2016.

[67] G. V, P. L, C. M, and et al, "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *JAMA*, vol. 316, no. 22, pp. 2402–2410, 2016.

[68] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014.

[69] S. Ceruso, S. Bonaque-González, A. Pareja-Ríos, J. M. Rodriguez-Ramos, J. G. Marichal-Hernández, D. Carmona-Ballester, and R. Oliva, "Artificial intelligence for the automatic detection of diabetic retinopathy with feedback from key areas," *Investigative Ophthalmology & Visual Science*, vol. 60, pp. 1435–1435, Jul 2019.

[70] L. N. Thibos, "Principles of hartmann-shack aberrometry," in *Vision Science and its Applications*, p. NW6, Optical Society of America, 2000.

[71] B. C. Platt and R. B. Shack, "History and principles of shack-hartmann wavefront sensing.," *Journal of refractive surgery*, vol. 17 5, pp. S573–7, 2001.

[72] J. W. Hardy and A. J. MacGovern, "Shearing interferometry: a flexible technique for wavefront measurement," in *Interferometric Metrology*, vol. 816, pp. 180–196, International Society for Optics and Photonics, 1987.

[73] B. C. Platt and R. B. Shack, "History and principles of shack-hartmann wavefront sensing.," *Journal of refractive surgery*, vol. 17 5, pp. S573–7, 2001.

[74] L. N. Thibos, "Principles of hartmann-shack aberrometry," in *Vision Science and its Applications*, p. NW6, Optical Society of America, 2000.

[75] W. Southwell, "Wave-front estimation from wave-front slope measurements," *J. Opt. Soc. Am.*, vol. 70, pp. 998–1006, Aug 1980.

[76] R. H. Hudgin, "Optimal wave-front estimation," *J. Opt. Soc. Am.*, vol. 67, pp. 378–382, Mar 1977.

[77] D. L. Fried, "Least-square fitting a wave-front distortion estimate to an array of phase-difference measurements," *J. Opt. Soc. Am.*, vol. 67, pp. 370–375, Mar 1977.

[78] A. Talmi and E. N. Ribak, "Wavefront reconstruction from its gradients," *J. Opt. Soc. Am. A*, vol. 23, pp. 288–297, Feb 2006.

[79] S. Ettl, J. Kaminski, M. C. Knauer, and G. Häusler, "Shape reconstruction from gradient data," *Appl. Opt.*, vol. 47, pp. 2091–2097, Apr 2008.

[80] P. Bon, S. Monneret, and B. Wattellier, "Noniterative boundary-artifact-free wavefront reconstruction from its derivatives," *Appl. Opt.*, vol. 51, pp. 5698–5704, Aug 2012.

[81] L. Huang and A. Asundi, "Improvement of least-squares integration method with iterative compensations in fringe reflectometry," *Appl. Opt.*, vol. 51, pp. 7459–7465, Nov 2012.

[82] G. Li, Y. Li, K. Liu, X. Ma, and H. Wang, "Improving wavefront reconstruction accuracy by using integration equations with higher-order truncation errors in the southwell geometry," *J. Opt. Soc. Am. A*, vol. 30, pp. 1448–1459, Jul 2013.

[83] H. Ren, F. Gao, and X. Jiang, "Improvement of high-order least-squares integration method for stereo deflectometry," *Appl. Opt.*, vol. 54, pp. 10249–10255, Dec 2015.

[84] L. Huang, J. Xue, B. Gao, C. Zuo, and M. Idir, "Spline based least squares integration for two-dimensional shape or wavefront reconstruction," *Optics and Lasers in Engineering*, vol. 91, pp. 221 – 226, 2017.

[85] A. Kolmogorov, "The local structure of turbulence in incompressible viscous fluid for very large Reynolds numbers," *Doklady Akademii Nauk Sssr*, vol. 30, no. 1890, pp. 301–305, 1941.

[86] R. Lane, A. Glindemann, J. Dainty, *et al.*, "Simulation of a kolmogorov phase screen," *Waves in random media*, vol. 2, no. 3, pp. 209–224, 1992.

[87] L. Llorente, S. Marcos, C. Dorronsoro, and S. A. Burns, "Effect of sampling on real ocular aberration measurements," *JOSA A*, vol. 24, no. 9, pp. 2783–2796, 2007.

[88] L. A. Poyneer, D. T. Gavel, and J. M. Brase, "Fast wave-front reconstruction in large adaptive optics systems with use of the fourier transform," *JOSA A*, vol. 19, no. 10, pp. 2100–2111, 2002.

