UNIVERSIDAD D CÓRDOBA

SISTEMA AUTOMÁTICO PARA LA DETECCIÓN Y CLASIFICACIÓN DE GRIETAS EN PAVIMENTOS

≪≫

AUTOMATIC SYSTEM FOR PAVEMENT CRACK DETECTION AND CLASSIFICATION

Programa de doctorado:

COMPUTACIÓN AVANZADA, ENERGÍA Y PLASMAS

Tesis presentada por:

FRANCISCO JAVIER RODRÍGUEZ LOZANO

Tesis dirigida por:

JOAQUÍN OLIVARES BUENO

JOSÉ MANUEL PALOMARES MUÑOZ

Fecha de depósito:

1 de Julio 2020.

TITULO: *Sistema automático para la detección y clasificación de grietas en pavimentos*

AUTOR: *Francisco Javier Rodríguez Lozano*

© Edita: UCOPress. 2020
Campus de Rabanales
Ctra. Nacional IV, Km. 396 A
14071 Córdoba

https://www.uco.es/ucopress/index.php/es/
ucopress@uco.es

VISTO BUENO

El **Dr. José Manuel Palomares Muñoz**, Profesor Contratado Doctor y el **Dr. Joaquín Olivares Bueno**, Titular de Universidad, ambos del Departamento de Ingeniería Electrónica y de Computadores de la Universidad de Córdoba,

CERTIFICAN:

Que el presente documento titulado:

"Sistema automático para la detección y clasificación de grietas en pavimentos"

ha sido realizada por **D. Francisco Javier Rodríguez Lozano** bajo nuestra dirección en el Departamento de Ingeniería Electrónica y de Computadores de la Universidad de Córdoba para optar al grado de **Doctor con Mención Internacional** .

En Córdoba, a 1 de Julio 2020.

Los Directores de la tesis doctoral:

PALOMARES MUÑOZ JOSE MANUEL - 23806151R

Firmado digitalmente por PALOMARES MUÑOZ JOSE MANUEL - 23806151R
Fecha: 2020.07.01 09:33:03 +02'00'

Fdo. José Manuel Palomares Muñoz

OLIVARES BUENO JOAQUIN - 26487222P

Firmado digitalmente por OLIVARES BUENO JOAQUIN - 26487222P
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-26487222P, givenName=JOAQUIN, sn=OLIVARES BUENO, cn=OLIVARES BUENO JOAQUIN - 26487222P
Fecha: 2020.07.01 09:25:44 +02'00'

Fdo. Joaquín Olivares Bueno

INFORME DE LOS DIRECTORES

Título de la tesis: Sistema automático para la detección y clasificación de grietas en pavimentos.

Doctorando: D. Francisco Javier Rodríguez Lozano.

Informe razonado de los directores de la tesis:

El doctorando ha realizado un trabajo de investigación novedoso en el que ha mostrado la evolución personal y académica desde ingeniero hasta investigador. Ha realizado numerosas propuestas y discusiones con los directores que han llevado a obtener las aportaciones científicas incluidas en esta tesis doctoral. El doctorando ha realizado una revisión bibliográfica de antecedentes científico-técnicos relativos al campo de investigación de la tesis, adquiriendo experiencia en el estado de la técnica.

Con ello se convierte en un experto en los campos de procesamiento de imágenes y la inteligencia artificial. Se ha enfrentado a la problemática de obtener grandes conjuntos de datos con los que poder experimentar, aportando así nuevos datasets útiles para la comunidad científica. Culmina el trabajo de tesis con la implementación de diversos algoritmos de procesamiento, detección y clasificación de imágenes aplicados a la detección de grietas en calzadas asfaltadas.

Los excelentes resultados obtenidos se han publicado en la revista con más alta clasificación del área de ingeniería civil, siendo la quinta en la temática correspondiente del campo de la informática, con ello, la propuesta queda sólidamente sustentada.

Además, el doctorando ha realizado estancias en un centro de investigación internacional de prestigio en el extranjero durante más de tres meses, con el objetivo de optar a la mención internacional del título de doctor. En dichas estancias, el doctorando ha adquirido y refinado las habilidades necesarias para desarrollar una investigación de calidad en el ámbito científico internacional.

Queda así demostrada la capacidad del doctorando en el ámbito investigador.

Por todo ello, se autoriza la presentación de la tesis doctoral.

En Córdoba, a 1 de Julio 2020.

Los Directores de la tesis doctoral:

PALOMARES MUÑOZ JOSE MANUEL - 23806151R

Firmado digitalmente por PALOMARES MUÑOZ JOSE MANUEL - 23806151R
Fecha: 2020.07.01 09:38:26 +02'00'

Fdo. José Manuel Palomares Muñoz

OLIVARES BUENO JOAQUIN - 26487222P

Firmado digitalmente por OLIVARES BUENO JOAQUIN - 26487222P
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-26487222P, givenName=JOAQUIN, sn=OLIVARES BUENO, cn=OLIVARES BUENO JOAQUIN - 26487222P
Fecha: 2020.07.01 09:26:19 +02'00'

Fdo. Joaquín Olivares Bueno

DECLARACIÓN

El doctorando y los directores de la presente Tesis Doctoral "***Sistema automático para la detección y clasificación de grietas en pavimentos***", garantizan, al firmar la misma, que el trabajo realizado es inédito (salvo por las publicaciones científicas derivadas de los presentes autores) y que hasta donde su conocimiento alcanza, respeta los derechos de otros autores al ser citados, cuando se han utilizado sus resultados o publicaciones.

Asimismo, se declara que todos los datos expuestos en esta investigación no han sido falseados y que cualquier error que pudiera existir en el documento, no ha sido introducido conscientemente.

En Córdoba, a 1 de Julio 2020.

El doctorando:

RODRIGUEZ LOZANO FRANCISCO JAVIER - 80158257Z

Firmado digitalmente por RODRIGUEZ LOZANO FRANCISCO JAVIER - 80158257Z
Fecha: 2020.07.01 02:57:25 +02'00'

Fdo. Francisco Javier Rodríguez Lozano

En Córdoba, a 1 de Julio 2020.

Los Directores de la tesis doctoral:

PALOMARES MUÑOZ JOSE MANUEL - 23806151R

Firmado digitalmente por PALOMARES MUÑOZ JOSE MANUEL - 23806151R
Fecha: 2020.07.01 09:39:02 +02'00'

Fdo. José Manuel Palomares Muñoz

OLIVARES BUENO JOAQUIN - 26487222P

Firmado digitalmente por OLIVARES BUENO JOAQUIN - 26487222P
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-26487222P, givenName=JOAQUIN, sn=OLIVARES BUENO, cn=OLIVARES BUENO JOAQUIN - 26487222P
Fecha: 2020.07.01 09:26:45 +02'00'

Fdo. Joaquín Olivares Bueno

INDICIOS DE CALIDAD

Siguiendo los requisitos de calidad impuestos por la normativa de la *Universidad de Córdoba* y su programa de doctorado en *Computación avanzada, energía y plasmas* para la presentación de Tesis Doctorales, a continuación se detalla la información de la publicación principal derivada de esta Tesis Doctoral:

- **Nombre de la Publicación:** *Benefits of ensemble models in road pavement cracking classification.*
- **Autores:** *F. J. Rodriguez-Lozano, F. León-García, J. C. Gámez-Granados, J. M. Palomares, y J. Olivares.*
- **Revista y año:** *Computer-Aided Civil and Infrastructure Engineering, Feb. 2020.*
- **DOI:** *10.1111/mice.12543*
- **Base de datos Internacional en la que está indexada:** *InCites Journal Citation Reports (JCR)*
- **Áreas temáticas en la base de datos de referencia:**
  - *Computer Science, Interdisciplinary Applications.*
  - *Construction & Building Technology.*
  - *Engineering, Civil.*
  - *Transportation Science & Technology.*
- **Índice de impacto de la revista en el año de publicación:** *8.552 (año 2019)*
- **Lugar que ocupa/nº de revistas por áreas temáticas:**
  - *5/109 (Computer Science, Interdisciplinary Applications) - Q1 (1^{er} Decil).*
  - *1/63 (Construction & Building Technology) - Q1 (1^{er} Decil).*
  - *1/134 (Engineering, Civil) - Q1 (1^{er} Decil).*
  - *1/36 (Transportation Science & Technology) - Q1 (1^{er} Decil).*

En memoria de:
Mi padrino, Jacinto Pérez Ventura.
Mi abuelo, Pedro Lozano Moreno.
Mi madrina, Isabel Perea Rincón.

Si un hombre no está agradecido por lo que tiene,
es probable que no sea agradecido por lo que tendrá.
Frank A. Clark [1]

AGRADECIMIENTOS

En primer lugar, debo expresar mi más sincero agradecimiento a mi padre, *Francisco Rodríguez Ferrera*, y a mi madre, *Isabel Lozano González*, por su dedicación, confianza y disposición en todo momento, y todo sea dicho, por aguantarme durante los momentos de altibajos, estrés y mal genio, que seguro que habrán sido más ocasiones de las que soy capaz de recordar. Lo mismo para el resto de personas que considero parte de mi familia.

Quiero agradecer también a *Juan Carlos Gámez Granados* y a *José Manuel Soto Hidalgo* por todo lo que me han apoyado, enseñado y por escucharme en cada momento. !Sois currantes como ninguno!

Tampoco me olvido de agradecerle a *Fernando León García* que llegó al final del camino del doctorado hace poco más de medio año. ¡Ya mismo estamos codificando y publicando todas las ideas que dejamos apuntadas!.

Por otro lado, agradecer a mis directores de tesis *Joaquín Olivares Bueno* y *José Manuel Palomares Muñoz* por permitirme recorrer este camino, y por sus continuas revisiones en los artículos que he elaborado así como sus comentarios siempre constructivos.

Quiero agradecer también a *Rafael Palomar Ávalos* y *Ole Jacob Elle* por ayudarme en todo lo posible en mis estancias realizadas en *The Intervention Centre* Oslo (Noruega). Tengo que admitir que era la primera vez que viajaba al extranjero y creo que mi experiencia no habría sido lo mismo sin ellos.

Por último, quisiera mostrar mis agradecimientos, a menudo olvidados, a las personas que han revisado la tesis doctoral para poder optar a la mención internacional, así como a los miembros del tribunal que desarrollan esta labor sin compensación por sus horas dedicadas más allá de el agradecimiento.

Sólo me queda pedir disculpas a aquellos que no haya mencionado, no porque no haya querido incluirlos, sino porque no he caído en la cuenta mientras redactaba este texto.

¡¡MUCHAS GRACIAS A TODOS!!

1 Frank A. Clark (1860 - 1936) abogado y político que ejerció medio siglo en el ámbito público y privado.

RESUMEN

Las carreteras son un tipo de elemento urbanístico utilizado por millones de personas a diario, y su estado en condiciones óptimas favorece la disminución de la tasa de accidentes de tráfico. El estado de la superficie del asfalto se ve alterado por un amplio abanico de defectos, y en concreto, las grietas cobran un interés especial debido a que su tratamiento en fases tempranas pueden suponer un ahorro en el coste de reparación y tratamiento del defecto en etapas posteriores, así como evitar la aparición de defectos derivados de ellas. Por este motivo, el mantenimiento de los pavimentos juega un papel fundamental tanto en la seguridad de los usuarios de este tipo de vías, como en términos económicos. Sin embargo, a pesar de la importancia que tiene, existen millones de kilómetros que necesitan ser inspeccionados, y esta labor se realiza en la mayoría de lo casos de forma manual mediante la inspección visual supervisada por expertos, siendo una tarea ineficiente en el tiempo. Por ello, esta Tesis Doctoral presenta un sistema para la detección y clasificación automática de defectos de grietas en pavimentos. Para ello, se aplican métodos de procesamiento de imágenes a las capturas tomadas de la superficie de los asfaltos para la extracción de características y su posterior optimización y representación a un nuevo espacio de atributos interpretables por una persona. Posteriormente estas características son utilizadas por un ensemble de modelos compuesto por varios algoritmos de aprendizaje automático, para realizar la clasificación de las grietas en sus tipos más comunes: grietas de tipo malla o cocodrilo, grietas longitudinales y grietas transversales. De acuerdo con los experimentos realizados y los resultados obtenidos, el sistema tiene la capacidad de trabajar en sistemas computacionales de recursos limitados, siendo susceptible de emplearse con restricciones de tiempo real, además de proporcionar mejores resultados frente a las propuestas existentes en la literatura científica. Esto hace posible que el sistema se pueda colocar en diferentes vehículos no especializados para la recolección y clasificación de los defectos en el mismo lugar donde ocurren, aliviando así la tareas llevadas a cabo por los expertos.

ABSTRACT

Roads are a type of urban element used by millions of people every day, and the optimal surface condition contributes to the reduction of the rate of traffic accidents. The condition of the asphalt surface is affected by a wide range of defects, and particularly, cracks have a special interest because their treatment in early stages represent savings in the repairing costs. As this prevents the appearance of defects derived from them rather than treating the defects in later stages. For this reason, pavement maintenance is fundamental in economic terms and the safety of the users. Millions of kilometers need to be examined, however, this work is mostly done manually through visual inspection supervised by experts, which is a time inefficient task. For this reason,

this Doctoral Thesis presents a system for the automatic detection and classification of cracking defects in pavements. For this purpose, image processing methods are applied to asphalt surface images extracting features of the cracks, reducing the number of features, and representing the cracks in a new interpretable space of attributes. These new attributes are used by an ensemble model composed of different automatic learning algorithms classifying the cracks into their most common types: mesh or alligator cracks, longitudinal cracks, and transverse cracks. According to the experiment results, the proposed system can work in computer systems with limited resources and could be used with real-time constraints. Also, the proposed methodology provides more accurate results compared to the existing proposals in the scientific literature. These features enable the system to be placed in non-specialized vehicles collecting and classifying the defects, in the same place where they occur, and simplifying the tasks carried out by experts.

LISTA DE CONTENIDOS

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ACRÓNIMOS

ANN Artificial Neural Networks.

BFTree Best-First tree.

CCR Concrete Crack Regions.

CIE Comission Internationale de l´Eclairage.

cm Centímetros.

CNN Convolutional Neural Networks.

CPU Central Processing Unit.

CSI Camera Serial Interface.

DL Description Length.

ELM Extreme Learning Machine.

FN Falses negatives.

FP Falses positives.

FPS Frames per seconds.

GB GigaByte.

GHz Gigahercio.

GPU Graphics processing unit.

HSV Hue, Saturation, Value

KNN k nearest neighbor.

LiDAR Light Detection And Ranging.

LMT Logistic model trees.

LPDDR Low-Power Double Data Rate Synchronous Dynamic Random Access Memory.

MIPI Mobile Industry Processor Interface.

MLC Maximum Likelihood Classification.

MP Métrica de poda.

MPS Minimal Path Selection.

NMF Nonnegative Matrix Factorization.

NN Neural Networks.

PCA Principal Components Analysis.

PCrCs Potential Crack Components.

ReLU Rectified Linear Unit.

RGB Red, Green, Blue

RIPPER Repeated Incremental Pruning to Produce Error Reduction.

s Segundos.

SM Shape Metric.

SoC System on Chip.

SURF Speeded Up Robust Features.

SVD Singular value decomposition.

SVM Support Vector Machine.

TN True negatives.

TP True positives.

UAV Unmanned Aerial Vehicle.

USB Universal Serial Bus.

1 INTRODUCCIÓN

Tu problema puede ser modesto, pero si desafía tu curiosidad y te lleva a jugar con tus capacidades inventivas, y si lo resuelves por tus propios medios, vas a experimentar la tensión y a disfrutar del triunfo del descubrimiento.
George Pólya [1]

ÍNDICE

1 George Pólya (1887 - 1985) matemático de origen húngaro que dedicó gran parte de su carrera profesional a encontrar métodos que permitiesen enseñar la metodología para resolver problemas. Entre estas metodologías cabe destacar sus avances en enfoques heurísticos, razonamiento inductivo y lógica inductiva entre otros.

1.1 MOTIVACIÓN

Hoy en día la población tiene numerosos medios de transporte terrestre para poder desplazarse dentro de los núcleos urbanos y entre localidades. En ambos casos, para realizar una comunicación efectiva entre zonas se suele requerir de caminos y rutas que estén adecuadamente dotadas para el uso de los diferentes medios de transportes. Un ejemplo de este tipo de elementos lo encontramos en las carreteras. De hecho en lo referente a este tipo de vías, sólo en España [64] en el año 2018, se contaban con 165.624 kilómetros de carreteras. ¿Puede imaginar cuantos millones de kilómetros de este elemento urbanístico se encuentran a lo largo del mundo?

Nuevamente, en el caso de España, al igual que en muchas zonas del resto del mundo, la cifra de vehículos que cruzan en condiciones "normales" (*siempre que no haya normativas que impidan la circulación como estados de alarma o riesgo de contaminación ambiental*) estos elementos urbanísticos es descomunal [65]. Y es que según la organización mundial de la salud, en el año 2016 el número de vehículos legalmente registrados en circulación a lo largo de todo el mundo era superior a 2,1 billones. Con tantos vehículos haciendo uso de las carreteras si las condiciones del pavimento no son óptimas, esto podría suponer un riesgo para los conductores y los pasajeros. En concreto, el mal estado de los pavimentos provoca un efecto sobre la aceleración y la estabilidad de los vehículos [20]. Y tal y como puede observarse en las conclusiones del estudio llevado a cabo por *Lee et a.* [99], la tasa de accidentes de tráfico es mucho mayor en aquellos casos en los que el pavimento esta en malas condiciones. Además, en un futuro, con la implantación de los vehículos autónomos [57] el estado óptimo de las carreteras cobrará una gran importancia para evitar accidentes inesperados en la toma de decisiones de estos vehículos.

Por tanto, el mantenimiento de las carreteras es fundamental en la seguridad [82], pero en términos económicos el impacto también es considerable. En general, el mantenimiento de las carreteras es un problema latente a nivel mundial debido a que los recursos económicos de los que disponen los países para realizar el mantenimiento, son limitados [14, 141]. Recordemos el interrogante del primer párrafo de esta sección que hacía referencia a la cantidad de kilómetros de carreteras en el mundo, ¿se imagina llevar a cabo una inspección visual de tantos tramos de vía por uno o diferentes expertos clasificando y anotando cada uno de los defecto de forma manual?. Esto sería costoso en términos temporales y en términos económicos. No obstante nada más alejado de la realidad, existen estudios actuales [144] que demuestran, que la adquisición de los datos (*ej. imágenes de grietas*) se realiza de forma automática empleando en muchos casos vehículos especializados [105], incurriendo en un coste de adquisición de dichos vehículos. Sin embargo, las tareas importantes de detección de los defectos y la clasificación de los mismos, permanecen en el 99.6 % de los casos como una tarea manual realizada por personas [144] en muchos países.

Además, el coste del mantenimiento, no recae sólo en la inspección visual, sino que existen diferentes tipos de defectos, tales como deformaciones visco-plásticas que forman por ejemplo baches, grietas y desgaste de la superficie, en las que todos ellos tienen un tratamiento de reparación que es diferente al igual que su coste. De entre todos este tipo de defectos, las grietas han cobrado un interés especial por las agencias de transporte y por los investigadores [88, 119, 175, 196], siendo las grietas longitudinales, transversales y en forma de malla o de piel de cocodrilo (*véase un ejemplo de estos tipos en la Figura 1.1*), las más comunes [61]. ¿Pero cuáles son las causas que las provocan y sus medidas de reparación?:

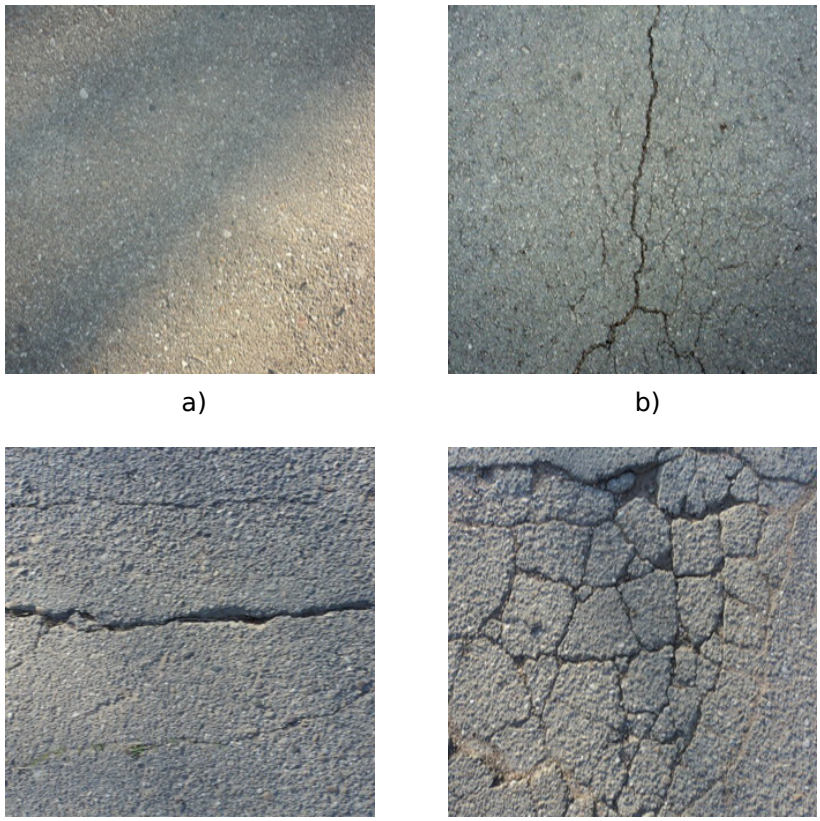


Figura 1.1
Diferentes tipos de grietas en la superficie de la carretera. a) Pavimento sano sin grietas. b) Grieta transversal. c) Grieta longitudinal. d) Grieta de tipo malla.

- *Grietas transversales*: Estas grietas son perpendiculares al eje de abscisas de la imagen (*si tomamos la imagen como referencia espacial*) y suelen estar causadas por cambios térmicos [158], por desprendimientos del terreno, el endurecimiento del aglutinante asfáltico [107] o las reflexiones provocadas por otras grietas bajo la superficie del asfalto.

- *Grietas longitudinales*: Estas grietas serían paralelas al eje de abscisas de la imagen y pueden estar causadas por la fatiga del asfalto

debido a una continua sobrecarga por los vehículos que circulan sobre ella [114], o a un área menos densa del compuesto asfáltico que se encuentra en las uniones entre pavimentos.

- *Grietas de tipo malla*: Este tipo de grietas, también conocidas como *piel de cocodrilo* (*debido a su similitud en la textura con la piel de este animal*), son el efecto de la fatiga del asfalto y de una base inestable del mismo [127]. Esta inestabilidad está provocada cuando las capas inferiores del asfalto no pueden soportar las capas superficiales, que a su vez están derivadas de un drenaje ineficiente así como el efecto de temperaturas extremas como heladas y congelamiento del asfalto [33, 59]. En este tipo de grietas puede ocurrir que se pierdan partes del pavimento si no se tratan a tiempo y darían lugar a baches y un deterioro progresivo de la superficie del asfalto.

En cuanto a las medidas para llevar a cabo la reparación de las grietas transversales y longitudinales, se distinguen dos tipos, cuando las grietas no son anchas y cuando sí lo son. Cuando la anchura de la grieta no es excesiva, la reparación consiste en sellar la mismas para evitar una entrada de humedad y agentes externos que sigan aumentando su tamaño [44], dejando el asfalto sin fisuras. Por el contrario para las grietas que son bastante pronunciadas y anchas, la solución consiste en sustituir la capa de asfalto completamente. Además, en problemas de desprendimiento del terreno, la solución empleada tiene que tener en cuenta la topología del mismo y llevar una reparación integral de toda la zona. En las grietas de tipo malla, debido a su naturaleza, la solución general suele ser reemplazar toda la superficie afectada con una nueva capa de asfalto.

En este contexto, la visión por computador [147, 174] junto con las técnicas de aprendizaje automático de la inteligencia artificial [146, 201], pueden ser un mecanismo útil en la recolección eficiente de las imágenes de las carreteras y su posterior manipulación para la clasificación en los diferentes tipos de defectos anteriormente detallados. En realidad, estas disciplinas, ya han sido de utilidad en otros campos de la ingeniería civil para la monitorización e inspección de estructuras [145], permitiendo automatizar tareas que requerían un gran esfuerzo y consumo de ejecución.

Por ello, se plantea como hipótesis que el uso de métodos de visión por computador junto con técnicas de aprendizaje automático permitirían obtener una herramienta automática y autónoma que podría emplearse en sistemas de recursos limitados y baja capacidad de cómputo para su utilización masiva en vehículos, evitando así un sobre-coste al requerido por los medios de reparación, o la necesidad de emplear vehículos especializados [105] para esta labor. Esto permitiría extraer de una forma eficaz e invariante en la medida de lo posible a la textura del pavimento, los defectos en caso de que existan y clasificarlos en los diferentes tipos mostrados en la Figura 1.1 de forma inmediata, sustituyendo o asistiendo, cuando sea posible la labor llevada a cabo por los expertos de forma manual.

1.2 OBJETIVOS

Una vez se ha establecido la motivación principal, esta Tesis Doctoral aborda la creación de un ***"sistema automático para la detección y clasificación de grietas en pavimentos"***. Para el desarrollo de este sistema, se establecen los siguiente objetivos:

- Realizar un sistema que permita realizar un procesamiento de las imágenes para extraer las características relevantes de los defectos.
- Realizar una clasificación de los diferentes defectos de la banda viaria en sus tipos más comunes.

Para alcanzar ambos objetivos, se plantean los siguientes objetivos concretos derivados de los dos anteriores:

- Para el primer objetivo centrado en realizar el procesamiento de las imágenes, se distinguen los siguientes puntos:
  - Implementación de métodos basados en visión por computador que permitan realizar un análisis y extracción de características de defectos en pavimentos, cuando los haya.
  - Permitir que las implementaciones de los métodos de procesamiento de imágenes sean explotables y puedan trabajar en hardware de recursos de cómputo limitados y de forma inmediata.
  - Implementación de métodos de reducción de datos para el análisis de defectos en pavimentos que permitan minimizar los datos empleados para la generación y uso de modelos de clasificación.
- Del mismo modo que sucede en el objetivo anterior y en concordancia con los resultados obtenidos de los objetivos derivados del mismo, se pueden definir los siguientes puntos para el proceso de la clasificación:
  - Clasificación de los diferentes tipos de defectos en carreteras o pavimentos alquitranados, con especial hincapié en las grietas y en sus tipos más comunes, los cuales corresponden con grietas longitudinales, transversales y grietas de tipo malla (*véase un ejemplo de estos tipos de grietas de forma visual en la Figura 1.1*).
  - Utilización de algoritmos de aprendizaje supervisado orientados a la clasificación que permitan aumentar la tolerancia a fallos en el proceso de clasificación de defectos.
  - Generación modelos de clasificación que sean interpretables y que sean ejecutables en dispositivos hardware de recursos limitados.
  - Análisis de la utilización de metaclasificadores para aprovechar sus ventajas en la toma de decisión de la clasificación.