[89] S. Ceruso, S. Bonaque-González, A. Pareja-Ríos, D. Carmona-Ballester, and J. Trujillo-Sevilla, "Reconstructing wavefront phase from measurements of its slope, an adaptive neural network based approach," *Optics and Lasers in Engineering*, vol. 126, p. 105906, 2020.

[90] B. Zhou, P. Krähenbühl, and V. Koltun, "Does computer vision matter for action?," *CoRR*, vol. abs/1905.12887, 2019.

[91] G. Percoco and A. J. S. Salmerón, "Photogrammetric measurement of 3d freeform millimetre-sized objects with micro features: an experimental validation of the close-range camera calibration model for narrow angles of view," *Measurement Science and Technology*, vol. 26, p. 095203, jul 2015.

[92] M. Yakar, "Using close range photogrammetry to measure the position of inaccessible geological features," *Experimental Techniques*, vol. 35, pp. 54 – 59, 09 2009.

[93] G. Percoco and A. J. S. Salmerón, "Photogrammetric measurement of 3d freeform millimetre-sized objects with micro features: an experimental validation of the close-range camera calibration model for narrow angles of view," *Measurement Science and Technology*, vol. 26, p. 095203, jul 2015.

[94] F. Remondino, A. Guarnieri, and A. Vettore, "3d modeling of close-range objects: Photogrammetry or laser scanning," *Proc SPIE*, vol. 5665, pp. 216–225, 12 2004.

[95] M. Samaan, R. Héno, and M. Deseilligny, "Close-range photogrammetric tools for small 3d archeological objects," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-5/W2, pp. 549–553, 07 2013.

[96] L. Lastilla, R. Ravanelli, and S. Ferrara, "3d high-quality modeling of small and complex archaeological inscribed objects: Relevant issues and proposed methodology," vol. XLII-2/W11, 05 2019.

[97] J.-C. Huang, C.-S. Liu, P.-J. Chiang, W.-Y. Hsu, J.-L. Liu, B.-H. Huang, and S.-R. Lin, "Design and experimental validation of novel 3d optical scanner with zoom lens unit," *Measurement Science and Technology*, vol. 28, p. 105904, sep 2017.

[98] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE Multimedia - IEEEMM*, vol. 19, pp. 4–10, 02 2012.

[99] J. A. Christian and S. P. Cryan, "A survey of lidar technology and its use in spacecraft relative navigation," 2013.

[100] L. Keselman, J. I. Woodfill, A. Grunnet-Jepsen, and A. Bhowmik, "Intel realsense stereoscopic depth cameras," *CoRR*, vol. abs/1705.05548, 2017.

[101] G. A. Atkinson, M. F. Hansen, M. L. Smith, and L. N. Smith, "A efficient and practical 3d face scanner using near infrared and visible photometric stereo," *Procedia Computer Science*, vol. 2, pp. 11 – 19, 2010. Proceedings of the International Conference and Exhibition on Biometrics Technology.

[102] O. Aubreton, A. Bajard, B. Verney, and F. Truchetet, "Infrared system for 3d scanning of metallic surfaces," *Machine Vision and Applications*, vol. 24, 10 2013.

[103] Y. Wang, Z. Lai, G. Huang, B. H. Wang, L. van der Maaten, M. Campbell, and K. Q. Weinberger, "Anytime stereo image depth estimation on mobile devices," *CoRR*, vol. abs/1810.11408, 2018.

[104] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, 2008.

[105] K. Konolige, "Small vision systems: Hardware and implementation," in *Robotics Research* (Y. Shirai and S. Hirose, eds.), (London), pp. 203–212, Springer London, 1998.

[106] P. Nyimbili, H. Demirel, D. Seker, and T. Erden, "Structure from motion (sfm) - approaches and applications," 09 2016.

[107] J. L. Schönberger and J. Frahm, "Structure-from-motion revisited," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4104–4113, 2016.

[108] O. Özyesil, V. Voroninski, R. Basri, and A. Singer, "A survey on structure from motion," *CoRR*, vol. abs/1701.08493, 2017.

[109] K. Xian, C. Shen, Z.-G. Cao, H. Lu, Y. Xiao, R. Li, and Z. Luo, "Monocular relative depth perception with web stereo data supervision," 09 2018.

[110] C. Wang, S. Lucey, F. Perazzi, and O. Wang, "Web stereo video supervision for depth prediction from dynamic scenes," *CoRR*, vol. abs/1904.11112, 2019.

[111] Z. Li and N. Snavely, "Megadepth: Learning single-view depth prediction from internet photos," *CoRR*, vol. abs/1804.00607, 2018.

[112] K. Lasinger, R. Ranftl, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *CoRR*, vol. abs/1907.01341, 2019.