1.3 ESTRUCTURA DE CAPÍTULOS

Debido a la motivación y a los objetivos explicados en las secciones anteriores que persigue la presente Tesis Doctoral, este documento se divide y organiza en los siguientes capítulos, como se muestra en la Figura 1.2:

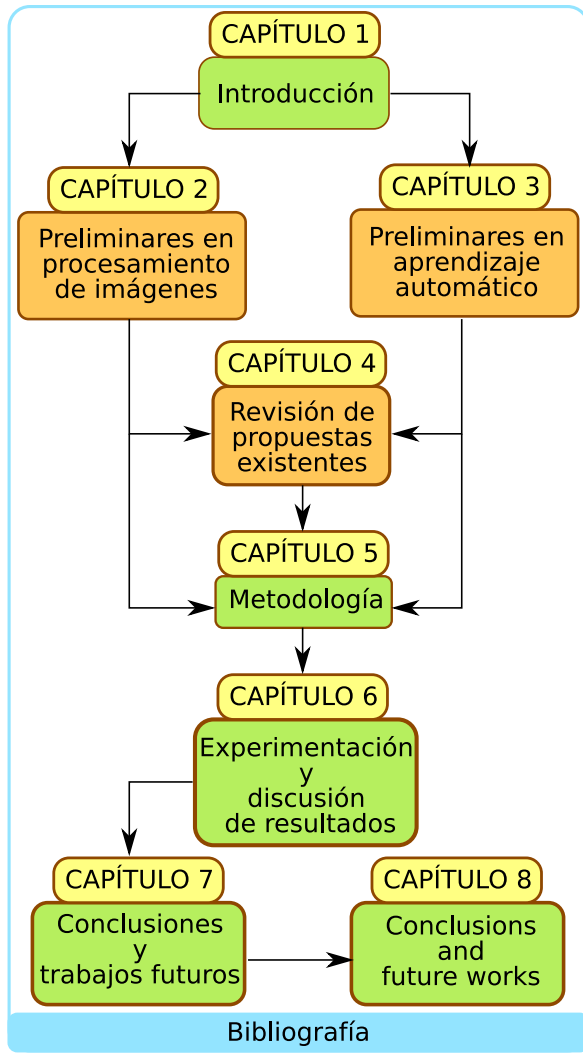


Figura 1.2
Esquema general de organización de los capítulos de esta tesis.

El Capítulo "1: *Introducción*" ha proporcionado la motivación y los objetivos que pretende alcanzar la presente Tesis Doctoral así como la sección actual para entender cómo se organiza el presente documento.

El Capítulo "2: *Preliminares en procesamiento de imágenes*" provee un enfoque general y descontextualizado de la temática de la tesis, de los diferentes métodos y algoritmos de procesamiento de imágenes que pueden ser empleados para tratar con estos elementos. Entre estos algoritmos, se destacan métodos para conocer cómo se representan las

imágenes, cómo mejorar la visualización de las imágenes, y cómo extraer información derivada de las mismas de forma local y global.

El Capítulo "3: *Preliminares en aprendizaje automático*" al igual que sucede en el Capítulo 2 se vuelve a descontextualizar de la temática de la tesis, proporcionando así una visión general de los diferentes tipos de algoritmos de aprendizaje automático que existen. Además, los algoritmos que han sido empleados por otros autores en la temática propia de la clasificación de grietas en pavimentos también quedan en su mayoría cubiertos en este capítulo, así como los empleados en la propuesta de esta Tesis Doctoral. Además, este capítulo proporciona información de las métricas para el análisis de rendimiento de los modelos de clasificación, y las diferentes metodologías del diseño de experimentos con los datos.

En el Capítulo "4: *Revisión de propuestas existentes*" se reanuda la temática de la Tesis Doctoral explicada en el Capítulo 1 y se categorizan las propuestas de los diferentes autores dependiendo de su objetivo, proporcionando una revisión y descripción de los enfoques que utilizan. Este capítulo se nutre de los conceptos explicados en los capítulos 2 y 3.

El Capítulo "5: *Metodología*" detalla las ideas principales para alcanzar los objetivos impuestos en el Capítulo 1 contemplando los métodos disponibles que existen para el procesamiento de imágenes y de aprendizaje automático de los capítulos 2 y 3 respectivamente. Además, se tienen en cuenta en la proposición de la metodología, aquellas limitaciones o vías no analizadas de la literatura científica existente, que se pueden encontrar resumidas al final del Capítulo 4.

El Capítulo "6: *Experimentación y discusión de los resultados*" presenta la experimentación realizada así como los principales resultados de la aplicación de las propuestas del Capítulo 5 en imágenes reales de calzadas alquitranadas. Además, este capítulo proporciona una comparación con las propuestas presentadas en el Capítulo 4 que más afinidad comparten con los objetivos de la presente Tesis Doctoral.

El Capítulo "7: *Conclusiones y trabajos futuros*" presenta las principales conclusiones obtenidas del desarrollo y los resultados obtenidos de los capítulos 5 y 6. Además, este capítulo presenta las próximas líneas de investigación que pueden derivar de la presente Tesis Doctoral, susceptibles de ser analizadas en futuros trabajos.

El Capítulo "8: *Conclusions and future works*" corresponde con una traducción en inglés del contenido del Capítulo 7, siguiendo los requisitos de la normativa impuesta en el *RD 99/2011* referente a la memoria de la Tesis Doctoral, para poder optar a la mención de *Doctor Internacional*.

Por último se encuentra el listado de toda la bibliografía a la que se hace referencia a lo largo de toda la memoria, proporcionando la información necesaria para que cualquier interesado pueda encontrarlas y consultarlas.

2

PRELIMINARES EN PROCESAMIENTO DE IMÁGENES

Si queremos que las máquinas piensen, tenemos que enseñarles a ver
Fei-Fei Li [1]

ÍNDICE

1 Fei-Fei Li (1975 - Actualidad) es profesora en el departamento de ciencias de la computación en la Universidad de Stanford y codirectora del instituto de Inteligencia Artificial centrado en el ser humano (HAI). Durante el periodo comprendido entre los años 2013 a 2018 fue directora del laboratorio de inteligencia artificial (SAIL) de Stanford.

2.1 INTRODUCCIÓN

En este capítulo se muestran todos los mecanismos previos referentes al procesamiento de imágenes, que el lector del presente documento necesita conocer para entender completamente la propuesta de esta tesis. Pero antes de exponer con detalle los diferentes métodos, detengámonos a recordar un momento, muy probablemente la siguiente frase nos resulte familiar: *"Una imagen vale más que mil palabras"*. Si bien mucha gente conoce dicha frase, es difícil encontrar el origen de la misma en palabras de algún personaje célebre, dado que la frase desde sus orígenes ha ido mutando hasta la que conocemos hoy en día. De hecho, en muchas ocasiones se atribuye su origen a Napoleón Bonaparte o incluso a Leonardo Da Vinci. Esta frase puede considerarse uno de los principales pilares de varios de los objetivos que persigue del procesamiento de imágenes.

El procesamiento de imágenes se puede considerar como el conjunto de métodos para obtener información de las imágenes o mejorar la información que obtenemos de las mismas. En realidad, el procesamiento de imágenes tiene muchos campos de aplicación tales como sistemas de seguridad [36, 49], sistemas de extracción de información [73, 183], sistemas de ayuda a la toma de decisión en ambientes clínicos [131, 149], sistemas de conducción automática [26, 35], sistemas de monitorización de estructuras civiles [55, 174], y casi una infinidad de ejemplos todos ellos con un nexo en común y evidente, las imágenes.

TÁNGARA CARIRROJA: *Ave de la familia "Cardinalidae" que habita principalmente en América del norte*

Todos conocemos qué son las imágenes, de hecho podemos observar la ilustración de un pájaro que corresponde con una TÁNGARA CARIRROJA macho en la Figura 2.1[182], y nuestro cerebro es capaz de interpretar y comprender la información visual que representa dicha imagen. Sin embargo esto nos lleva al siguiente interrogante, ¿qué es para una computadora una imagen?

PIXEL: *Procedente de"picture element" en inglés, es la unidad de información básica que componen las imágenes en un computador. En español se utiliza el mismo término acentuado "Píxel"*

Para una computadora, una imagen no es más que un conjunto de bits almacenados en memoria. En concreto nuestro colorido amigo de la Figura 2.1 está representado con una imagen en color (*en las siguientes secciones se matiza cómo se representa el color*) de 8-bits por cada canal de color. Esto significa que la información de cada PIXEL de la imagen, está representado con un valor entre [0 y 255]. Este rango de valores representa la profundidad de la imagen, que se puede entender como la precisión con la que un píxel es capaz de representar la información. Evidentemente, las imágenes no están limitadas a 8-bits, sino que en la práctica se emplean profundidades mayores y menores dependiendo tanto de los sensores utilizados para captar las imágenes, así como de las necesidades del problema que se desea tratar.

Por lo tanto, los píxeles componen las imágenes, y estas últimas suelen ser representadas internamente en una computadora por dos estructu-

Figura 2.1
Imagen de Tángara Carirroja (*Piranga ludoviciana*) macho.

ras a la hora de procesarlas [67]: representación vectorial (I_{Vc} o I_{Vf}), y representación matricial (I_m).

$$I_m = \begin{bmatrix} P_{0,0} & P_{0,1} & P_{0,2} & \dots & P_{0,c-1} \\ P_{1,0} & P_{1,1} & P_{1,2} & \dots & P_{1,c-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ P_{f-1,0} & P_{f-1,1} & P_{f-1,2} & \dots & P_{f-1,c-1} \end{bmatrix} \tag{2.1}$$

$$I_{Vc} = \begin{bmatrix} P_{0,0} \\ P_{1,0} \\ \vdots \\ P_{f-1,0} \\ P_{0,1} \\ \vdots \\ P_{f-1,c-1} \end{bmatrix} \qquad I_{Vf} = \begin{bmatrix} P_{0,0} \\ P_{0,1} \\ \vdots \\ P_{0,c-1} \\ P_{1,0} \\ \vdots \\ P_{f-1,c-1} \end{bmatrix} \tag{2.2}$$

La principal diferencia entre la representación matricial (I_m) y la representación vectorial (I_{Vc} o I_{Vf}), es que en la primera los datos, en este caso los píxeles ($P_{i,j}$), suelen estar organizados en una estructura multidimensional como se muestra en la Expresión 2.1 donde el tamaño está delimitado por el número de filas (f) y columnas (c) para una imagen en dos dimensiones (*2D*). Y en la representación vectorial los píxeles se almacenan en una estructura unidimensional (*1D*) de dos formas distintas, representando las columnas primero (I_{Vc}) o almacenado las filas primero (I_{Vf}) como puede observarse en la Expresión 2.2. En la presente Tesis Doctoral no se emplearán imágenes de más de dos dimensiones (*2D*), pero sepa el lector que esto depende del campo de aplicación y que, por ejemplo, en el campo sanitario lo más usual es

utilizar imágenes de tres dimensiones ($3D$).

Hasta este punto, se conoce qué es y qué estructuras representan una imagen en una computadora, ¿pero los colores que pueden componer la imagen de la Figura 2.1 u otra cualquiera son siempre los mismos en una computadora?, ¿Se puede extraer información de las imágenes más allá de nuestra percepción visual?, ¿Es posible mejorar la calidad de las imágenes? En los siguientes apartados se detallan diferentes métodos de procesamiento de imágenes que proporcionan la respuesta a estos interrogantes. No obstante, las siguientes secciones no pretenden ni mucho menos cubrir todo el abanico posible de métodos de procesamiento de imágenes, sino presentar y explicar los conceptos y mecanismos fundamentales necesarios para que el lector pueda comprender en su totalidad la presente Tesis Doctoral.

2.2 REPRESENTACIONES DEL COLOR

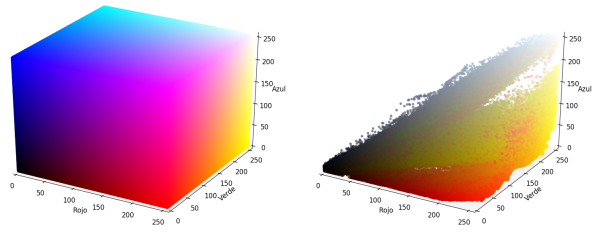
La representación del color se refiere a qué espacios, modelos de colores o gama de colores van a representar las imágenes en una computadora, y en definitiva que valores van a tener cada uno de los píxeles que la componen. Si bien la imagen de la tángara carirroja de la Figura 2.1 es una de las representaciones más comunes en imágenes (*en concreto una representación en* RGB), una misma imagen puede representarse de diferentes formas [71, 84]. Algunas de estas representaciones pueden resultar interpretables visualmente y otras que bajo ojos no expertos darían lugar a imaginar que la cámara, el sensor que ha tomado las fotografías, o el propio monitor que muestra las imágenes está defectuoso. Sin embargo, esta variedad de representaciones, lejos de ser una cuestión de gustos, es una cuestión de necesidades, que dependiendo del problema a tratar pueden simplificar el procesamiento requerido por los diferentes métodos. Por ello, las secciones siguientes muestran algunos de los modelos y espacios de colores más habituales, y cómo obtener sus representaciones desde la que se considera el estándar (*el modelo RGB*):

RGB: *Del inglés "Red (R", "Green (G)", "Blue (B)"*

2.2.1 *Modelo RGB*

El modelo de color RGB [173] está compuesto por tres dimensiones o canales que corresponden con los colores primarios: Rojo, verde y azul. Este modelo consigue representar el espectro de colores mediante la adición de cada una de las dimensiones detalladas anteriormente. El modelo normalizado, suele ser representado gráficamente mediante un cubo en el que cada uno de los ejes es una dimensión de color que toma valores en el rango $[0-1]$. Dicho cubo tiene su origen en el color *negro* con los valores de mínima intensidad $(0,0,0)$, y en la arista opuesta el color *blanco* con los valores de máxima intensidad $(1,1,1)$. En la Figura 2.2 a) se muestra un ejemplo de este cubo completo en un rango de intensidades de $[0-255]$ que es el usado en el caso de las imágenes de 8-bits. La Figura 2.2 b) muestra el mismo rango de valores del cubo pero incluyendo únicamente los píxeles que componen la tángara carirroja de la Figura 2.1. En realidad, la Figura 2.1 que

nuestros ojos eran capaces de observar con colores vivos y realistas, no es más que la adición de los valores de los tres canales como se puede observar en la Figura 2.3 a), c), e). Como la visualización de los tonos en un canal monocromático cuesta observarse, se ha facilitado en la misma figura (b), d) y f)) la representación de las intensidades de cada canal en escalas de grises (*se describirá en la siguiente sección*).



a) b)

Figura 2.2
Cubos de modelo de color RGB. a) Cubo completo; b) Píxeles correspondientes a la Figura 2.1.

a) b)

c) d)

e) f)

Figura 2.3
Separación de imagen RGB en componentes. a) Canal R; b) Escalas de grises del canal R; c) Canal G; d) Escalas de grises del canal G; e) Canal B; f) Escalas de grises del canal B.

2.2.2 *Representación en escala de grises*

La representación en escalas de grises consiste en la visualización y representación monocromática de una imagen. A diferencia del modelo de color RGB, esta representación muestra las intensidades (*denominadas en ocasiones como brillo*) en un único canal [109]. En la Figura 2.3 ya se observó un ejemplo en el que las intensidades de cada canal del modelo RGB se representaron en escala de grises en el rango $[0-255]$. Sin embargo, una imagen en escala de grises no está limitada a representar un único canal, recordemos que una imagen en RGB consistía en un modelo aditivo para representar la imagen final, por lo que es posible convertir una imagen en RGB a escalas de grises conservando esta componente aditiva [89] tal y como muestra la Ecuación 2.3:

$$P_{GS}(i,j) = 0.21 \cdot P_{C_R}(i,j) + 0.72 \cdot P_{C_G}(i,j) + 0.07 \cdot P_{C_B}(i,j) \quad (2.3)$$

Donde:

- $P_{GS}(i,j)$: Píxel en escala de grises.
- $P_{C_R}(i,j)$: Canal rojo de un píxel de una imagen RGB.
- $P_{C_G}(i,j)$: Canal verde de un píxel de una imagen RGB.
- $P_{C_B}(i,j)$: Canal azul de un píxel de una imagen RGB.

La Figura 2.4 muestra el resultado de aplicar la ecuación anterior a la imagen de la Figura 2.1 en la que cada píxel se encuentra representado únicamente por un canal en el rango $[0-255]$. Puede observarse claramente las diferencias que existen entre la representación en escala de grises de cada uno de los canales individuales mostrados en la Figura 2.3, donde dependiendo del canal la percepción visual de los valores del ave eran diferentes entre cada canal y en la Figura 2.4 el resultado es más cercano a la imagen en color completa.

Figura 2.4
Representación en escala de grises imagen de Tángara Carirroja.

2.2.3 *Espacio de color HSV*

El espacio de color HSV [1] al igual que el modelo RGB se representa con tres componentes, HUE, SATURATION, VALUE. No obstante, aunque tenga tres componentes, el significado de cada una de ellas difiere del modelo RGB y es más cercano a cómo los humanos percibimos los colores, describiendo cada componente el matiz o tono del color, en términos de su pureza y luminosidad. A diferencia del modelo RGB, este espacio de color no se suele representar como un cubo, sino como una figura cónica invertida en la que:

HUE: *En español "Matiz"*
SATURATION: *En español "Saturación"*
VALUE: *En español "Valor"*

- H: Representa en tono del color y corresponde con un ángulo en una de las circunferencias que forman el cono de este espacio de color. Sus valores corresponden con un ángulo en el rango $[0° - 360°]$, tomando los colores primarios rojo, verde y azul del modelo RGB, los valores [0° y 360°], [120°] y [240°] respectivamente.
- S: Representa la saturación o pureza del color, y corresponde con el radio de la circunferencia de la componente anterior (H). El valor de la saturación está comprendido en un rango de $[0 - 1]$ correspondiendo el valor máximo a los valores de color puro y a medida que se decrementa dicho valor el color tiene mezclados otros tonos.
- V: Representa el valor o luminosidad. Esta componente también se puede encontrar con el nombre de BRIGHTNESS (B) y corresponde con el eje central del cono, el cual proporciona el valor de intensidad en la luminosidad del color en el rango $[0 - 255]$.

BRIGHTNESS: *En español "Brillo"*

Las componentes H, S y V de este espacio se pueden obtener desde las imágenes en el espacio RGB mediante las Ecuación 2.5, Ecuación 2.6 y Ecuación 2.7 respectivamente. Nótese que el cálculo de cada componente se realiza para cada píxel (i, j), y que dicha notación se ha omitido para simplificar la lectura manteniéndose así en las sucesivas secciones hasta que se especifique lo contrario.

$$\begin{aligned} M &= max(P_{C_R}, P_{C_G}, P_{C_B}) \\ m &= min(P_{C_R}, P_{C_G}, P_{C_B}) \end{aligned} \tag{2.4}$$

$$H = \begin{cases} 60 \cdot \frac{P_{C_G} - P_{C_B}}{M-m} & \text{si } M = P_{C_R} \\ 120 + 60 \cdot \frac{P_{C_B} - P_{C_R}}{M-m} & \text{si } M = P_{C_G} \\ 240 + 60 \cdot \frac{P_{C_R} - P_{C_B}}{M-m} & \text{si } M = P_{C_B} \end{cases} \tag{2.5}$$

$$S = \begin{cases} \frac{M-m}{M} & \text{si } M \neq 0 \\ 0 & \text{En caso contrario} \end{cases} \tag{2.6}$$

$$V = M \tag{2.7}$$

Donde la mayoría de ecuaciones comparten los siguientes parámetros:

M: Máximo valor del píxel RGB obtenido con la Ecuación 2.4.

m: Mínimo valor del píxel RGB obtenido con la Ecuación 2.4.

P_{C_R}: Canal rojo de un píxel de una imagen RGB.

P_{C_G}: Canal verde de un píxel de una imagen RGB.

P_{C_B}: Canal azul de un píxel de una imagen RGB.

H: Componente de tono para un píxel.

S: Componente de saturación para un píxel.

V: Componente de brillo para un píxel.

En la Figura 2.5 se puede observar la separación de los diferentes canales de HSV y su representación en escalas de grises de la tángara carirroja que se ha usado en las anteriores secciones. Se puede observar que para el canal V, la representación en escala de gris es exactamente igual que los valores del canal debido a que representan la misma información.

Figura 2.5
Separación de imagen HSV en componentes. a) Canal Matiz (H); b) Representación en escalas de grises del canal Matiz (H); c) Canal Saturación (S); d) Representación en escalas de grises del canal Saturación (S); e) Canal Valor (V); f) Representación en escalas de grises del canal Valor (V).

2.2.4 *Espacio de color YCbCr*

El espacio de color YCbCr [21] es una codificación no lineal del modelo RGB que se basa en la premisa de representar los colores teniendo en cuenta la alta capacidad que tiene el ojo humano en cuanto detectar variaciones de luminosidad separando esta última en una componente independiente (Y) de los colores y representando la CROMINANCIA en el resto de componentes (Cb y Cr). Este espacio de color se puede considerar como una versión escalada y desplazada de otro espacio de color llamado YUV [78]. En la práctica este espacio de color es usado para la transmisión de señales digitales de vídeo [80] (*a diferencia de YUV que está pensado para transmisión analógica*) debido a que permite representar los colores con menos cantidad de información que el formato RGB. Al igual que el modelo de color RGB y HSV este espacio de color está definido tal y como se ha comentado antes, por tres componentes:

CROMINANCIA: *Información respecto al color.*

- Y: Representa la componente de luminancia o luma que corresponde con un canal monocromático compuesto por los valores de intensidad del brillo de la imagen tomando valores comprendidos en el rango $[16-235]$.
- Cb: Representa la crominancia azul que se define como la diferencia entre las tonalidades azules y un valor de referencia como puede ser la componente de luminancia. Esta componente toma valores en el rango $[16-240]$.
- Cr: Representa la crominancia roja que se define como la diferencia entre las tonalidades rojas y un valor de referencia como puede ser la componente de luminancia. Esta componente toma los mismos rangos de valores que la crominancia azul $[16-240]$.

Dada una imagen en formato RGB, la transformación de los canales R, G y B a las componentes Y, Cb, Cr se llevan a cabo mediante la Ecuación 2.8, Ecuación 2.9 y Ecuación 2.10 para cada canal respectivamente.

$$Y = 16 + \frac{65.783 \cdot P_{C_R}}{256} + \frac{129.057 \cdot P_{C_G}}{256} + \frac{25.064 \cdot P_{C_B}}{256} \tag{2.8}$$

$$C_b = 128 - \frac{37.945 \cdot P_{C_R}}{256} - \frac{74.494 \cdot P_{C_G}}{256} + \frac{112.439 \cdot P_{C_B}}{256} \tag{2.9}$$

$$C_r = 128 + \frac{112.439 \cdot P_{C_R}}{256} - \frac{94.154 \cdot P_{C_G}}{256} - \frac{18.285 \cdot P_{C_B}}{256} \tag{2.10}$$

Donde:

P_{C_R}: Canal rojo de un píxel de una imagen RGB.

P_{C_G}: Canal verde de un píxel de una imagen RGB.

P_{C_B}: Canal azul de un píxel de una imagen RGB.

Y: Componente luma del píxel.

C_b: Componente cromática azul del píxel.

C_r: Componente cromática roja del píxel.

En la Figura 2.6 se puede observar cada uno de los canales del espacio de color YCbCr y su representación en escalas de grises, para la imagen de la tángara carirroja que hemos usado en las secciones anteriores.

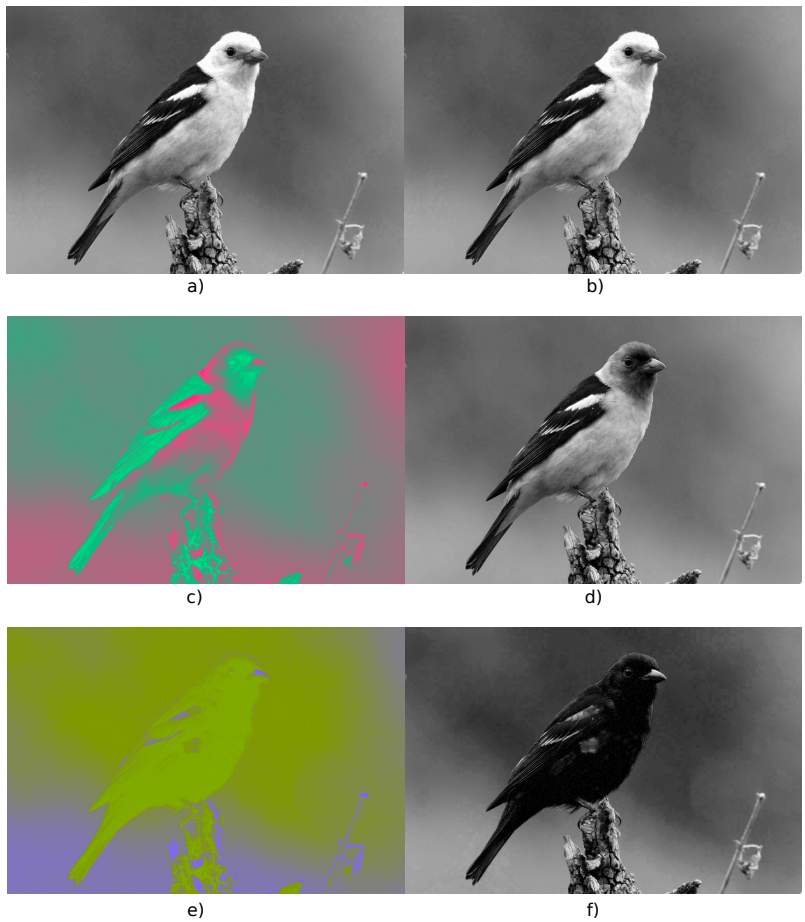


Figura 2.6
Separación de imagen YCbCr en componentes. a) Componente Luma (Y); b) Representación en escalas de grises del canal Luma (Y); c) Canal crominancia rojo (Cr); d) Representación en escalas de grises del canal crominancia rojo (Cr); e) Canal crominancia azul (Cb); f) Representación en escalas de grises del canal crominancia azul (Cb).

2.2.5 *Espacio de color CIE L*a*b**

El espacio de color CIE L*a*b* [38], también denominado CIELAB o simplemente Lab, se desarrolló como un intento de representar todos los colores que es capaz de visualizar el ojo humano. Este espacio de color al igual que los anteriormente descritos (*excepto la representación en escala de grises*) está formado por tres componentes. Como el objetivo de este espacio de color es ser capaz de representar millones de colores imitando al ojo humano, la relación entre cada una de las componentes es logarítmica de forma similar a la respuesta del ojo ante los colores a diferencia del modelo de color RGB, el cual era un modelo lineal. En realidad a diferencia de la representación en forma de cubo que tenía el modelo RGB, en este caso la representación suele ser una esfera cromática en la que cada una de las componentes del modelo son ortogonales entre sí y representan lo siguiente:

- L: Representa la componente de luminosidad la cual toma valores en el rango $[0 - 100]$ donde el valor mínimo representa el negro y el opuesto el blanco. Esta componente es la que se encuentra como un eje vertical en mitad de la esfera que forma el modelo.
- a: Esta componente se considera oponente, en el sentido que representa el grado de rojo y verde que contiene un color, considerando que ambos colores no pueden aparecer al mismo tiempo. Esta componente tiene dos extremos, un extremo positivo (+a) que representa el rojo y un extremo negativo que representa el verde (-a) y aunque en un principio el modelo no considera límites numéricos en esta componente, en la práctica se suelen emplear el rango [-128 a 127].
- b: Al igual que la componente anterior, también representan dos colores oponentes, el azul (-b) y el amarillo (+b). Esta componente se encuentra representada de forma ortogonal a las dos anteriores en la esfera, y al igual que la componente anterior tampoco tiene límites, pero se suele utilizar el mismo rango de la componente a.