[113] S. Pertuz, D. Puig, and M. García, "Analysis of focus measure operators in shape-from-focus," *Pattern Recognition*, vol. 46, 11 2012.

[114] M. T. Mahmood, "Shape from focus by total variation," in *IVMSP 2013*, pp. 1–4, 2013.

[115] M. Möller, M. Benning, C. Schönlieb, and D. Cremers, "Variational depth from focus reconstruction," *CoRR*, vol. abs/1408.0173, 2014.

[116] A. Muhammad and T.-S. Choi, "Learning shape from focus using multilayer neural networks," in *Vision Geometry VIII* (L. J. Latecki, R. A. Melter, D. M. Mount, and A. Y. Wu, eds.), vol. 3811, pp. 366 – 375, International Society for Optics and Photonics, SPIE, 1999.

[117] C. Hazirbas, S. G. Soyer, M. C. Staab, L. Leal-Taixé, and D. Cremers, "Deep depth from focus," in *Asian Conference on Computer Vision (ACCV)*, December 2018.

[118] I. Vasiljevic, N. Kolkin, S. Zhang, R. Luo, H. Wang, F. Z. Dai, A. F. Daniele, M. Mostajabi, S. Basart, M. R. Walter, and G. Shakhnarovich, "DIODE: A Dense Indoor and Outdoor DEpth Dataset," *CoRR*, vol. abs/1908.00463, 2019.

[119] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *IEEE International*

Conference on Computer Vision and Pattern Recognition (CVPR), 2016. arXiv:1512.02134.

[120] Y. Kim, H. Jung, D. Min, and K. Sohn, "Deep monocular depth estimation via integration of global and local predictions," *IEEE Transactions on Image Processing*, vol. 27, no. 8, pp. 4131–4144, 2018.

[121] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

[122] K. Xian, C. Shen, Z. Cao, H. Lu, Y. Xiao, R. Li, and Z. Luo, "Monocular relative depth perception with web stereo data supervision," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[123] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)* (A. Fitzgibbon et al. (Eds.), ed.), Part IV, LNCS 7577, pp. 611–625, Springer-Verlag, Oct. 2012.

[124] D. Scharstein and C. Pal, "Learning conditional random fields for stereo," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[125] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[126] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," in *Pattern Recognition* (X. Jiang, J. Hornegger, and R. Koch, eds.), (Cham), pp. 31–42, Springer International Publishing, 2014.

[127] J. T. Barron and B. Poole, "The fast bilateral solver," *CoRR*, vol. abs/1511.03296, 2015.

[128] M. Potmesil and I. Chakravarty, "A lens and aperture camera model for synthetic image generation," *SIGGRAPH Comput. Graph.*, vol. 15, p. 297–305, Aug. 1981.

[129] M. Kraus and M. Strengert, "Depth-of-field rendering by pyramidal image processing," *Comput. Graph. Forum*, vol. 26, pp. 645–654, 09 2007.

[130] P. H. O. Pinheiro, T. Lin, R. Collobert, and P. Dollár, "Learning to refine object segments," *CoRR*, vol. abs/1603.08695, 2016.

[131] G. Lin, A. Milan, C. Shen, and I. D. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," *CoRR*, vol. abs/1611.06612, 2016.

[132] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," *CoRR*, vol. abs/1612.01105, 2016.

[133] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry, "End-to-end learning of geometry and context for deep stereo regression," *CoRR*, vol. abs/1703.04309, 2017.

[134] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[135] "Flare 2MP." http://www.ioindustries.com/flare2mp.html. [Online; accessed 05-October-2020].

[136] "C-C-39N0-250." https://www.corning.com/worldwide/en/products/advanced-optics/product-materials/corning-varioptic-lenses/auto-focus-lens-modules-c-c-series/varioptic-C-C-39N0-250.html. [Online; accessed 29-May-2021].

[137] "Plano-Convex Lens Kit." https://www.thorlabs.com/thorproduct.cfm?partnumber=LSC01-A. [Online; accessed 19-October-2020].

[138] "FFmpeg." http://www.http://ffmpeg.org/. [Online; accessed 16-May-2021].

[139] Z. Teed and J. Deng, "RAFT: recurrent all-pairs field transforms for optical flow," *CoRR*, vol. abs/2003.12039, 2020.

[140] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017.

[141] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, and A. Ku, "Image transformer," *CoRR*, vol. abs/1802.05751, 2018.

[142] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit,

and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020.

[143] S. Ceruso, R. Oliva-Garcia, and J. M. Rodríguez-Ramos, "Method for depth estimation for a variable focus camera."