Del mismo modo que en los modelos anteriores, es posible obtener una transformación desde los valores de una imagen RGB y obtener una imagen en el espacio de color Lab, pero esta conversión no es tan directa como en los casos anteriores. El primer paso para obtener las componentes L, a, b de este modelo es convertir los valores RGB al espacio de color imaginario CIE XYZ [47, 78] mediante las Ecuación 2.11, Ecuación 2.12 y Ecuación 2.13 respectivamente para cada componente X, Y, Z de este último modelo:

$$X = \frac{0.433910 \cdot P_{C_R}}{255} + \frac{0.376220 \cdot P_{C_G}}{255} + \frac{0.189860 \cdot P_{C_B}}{255} \tag{2.11}$$

$$Y = \frac{0.212649 \cdot P_{C_R}}{255} + \frac{0.715169 \cdot P_{C_G}}{255} + \frac{0.072182 \cdot P_{C_B}}{255} \tag{2.12}$$

$$Z = \frac{0.017756 \cdot P_{C_R}}{255} + \frac{0.109478 \cdot P_{C_G}}{255} + \frac{0.872915 \cdot P_{C_B}}{255} \tag{2.13}$$

Donde:

P_{C_R}: Canal rojo de un píxel de una imagen RGB.

P_{C_G}: Canal verde de un píxel de una imagen RGB.

P_{C_B}: Canal azul de un píxel de una imagen RGB.

X: Componente X del modelo CIE XYZ.

Y: Componente Y del modelo CIE XYZ.

Z: Componente Z del modelo CIE XYZ.

Una vez obtenida la representación de un píxel RGB en el modelo CIE XYZ, es posible obtner las componentes del modelo CIE LAB empleando la Ecuación 2.15, Ecuación 2.16 y Ecuación 2.17 respectivamente para cada componente [47]:

$$f(n) = \begin{cases} n^{\frac{1}{3}} & \text{si } n > 0.008856 \\ (7.787 \cdot n) + \frac{16}{116} & n \leqslant 0.008856 \end{cases} \tag{2.14}$$

$$L = \begin{cases} 116 \cdot Y^{\frac{1}{3}} & \text{si } Y > 0.008856 \\ 903.3 \cdot Y & \text{si } Y \leqslant 0.008856 \end{cases} \tag{2.15}$$

$$a = 500 \cdot (f(X) - f(Y)) \tag{2.16}$$

$$b = 200 \cdot (f(X) - f(Y)) \tag{2.17}$$

Donde:

L: Componente de luminosidad del modelo CIE LAB para un píxel.

a: Componente oponente rojo-verde del modelo CIE LAB para un píxel.

b: Componente oponente azul-amarillo del modelo CIE LAB para un píxel.

X: Componente X del modelo CIE XYZ.

Y: Componente Y del modelo CIE XYZ.

$f(X)$: Función para el cálculo de las componentes oponentes del modelo CIE LAB haciendo uso de la componente X del modelo CIE XYZ empleando la Ecuación 2.14.

$f(Y)$: Función para el cálculo de las componentes oponentes del modelo CIE LAB haciendo uso de la componente Y del modelo CIE XYZ empleando la Ecuación 2.14.

En la Figura 2.7 puede observarse el resultado de la separación de los canales del modelo CIE LAB para la imagen de la Figura 2.1. Además, en la parte derecha de la Figura, se puede apreciar la representación en escala de grises de los valores de cada uno de los canales.

Figura 2.7
Separación de imagen Lab en componentes. a) Canal L; b) Representación en escalas de grises del canal L; c) Canal a; d) Representación en escalas de grises del canal a; e) Canal b; f) Representación en escalas de grises del canal b.

2.3 MÉTODOS DE SEGMENTACIÓN BASADOS EN UMBRALES

SEGMENTACIÓN MEDIANTE UMBRALES: *En ingles "Thresholding"*

Uno de los mecanismos más básicos del procesamiento de imágenes es la SEGMENTACIÓN MEDIANTE UMBRALES de valores [159]. Este mecanismo permite realizar las siguientes tareas entre otras:

- Separar partes de una imagen.
- Eliminar el fondo de una imagen.
- Modificar los valores de una imagen de forma global truncándolos a un valor dado.

Existen diferentes enfoques para llevar a cabo las tareas anteriormente citadas, y todas ellas comparten un elemento en común, el uso de un valor de umbral. Este valor de umbral (th) se puede determinar manualmente o mediante métodos o algoritmos que lo determine automáticamente como veremos en las siguientes secciones. Previamente al comienzo de los diferentes mecanismos de segmentación basado en valores, hay que indicar que por simplicidad en la visualización de los resultados, se va a utilizar la imagen monocromática en escala de grises de la Figura 2.4. Sin embargo, estos métodos no están limitados a funcionar en imágenes de un único color sino que se pueden emplear en imágenes de cualquier naturaleza. Por ejemplo, en una imagen en el espacio de color RGB, tendríamos dos opciones, usar el mismo umbral para cada canal o emplear umbrales diferentes para cada uno de los canales.

2.3.1 *Segmentación con umbral binaria*

El primer enfoque consiste en generar una imagen binaria o bitonal en la que sólo existen dos valores posibles para cada píxel utilizando un valor de umbral. De este modo, todos los píxeles de la imagen que sean iguales o superiores a dicho umbral se ponen a un valor máximo establecido, y aquellos que no, a un valor mínimo, que generalmente suele ser igual a 0. La expresión 2.18 muestra el cálculo para cada píxel de la imagen.

$$UB_{x,y} = \begin{cases} maxVal & \text{si } P_{x,y} > th \\ minVal & \text{En caso contrario} \end{cases} \tag{2.18}$$

Donde:

$P_{x,y}$: Píxel de la imagen para ser analizado.

th: Umbral de corte para segmentar la imagen.

$maxVal$: Valor máxima de la segmentación.

$minVal$: Valor mínimo de la segmentación.

$UB_{x,y}$: Nuevo píxel umbralizado.

La Figura 2.8 muestra un ejemplo de la segmentación de la Figura 2.4 en el que se ha establecido como umbral el valor 80 y como valores mínimo y máximos 0 y 255 respectivamente.

a) b)

Figura 2.8
Resultado de segmentación binaria. a) Imagen Tángara carirroja en escala de grises. b) Resultado de la segmentación binaria.

2.3.2 *Segmentación con umbral truncada*

Este enfoque de segmentación es similar al anterior y consiste en modificar los píxeles de una imagen de forma global mediante un umbral. En este caso, no existen un valor máximo ni mínimo al que establecer los píxeles, sino que todos los valores que sean igual o superior que el umbral definido, se establecen al valor de dicho umbral y los que no, mantienen su valor original tal y como se detalla en la Expresión 2.19.

$$UT_{x,y} = \begin{cases} th & \text{si } P_{x,y} > th \\ P_{x,y} & \text{En caso contrario} \end{cases} \tag{2.19}$$

Donde:

$P_{x,y}$: Píxel de la imagen para ser analizado.

th: Umbral de corte para segmentar la imagen.

$UT_{x,y}$: Nuevo píxel truncado.

En la Figura 2.9 se puede observar el efecto de realizar la segmentación truncada con un valor de umbral de 80, el mismo que en la segmentación binaria.

a) b)

Figura 2.9
Resultado de aplicar una segmentación truncada. a) Imagen Tángara carirroja en escala de grises. b) Resultado de segmentación truncada.

2.3.3 *Segmentación con umbral adaptativa*

En la segmentación mediante un umbral adaptativo, a diferencia de los dos métodos detallados anteriormente, el valor del umbral se calcula automáticamente en función los píxeles VECINOS de uno dado. Además, en este caso las reglas de decisión al analizar el píxel no están definidas, siendo posible emplear el comportamiento de la segmentación binaria o de la segmentación truncada dependiendo del problema que se pretenda solventar. Por ejemplo, en el caso de la segmentación binaria, se llevaría a cabo con la Expresión 2.20, donde el nuevo parámetro $th_{x,y}$ corresponde con el valor del umbral calculado dinámicamente.

VECINOS: *píxeles adyacentes*

$$UA_{x,y} = \begin{cases} maxVal & \text{si } P_{x,y} > th_{x,y} \\ minVal & \text{En caso contrario} \end{cases} \tag{2.20}$$

Existen dos formas de realizar el cálculo del umbral adaptativo $th_{x,y}$ (*ambas formas se explicarán en detalle en la Sección 2.4*):

- Cálculo del umbral basado en la media: En este caso cada píxel $P_{x,y}$ es evaluado con la media de él mismo y de todos su vecinos dado un tamaño de vecindad K. El resultado se puede observar en la Figura 2.10 a) para un tamaño de k igual a 13.
- Cálculo del umbral basado en distribución gaussiana: En este caso cada píxel $P_{x,y}$ es evaluado con la importancia del valor del propio píxel y de sus vecinos dado un tamaño de vecindad k y basándose en la forma de una distribución gaussiana centrada en el píxel analizado. El resultado se puede observar en la Figura 2.10 b) para un tamaño de k igual a 13.

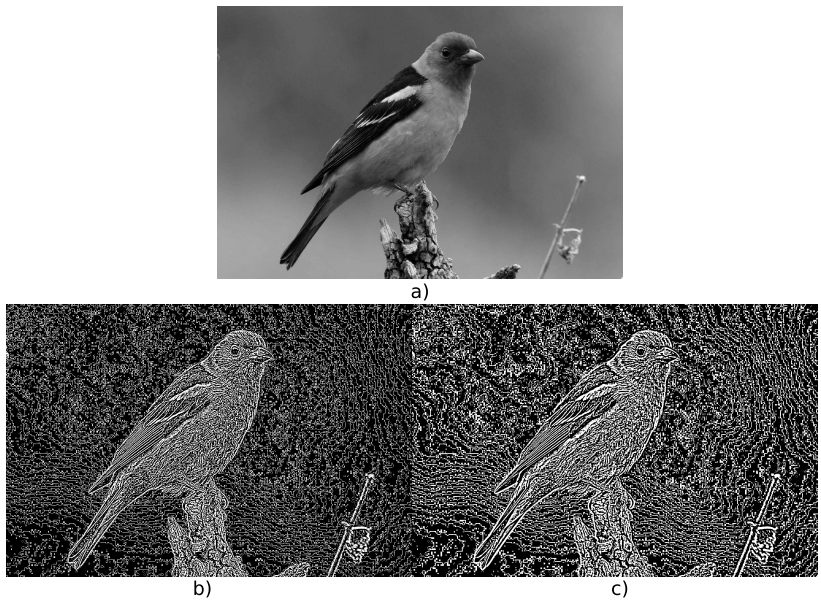


Figura 2.10
Resultado de aplicar una Segmentación binaria adaptativa. a) Imagen Tángara carirroja en escala de grises. b) Resultado aplicando la media de vecinos. c) Resultado aplicando la distribución gaussiana.

2.3.4 *Método de Otsu*

El método de Otsu [200] se centra en el cálculo de un umbral global th de forma automática a diferencia de la segmentación adaptativa en la que el umbral era local a cada píxel. Considerando la imagen en escala de grises de la Figura 2.4 donde los valores de la imagen quedan comprendidos en el rango $[0, L-1]$ (*siendo* $L-1 = 255$ *el valor máximo en escala de grises para una imagen de 8-bit*), el método de Otsu divide los píxeles de la imagen en dos clases $C_0 = [0, th]$ y $C_1 = [th+1, L-1]$. Por lo tanto el umbral th queda definido como el valor que minimiza la varianza entre clases calculándose con la Ecuación 2.21.

$$\sigma_t^2 = \sum_{i=0}^{t} P_i \cdot \sum_{i=t+1}^{L-1} P_i \cdot \left(\frac{\sum_{i=0}^{t} iP_i}{\sum_{i=0}^{L-1} P_i} - \frac{\sum_{i=t+1}^{L-1} iP_i}{1 - \sum_{i=0}^{L-1} P_i} \right)^2 \tag{2.21}$$

Donde:

σ_t^2: Es la varianza entre las clases C_0 y C_1.

P_i: Es la probabilidad de cada valor de gris en la imagen en escala de grises.

$L-1$: Máximo valor en escala de grises.

t: Valor que separa los píxeles en las clases C_0 y C_1.

Al igual que sucedía con la segmentación adaptativa, a centrase en el cálculo automático de un umbral, las reglas de decisión al analizar un píxel no están definidas pudiéndose aplicar una segmentación binaria o truncada. De esta forma, la Figura 2.11 muestra el resultado de una segmentación binaria tomando 115 como el valor del umbral calculado por el método de Otsu.

Figura 2.11
Resultado de aplicar el método de Otsu para la segmentación binaria. a) Imagen Tángara carirroja en escala de grises. b) Resultado de aplicar el umbral calculado por el método de Otsu.

2.4 MÉTODOS DE MEJORA DE LA IMAGEN

RUIDO: *En imágenes el ruido se entiende como pequeñas perturbaciones esporádicas que aparecen a lo largo de la imagen*

¿Alguna vez ha sacado o visto una fotografía en la que había RUIDO? ¿O alguna vez ha intentado sacar una fotografía de un documento en el que la iluminación hacía que se proyectasen sombras que ocultasen parcialmente su contenido? Los interrogantes anteriores son problemas usuales en el procesamiento de imágenes y se mitigan usando los denominados métodos de mejora. Los métodos de mejora se deben de entender como un conjunto de herramientas que nos permiten analizar y modificar el contenido de las imágenes para proporcionar una versión más adecuada de las mismas. Cómo los interrogantes anteriores sólo son dos ínfimos ejemplos de los problemas que pueden ocurrir, estos métodos se emplean en diferentes escenarios y con diversos objetivos entre los que se pueden destacar los siguientes:

- Reducción de ruido.
- Mejora del contraste o la iluminación.
- Enfatizar detalles o incluso eliminar detalles.
- Comprimir la imagen.

Como existen muchos tipos de métodos, el presente documento sólo describirá algunos, centrándose en aquellos métodos basados en la reducción del ruido de la imagen (*ej. difuminado, gaussiano, bilateral*) y en uno de los métodos utilizados para mejorar la iluminación en las imágenes (*Transformada Logarítmica*).

Por lo general, los métodos de mejora basados en reducción de ruido se suelen llamar *filtros* debido a que basan su funcionamiento en pequeñas matrices aplicadas a nivel local que actúan a modo de máscara y dependiendo de los valores que la componen, se calcula el nuevo valor para un píxel en la imagen mejorada tal y como se muestra en la Figura 2.12. La modificación de dichos filtros es la que define el objetivo del algoritmo y el tamaño del filtro la densidad de píxeles que abarca.

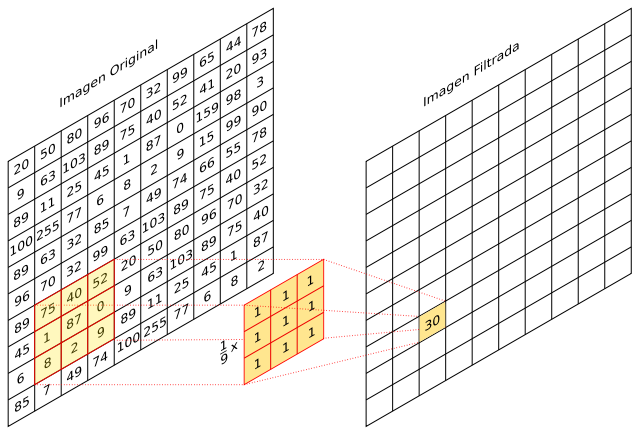


Figura 2.12
Esquema general de convolución y filtrado.

2.4.1 *Filtro de difuminado*

FILTRO DE DIFUMINADO: *En inglés "Blur filter"*

El FILTRO DE DIFUMINADO [28], también conocido como *Tfiltro de la media* o *Tfiltro de emborronamiento*, se utiliza para reducir el ruido de una imagen de forma global teniendo en cuenta la importancia del valor de los vecinos del píxel. De esta forma, en el filtro de difuminado todos los vecinos tienen el mismo peso en el valor final del filtrado correspondiendo esta metodología a la media aritmética de un píxel y todos sus adyacentes. El cálculo de esta media al depender del tamaño de la vecindad (N), se puede expresar con una matriz similar a la que se mostraba en la Figura 2.12. Esta figura mostraba una matriz considerando los vecinos inmediatos a un píxel (K_{3x3}) con un valor de N igual a 3. Sin embargo, este filtro se puede generalizar para tamaños de vecindad mayores tal y como muestra la Expresión 2.22, teniendo en cuenta que el número de filas y de columnas debe de ser impar.

$$K_{NxN} = \frac{1}{N \cdot N} \left[\begin{array}{ccc|c} 1 & \dots & 1 & \\ \vdots & \vdots & \vdots & N \\ 1 & \dots & 1 & \\ \hline & N & & \end{array}\right] \tag{2.22}$$

Estos filtros se pueden emplear tanto en imágenes monocromáticas como en imágenes en color teniendo en cuenta cada una de sus componentes por separado a la hora de aplicar el filtro K_{NxN}. El resultado de este filtrado con una matriz de vecindad K_{9x9} aplicada a la Tángara carirroja de la Figura 2.1 se puede observar en la Figura 2.13, junto a pequeñas regiones aumentadas para apreciar con detalle el efecto del filtro.

Figura 2.13
Ejemplo de filtrado de difuminado y detalles.

2.4.2 *Filtro Gaussiano*

El filtro gaussiano [197] tiene un funcionamiento similar al filtro de difuminado, pero deja de ser un método lineal. Como su propio nombre indica en este caso la matriz de vecindad K_{NxN} se construye a través de una función de distribución gaussiana bidimensional atendiendo a la Ecuación 2.23, donde σ corresponde con la desviación típica. La representación de la función de distribución gaussiana para un valor de σ igual a 3 se puede apreciar en la Figura 2.14 a). De este modo, los vecinos toman una importancia diferente en el resultado del filtro, teniendo más peso aquellos centrados en el píxel evaluado.

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.23}$$

Dado que la función de distribución gaussiana toma valores continuos, los valores que componen la matriz de vecindad K_{NxN}, se suele aproximar mediante el Triángulo de Pascal [85] el cual se puede apreciar desarrollado hasta la profundidad 6 del triángulo en la Figura 2.14 b).

Triángulo de Pascal: *Triángulo de números en el que cada fila corresponde con el desarrollo de las potencias del binomio de Newton*

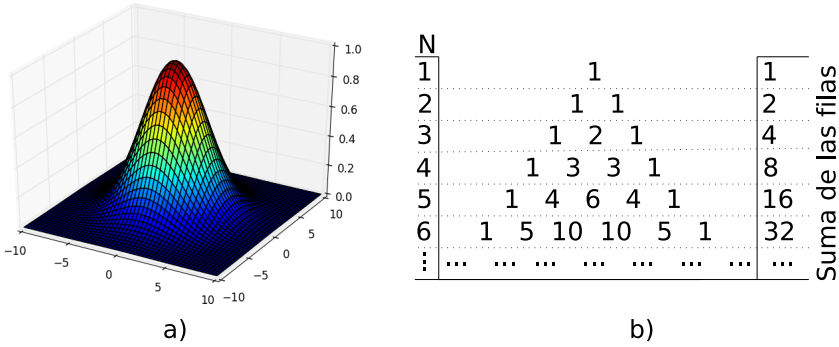


Figura 2.14
Relación triángulo de Pascal y kernel gaussiano.

Teniendo en cuenta lo anterior, una matriz de vecindad K_{NxN} se formaría como una multiplicación entre matrices, en la que el primer multiplicando es una matriz columna compuesta por una fila del triángulo, multiplicada a su vez por una constante equivalente a uno entre la suma de los elementos de dicha matriz columna, y la segunda parte de la multiplicación corresponde con una matriz formada por filas del triángulo de Pascal repetidas N, y de nuevo multiplicada dicha matriz por la misma constante que en el caso anterior. En la Expresión 2.24 se puede observar el cálculo para un ejemplo de matriz de vecindad K_{3x3}.

$$K_{3x3} = \frac{1}{4} \cdot \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix} \cdot \frac{1}{4} = \frac{1}{16} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.24}$$

La Figura 2.15 muestra el resultado de aplicar un filtro gaussiano de tamaño K_{9x9} a la Figura 2.1, además se han aumentado algunos detalles para que sea posible una comparación visual con el resultado del

filtro de difuminado anterior. Nótese que en este caso pese a ser ambos filtros del mismo tamaño, en el filtrado gaussiano, hay zonas que se emborronan menos y mantienen algunos de los detalles a diferencia del filtro de difuminado (*véase la Figura 2.13*).

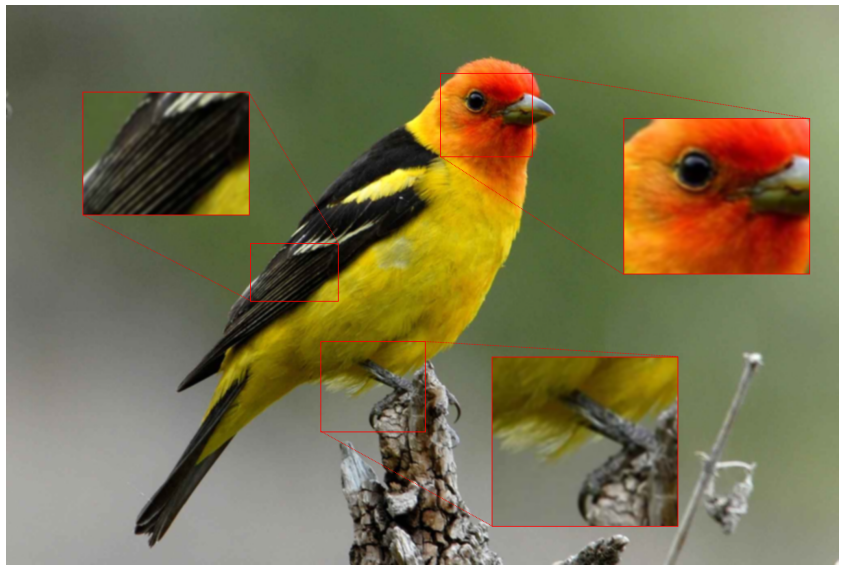


Figura 2.15
Ejemplo de filtrado gaussiano y detalles.

2.4.3 *Filtro Bilateral*

El filtro bilateral [180] se trata de un método no lineal al igual que ocurría con el filtrado gaussiano detallado en la sección anterior. En este caso el objetivo de dicho filtro es realizar un suavizado o difuminado de la imagen respetando los bordes (*este concepto se explicará en la Sección 2.5*). Además, el filtro bilateral comparte aspectos en común con el filtrado gaussiano, ya que emplea la misma fórmula de la distribución gaussiana pero en una dimensión tal y como se muestra en la Ecuación 2.25:

$$F(I) = \frac{\sum_{w \in K} G_{\sigma_k}(\|p - w\|) G_{\sigma_r}(I_p - I_w) I_w}{\sum_{w \in K} G_{\sigma_k}(\|p - w\|) G_{\sigma_r}(I_p - I_w)} \tag{2.25}$$

Donde:

I: Corresponde con la imagen que va a ser filtrada.

F(I): Corresponde con la imagen resultante del filtrado bilateral.

K: Corresponde con la matriz de vecindad.

w: Posición de un píxel dentro de la matriz de vecindad.

p: Posición actual del píxel de la imagen I que se desea filtrar.

I_p: Valor del píxel que se va a filtrar (p).

I_w: Valor del píxel que está dentro de la matriz de vecindad w.

$||p - w||$: Distancia euclídea entre las posiciones de los píxeles p y w.

G_{σ_K}: Parámetro de control de emborronamiento espacial calculado mediante la Ecuación 2.26.

G_{σ_r}: Parámetro de control de rango de amplitud mínimo de los bordes calculado mediante la Ecuación 2.26.

$$G_\sigma(x) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2}{2\sigma^2}} \tag{2.26}$$

De forma análoga a los dos filtros anteriores la Figura 2.16 muestra el resultado del filtrado bilateral, donde se ha vuelto a utilizar la Figura 2.1 resaltando en ella los mismos detalles que en los casos anteriores así como el mismo tamaño de la matriz de vecindad K_{9x9}. Se puede apreciar en resultado cómo los detalles del ala, las plumas y el cerco de los ojos, en este filtrado permanecen visibles a diferencia de lo que ocurría en la Figura 2.15 para el filtrado gaussiano y en la Figura 2.13 para el filtro de difuminado.

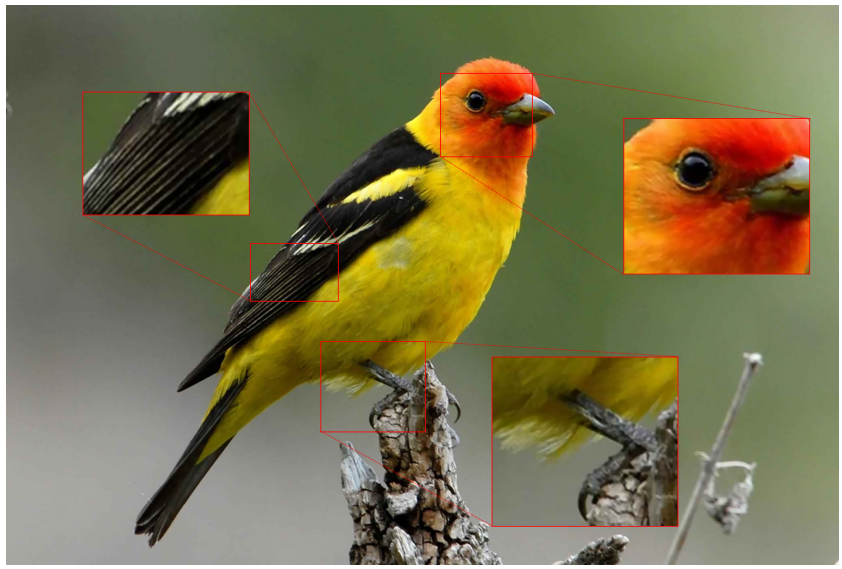


Figura 2.16
Ejemplo de filtrado bilateral y detalles.

2.4.4 *Transformada logarítmica*

La transformada logarítmica [45, 134, 156] es un método que queda fuera de los mecanismos de reducción de ruido anteriormente explicados y se centra en realizar una rectificación a nivel global de la iluminación de la imagen con el objetivo de modificar los tonos claros y oscuros (*niveles de intensidad*). Al estar centrado en niveles de intensidad, lo habitual es aplicarlo a imágenes en escalas de grises. Sin embargo, la transformada logarítmica no se puede aplicar directamente, sino que es necesario convertir la imagen en su opuesta mediante el NEGATIVO. Una vez calculado el negativo de la imagen y tal y como el propio nombre del método indica, se calcula el logaritmo que permite que el comportamiento del rango dinámico de la iluminación deje de ser lineal y pase a ser logarítmico como se detalla la Ecuación 2.27.

NEGATIVO: *Opuesto de una imagen. Por ejemplo en escala de grises sería* $Negativo = 255 - (Valor\_de\_gris)$ *(con imágenes de 8-bits de profundidad)*

$$LT(i,j) = \alpha \cdot Klog(N(i,j)) + \beta \cdot (log(N(i,j)) - Klog(N(i,j))) \quad (2.27)$$

Donde:

$LT(i,j)$: Píxel resultante de aplicar la transformada logarítmica.

$N(i,j)$: Corresponde con el negativo de un píxel en escala de grises $(255 - P_{GS}(i,j))$.

$log(N(i,j))$: Logaritmo en base diez del negativo de un píxel de la imagen en escala de grises.

$Klog(N(i,j))$: Logaritmo en base diez del resultado del promedio de los vecinos de un píxel en la posición (i,j) empleando un tamaño de vecindad de K_{3x3} idéntico al filtrado de difuminado.

α: Parámetro que configura el sesgo de brillo. Valores cercanos a 0 más peso en tonos oscuros, valores próximos a 1 más peso en tonos blancos.

β: Parámetro que configura el sesgo de nitidez de la imagen.

En la Figura 2.17 se puede observar el efecto de la transformada logarítmica en el que se han utilizado un valor de $\alpha = 0.8$ y $\beta = 1$ consiguiendo con dichos valores potenciar los tonos claros y mantener la nitidez de la imagen igual que la original.

a) b)

Figura 2.17
Ejemplo de transformada logarítmica. a) Imagen en escala de grises. b) Resultado de la transformada logarítmica.

2.5 EXTRACCIÓN DE BORDES

DETECCIÓN DE BORDES: *En inglés "Edge Detection"*

Otro de los problemas más comunes en el procesamiento de imágenes es la extracción o DETECCIÓN DE BORDES [22, 135, 204]. Pero antes de comenzar con los diferentes métodos de procesamiento, es necesario conocer que es un borde y cuál es la metodología para detectar dichas características.

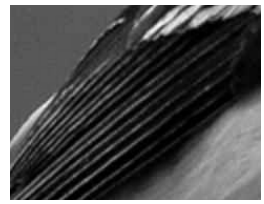


Figura 2.18
Detalles del ala de la tángara carirroja en escala de grises.

En lo relativo al procesamiento de imágenes, un borde se define como un cambio local significativo en la intensidad de la imagen como se puede observar en la Figura 2.18 correspondiente al ala de la tángara carirroja, extraída de la imagen en escala de grises de la Figura 2.4. Estas diferencias entre intensidades se pueden deber a muchos motivos entre ellos:

- Límite de profundidad de los objetos.
- Cambio de colores en los objetos.
- Sombras entre los objetos.
- Discontinuidad de la texturas de las superficies.

El objetivo de obtener los bordes es conseguir detectar los límites de los objetos, así como los pequeños segmentos de píxeles que están unidos, y a partir de ellos poder extraer información de la imagen para calcular por ejemplo las líneas, o el número de circunferencias que existen en la imagen. Por otro lado, además de extraer información relativa a los limites de los objetos, la detección de bordes permite eliminar gran parte de la información de la imagen manteniendo las características fundamentales. Pero ¿cómo se lleva a cabo este proceso? y ¿qué métodos existen para realizar la extracción?.

En general los métodos de detección de bordes se pueden dividir en tres categorías [118], métodos basados en derivadas de primer orden (*ej. método de robert, sobel, prewitt*), métodos basados en derivadas de segundo orden (*ej. método laplaciano*) y métodos basados en criterios de optimización (*ej. método de Canny*).

Los métodos de derivada de primer orden, suelen estar basados en el cálculo del operador del gradiente descendiente de una función, donde el gradiente representa la dirección de máxima variación en un punto. De este modo, los puntos con un valor mayor en el gradiente corresponderían con los bordes. Para el cálculo del gradiente así como su

magnitud y ángulo se emplean la Ecuación 2.28, Ecuación 2.29 y Ecuación 2.30 respectivamente.

$$\nabla f(x,y) = \left[\frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y}\right] \tag{2.28}$$

$$M|\nabla| = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2} \tag{2.29}$$

$$\omega = arctang\left(\frac{\frac{\partial f(x,y)}{\partial y}}{\frac{\partial f(x,y)}{\partial x}}\right) \tag{2.30}$$

Donde:

$\nabla f(x,y)$: Vector del gradiente descendente para una función bidimensional.

M: Magnitud del vector $\nabla f(x,y)$.

ω: Ángulo del vector $\nabla f(x,y)$.

Los métodos de derivada de segundo orden, suelen cuantificar los cambios en una función empleando para ello la segunda derivada de dicha función. En este caso al calcular la segunda derivada mediante la Ecuación 2.31, un borde queda representado por aquellos puntos donde la derivada cruza en el valor 0.

$$\nabla^2 f(x,y) = \left[\frac{\partial^2 f(x,y)}{\partial x^2}, \frac{\partial^2 f(x,y)}{\partial y^2}\right] \tag{2.31}$$

Finalmente, los métodos con criterios de optimización suelen estar basados en los dos anteriores junto con una serie de directrices y herramientas para descartar o considerar que píxeles se consideran bordes en una imagen.

Hay que destacar tal y como se ha comentado anteriormente, que los bordes se consideran un cambio abrupto de intensidad, y por ello las imágenes que se utilizan para el cálculo de los mismos, suelen ser imágenes monocromáticas que representen la intensidad en un único color como la escala de grises o algunos de los canales independientes explicados en la Sección 2.2. Sin embargo, esto no quiere decir que siempre se empleen imágenes monocromáticas, dado que existen estudios [50, 92, 203] de métodos de detección de bordes aplicados directamente en imágenes en color que no serán cubiertos en la presente Tesis Doctoral.

De este modo, las siguientes secciones detallan algunos de los métodos más usuales de la detección de bordes en imágenes monocromáticas empleando como ejemplo la imagen en escala de grises de la Figura 2.4.

2.5.1 *Método de Robert*

El método de Robert [74, 163] se enmarca dentro del grupo de algoritmos de derivada de primer orden, y hace uso del gradiente descendente. Sin embargo, la Ecuación 2.28 que calcula el gradiente de una función, se puede aproximar mediante diferencias finitas tal y como muestra la Ecuación 2.32, donde los términos i y j representan coordenadas de la imagen en términos de filas y columnas.

$$\nabla f(i,j) = \sqrt{f(i,j)} - \sqrt{f(i+1,j+1)}, \sqrt{f(i+1,j)} - \sqrt{f(i,j+1)} \quad (2.32)$$

Estas diferencias finitas, se pueden representar mediante un pequeña matriz de convolución con un funcionamiento similar al detallado en la sección 2.4. En este caso el operador de Robert se representa con un filtro K_x para la derivada a lo largo del eje "X", y un filtro K_y para la derivada a lo largo del eje "Y" tal y como muestra la Expresión 2.33. La magnitud del gradiente se calcula mediante una aproximación de la Ecuación 2.29 en el que la raíz cuadrada se sustituye por la suma de los términos de la misma en valor absoluto evitando así la carga computacional en la operación.

$$K_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad K_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad (2.33)$$

En la Figura 2.19 se proporciona el resultado de la magnitud del gradiente (*a)*) que corresponde con las derivadas en los ejes "X" e "Y" junto con el cálculo del gradiente para cada eje (*b)* y *c)*). Además se observa que los bordes detectados son prácticamente inapreciables.

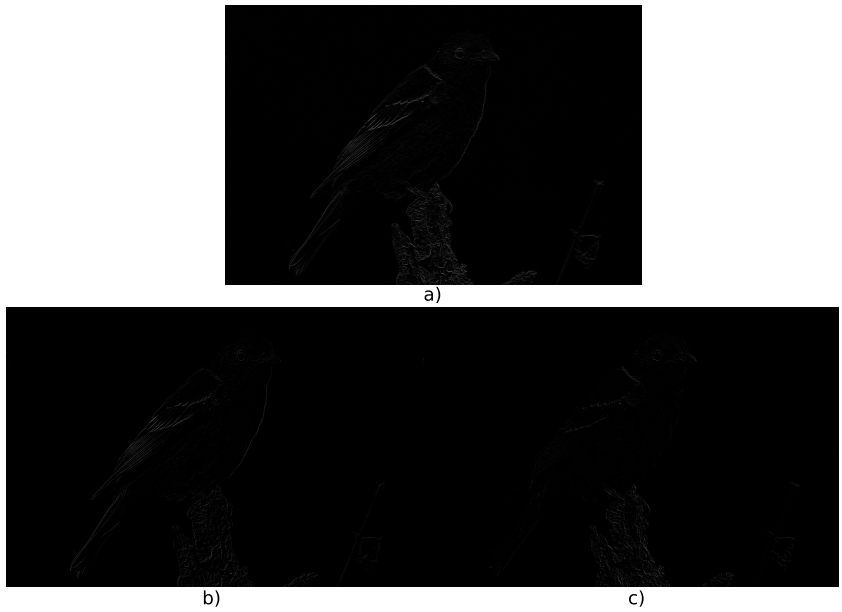


Figura 2.19
Ejemplo de aplicación del método de Robert. a) Método de Robert en eje X e Y. b) Método de Robert en el eje X. c) Método de Robert en el eje Y.

2.5.2 *Método de Sobel*

El método de Sobel [95, 171] al igual que el método anterior se trata de un enfoque que hace uso de las derivadas de primer orden para poder obtener de esta forma los bordes de una imagen. En este método se vuelve a realizar una aproximación de las derivadas parciales para cada eje, pero a diferencia del método de Robert, la aproximación queda expresada por la Ecuación 2.34, donde los términos i y j representan coordenadas de la imagen en términos de filas y columnas.

$$\begin{aligned} \frac{\partial f(x,y)}{\partial x} &= f(i+1,j) - \cdot f(i-1,j) \\ \frac{\partial f(x,y)}{\partial y} &= f(i,j+1) - \cdot f(i,j-1) \end{aligned} \tag{2.34}$$

Al igual que ocurría en el método de Robert, la diferencia finita se puede representar en formal de filtros para obtener los bordes en el sentido del eje "X" (K_x) e "Y"(K_y) como se muestra en la Expresión 2.35.

$$K_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad K_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{2.35}$$

En la Figura 2.19 se puede observar el resultado de la magnitud del gradiente (*a)*) que corresponde con la suma de ambos filtros en valor absoluto contemplando ambos ejes, junto con el cálculo del gradiente para cada eje por separado (*b)* y *c)*).

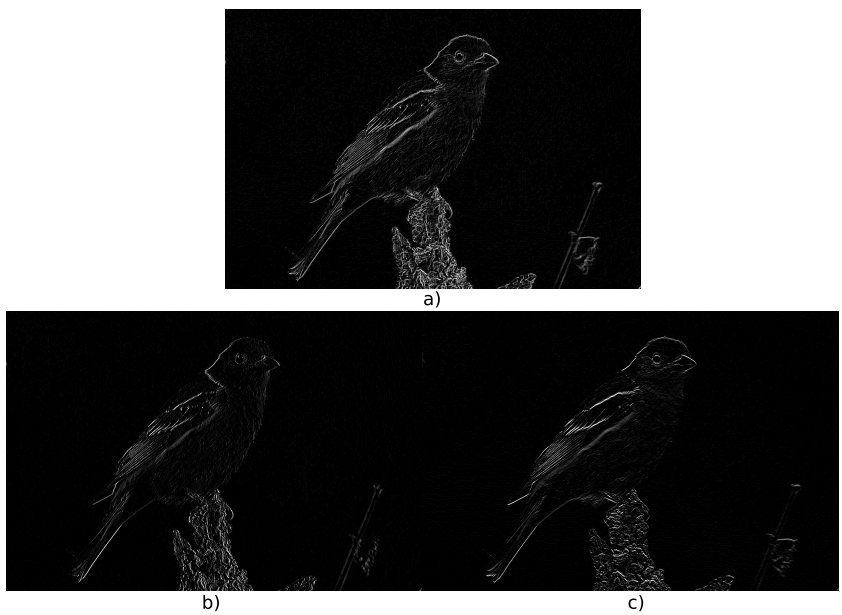


Figura 2.20
Ejemplo de aplicación del método de Sobel. a) Método de Sobel en eje X e Y. b) Método de Sobel en el eje X. c) Método de Sobel en el eje Y.

2.5.3 *Método de Prewitt*

El método de Prewitt [126, 191] es el último método de gradiente del primer orden que se va a explicar en la presente Tesis Doctoral, y comparte muchas similitudes con el método de Sobel anteriormente explicado. El cálculo de la magnitud y el gradiente se realiza de la misma forma que en Sobel, pero la aproximación de la derivada en forma de matrices difiere respecto a este último. En este caso las matrices que aproximan el cálculo de las derivadas parciales a lo largo del eje "X" (K_x) e "Y" (K_y) están representadas en la Expresión 2.36. Se puede observar la similitud respecto a Sobel ya que el elemento que cambia es la fila central en K_x y la columna central en K_y.

$$K_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \qquad K_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \tag{2.36}$$

En la Figura 2.21 se puede apreciar el resultado de la magnitud del gradiente (*a)*) contemplando ambos ejes, junto con el cálculo del gradiente para cada eje por separado (*b)* y *c)*). Nótese que a diferencia del resultado obtenido por el método de Sobel en la Figura 2.20, existen pequeños bordes de las plumas del ala correspondientes a la región mostrada en la Figura 2.18 que no se detectan por el método.

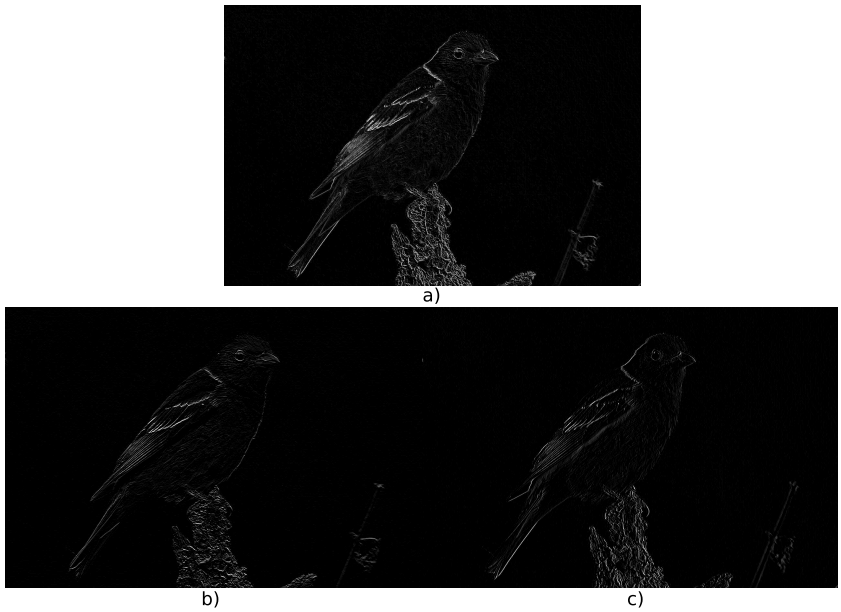


Figura 2.21
Ejemplo de aplicación del método de Prewitt. a) Método de Prewitt en eje X e Y. b) Método de Prewitt en el eje X. c) Método de Prewitt en el eje Y.

2.5.4 *Método Laplaciano*

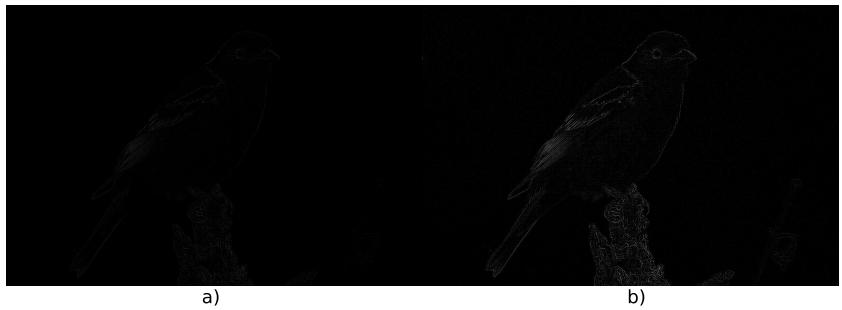
El método laplaciano [117, 187] se categoriza dentro de los operadores que hacen uso de las derivadas de segundo orden tal y como muestra la Ecuación 2.31. Sin embargo, al igual que ocurría con los métodos anteriores, es posible discretizar las derivadas de la ecuación anterior con diferencias finitas en la que cada derivada parcial de segundo orden queda expresada por la Ecuación 2.37.

$$\begin{aligned} \frac{\partial^2 f(x,y)}{\partial x^2} &= f(i+1,j) - 2\cdot f(i,j) + f(i-1,j) \\ \frac{\partial^2 f(x,y)}{\partial y^2} &= f(i,j+1) - 2\cdot f(i,j) + f(i,j-1) \end{aligned} \tag{2.37}$$

En el método laplaciano, las derivadas se pueden representar en forma de matrices de convolución. Sin embargo, a diferencia de los métodos de derivadas de primer orden explicados anteriormente, el método laplaciano no necesita un filtro para recorrer el eje "X" e "Y", sino que con un único filtro se pueden obtener los bordes en ambas direcciones, optimizando así el tiempo de procesamiento necesario para el cálculo de los bordes. Además, el filtro del método laplaciano se puede representar de dos formas distintas, considerando sólo la información de los bordes en sentido horizontal y vertical (K_n) o considerando además la información de los ejes diagonales a los dos anteriores ($K_{diagonal}$) tal y como muestra la Expresión 2.38.

$$K_n = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad K_{diagonal} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.38}$$

En la Figura 2.22 se puede observar el resultado del método laplaciano para la imagen de la Figura 2.4 haciendo uso de los filtros de la Expresión 2.38. Se puede apreciar que en el resultado considerando el filtro horizontal y vertical, los bordes son prácticamente inapreciables en la imagen, siendo éstos un poco más enfatizados en el resultado del filtro diagonal. Sin embargo, se observa que a diferencia de los métodos como el de *Sobel* o *Prewitt*, el método laplaciano, proporciona bordes menos definidos.



a) b)

Figura 2.22
Ejemplo de aplicación del Método Laplaciano. a) Método Laplaciano en eje X e Y. b) Método Laplaciano considerando diagonales.

2.5.5 *Método de Canny*

El método de Canny [30, 133] se engloba dentro de los métodos de detección de bordes basados en criterios de optimización y consiste básicamente en cuatro pasos:

1. *Cálculo del gradiente*: El primer paso consiste en emplear el método de Sobel para realizar el cálculo de la magnitud del gradiente y su dirección.

2. *Supresión de los valores no máximos*: El gradiente obtenido contiene bordes que no tienen un valor máximo proporcionando falsos bordes. Para resolver este problema, se compara cada píxel y sus vecinos en la dirección del gradiente y se toma la siguiente regla de decisión: *Si el píxel que se está analizando tiene el valor máximo de entre todos sus vecinos, se mantiene como borde, en caso contrario, se descarta como borde.*

3. *Segmentación basada en umbrales*: En este paso se descartan todos los píxeles que no correspondan con bordes usando dos umbrales. Un umbral mínimo (th_1) en el que todos los píxeles por debajo del mismo son descartados como bordes, y un umbral máximo (th_2), a partir del cual todos los píxeles serán considerados como bordes. El resto de píxeles se analizan en el siguiente paso.

4. *Proceso de histéresis*: El último paso del método consiste en realizar un seguimiento de cada uno de los bordes de tal forma que si uno de los píxeles candidatos a bordes entre los valores [th_1,th_2] no está conectado a un píxel que se ha considerado como borde en el paso anterior, se considera como un falso borde y no sería detectado como tal.

Tal y como se ha detallado en los pasos, los valores de los umbrales th_1 y th_2 necesitan ajustarse para que el método de Canny proporcione un resultado adecuado, siendo en ocasiones una tarea difícil encontrar los valores de ambos umbrales. Como se muestra en la Figura 2.23 la imagen de la izquierda (*a)*) muestra un ejemplo del método de Canny aplicado a la imagen de la Figura 2.4, en el que ambos umbrales se han establecido a 0, y en la parte derecha (*b)*) en el que se han ajustado ambos umbrales, siendo $th_1 = 19$ y $th_2 = 110$.

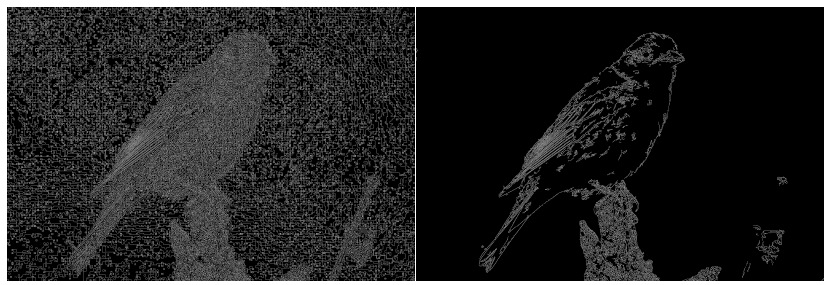


Figura 2.23
Ejemplo de aplicación del método de Canny. a) Método de Canny con umbral mínimo y máximo establecidos a 0. b) Método de Canny con umbral mínimo y máximo ajustados.

2.6 OPERADORES MORFOLÓGICOS

Esta sección aborda los operadores morfológicos que son una serie de modificadores a nivel local de píxeles que nos permiten manipular la forma de los objetos en una imagen. Estos operadores se suelen emplear en imágenes binarias, como las que observamos en los resultados de las segmentaciones basadas en umbrales de la Sección 2.3. No obstante, no se van a aplicar estos operadores a las imágenes de dicha sección debido a su complejidad en cuando a la forma, sino que se va a hacer uso del escudo oficial de la titulación del Grado en Ingeniería Informática [52] de la Universidad de Córdoba al que se le ha aplicado una segmentación basada en el método de Otsu para obtener la imagen binaria que comentábamos anteriormente. Tal y como se puede observar en la Figura 2.24, el escudo tiene una forma simple, compuesta por una corona junto a un núcleo toroidal de ferrita, hilos de lectura y escritura y dos ramas simétricas una de laurel y otra de olivo, que nos ayudará a comprender cada uno de los operadores de las siguientes secciones.

Figura 2.24
Escudo de titulación de Ingeniería Informática de la Universidad de Córdoba. a) Escudo con colores originales. b) Resultado del escudo tras segmentar la imagen.

2.6.1 *Dilatación y Erosión*

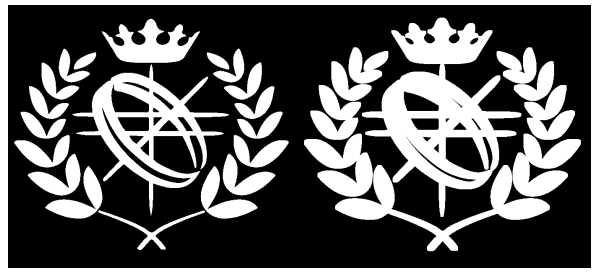
Los dos primeros operadores que se contemplan son los de *dilatación* y *erosión* que son considerados los operadores básicos. El primero de ellos, el operador de dilatación, se centra en expandir la forma de un objeto siguiendo la Ecuación 2.39:

$$\varphi(I) = \{i + s | i \in I \wedge s \in K\} \tag{2.39}$$

Donde:

- $\varphi(I)$: Resultado del operador de dilatación para la imagen I.
- i: Píxel con coordenadas (i, j) de la imagen I.
- K: Matriz de vecindad K_{NxN} únicamente compuesta de todos sus elementos puestos a 1 y aplicada como una máscara sobre la imagen.
- s: Píxel de la matriz de vecindad.

Si se analiza detenidamente la Ecuación 2.39, esta representa la siguiente regla de decisión: *Si al menos un píxel vecino de* K_{NxN} *sobre el píxel que se está analizando* i *tiene un valor diferente de* 0*, el píxel analizando (*i*) se pone al máximo valor (*255 *para imágenes binarias de 8-bits)*. En la Figura 2.25 se puede observar cómo todos los elementos que componen el escudo se han expandido llegando a cerrar partes del núcleo de ferrita con un valor de vecindad K_{3x3}.



a) b)

Figura 2.25
Operador de dilatación en escudo de titulación de informática. a) Imagen binaria del escudo. b) Resultado del operador de dilatación.

El segundo operador básico, es el operador de erosión, y realiza la operación opuesta al operador de dilatación. En este caso son los píxeles del fondo, y no los de la forma del objeto, los que se expanden, contrayendo así la forma de la figura. La Ecuación 2.40 es la que rige este comportamiento.

$$\zeta(I) = \{i | \forall s \in K, i + s \in I\} \tag{2.40}$$

Donde:

$\zeta(I)$: Resultado del operador de erosión para la imagen I.

i: Píxel con coordenadas (i, j) de la imagen I.

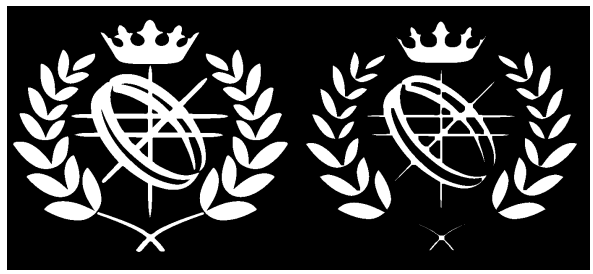
K: Matriz de vecindad K_{NxN} únicamente compuesta de todos sus elementos puestos a 1 y aplicada como una máscara sobre la imagen.

s: Píxel de la matriz de vecindad.

Nuevamente si analizamos el comportamiento de la Ecuación 2.40, detectaremos que representa la siguiente regla: *Si todos los píxeles vecinos de* K_{NxN} *sobre el píxel que se está analizando tienen un valor diferente de* 255*, el píxel analizado (*i*) se pone al mínimo valor* 0. El resultado de erosionar el escudo de la titulación de informática con un valor de vecindad K_{3x3}, se puede observar en la Figura 2.26 donde todos los elementos que componen la forma se han visto contraídos, e incluso han desaparecido como en el caso de las ramas de laurel y olivo.

2.6.2 *Apertura y cierre*

Hemos observado como los operadores básicos anteriores conseguían un objetivo opuesto, y esta oposición de objetivos hacen que surjan los



a) b)

Figura 2.26
Operador de erosión en escudo de titulación de informática. a) Imagen binaria del escudo. b) Resultado del operador de erosión.

operadores de apertura y cierre. Estos operadores lo que persiguen es aprovechar el objetivo de cada uno de los operadores anteriores combinándolos para dar solución a determinados problemas.

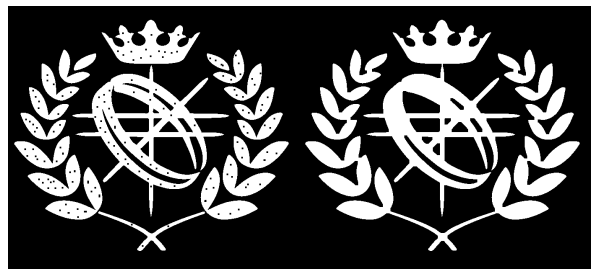
La apertura combina los dos operadores anteriores de forma que primero se realiza una erosión ($\zeta(I)$) y luego se realiza una dilatación ($\varphi(I)$). Con esto se consigue realizar una reducción de ruido muy básica y eliminar aquellos pequeños artefactos que aparecen en una imagen. Como el escudo de la Figura 2.24 no dispone de ruido y sería muy difícil ver el efecto de este operador, se ha introducido ruido manualmente en el fondo tal y como se puede observar en la Figura 2.27. Podemos observar en esta última figura cómo al aplicar el operador de apertura con un valor de vecindad K_{3x3}, se obtiene un resultado muy similar a la imagen original sin ruido.

a) b)

Figura 2.27
Operador de apertura en escudo de titulación de informática. a) Imagen binaria del escudo con ruido en fondo. b) Resultado del operador de apertura.

Del mismo modo que ocurría con los operadores básico, el operador de cierre es el opuesto del de apertura, y en este caso se realiza una dilatación ($\varphi(I)$) seguida de una erosión ($\zeta(I)$). Mediante la aplicación de estos dos enfoques, se consigue rellenar los elementos huecos que puedan aparecer en una figura, pero que a diferencia del simple uso de la dilatación este operador mantiene en cierta medida el tamaño del objeto original. Nuevamente se ha utilizado la imagen del escudo de la titulación de informática y usado un valor de vecindad K_{3x3} como en los casos anteriores, pero esta vez se ha introducido ruido de forma manual dentro de la forma del objeto, como se puede observar en la

Figura 2.28. Además, en esta misma figura se puede observar cómo todas las zonas huecas se han rellenado, aunque el núcleo de ferrita se encuentra ligeramente rellenado.



a) b)

Figura 2.28
Operador de cierre en escudo de titulación de informática. a) Imagen binaria del escudo con ruido en escudo. b) Resultado del operador de cierre.

2.7 ANÁLISIS DE INFORMACIÓN GLOBAL

Hasta ahora todos los métodos de procesamiento de imágenes que se han visto en las secciones anteriores (*con la excepción de las representaciones del color, Sección 2.2*) se basaban en la modificación de los píxeles en la imagen y en la extracción de información a nivel local al propio píxel o considerando los vecinos más cercanos. Sin embargo, existen métodos que se centran en extraer información del ámbito global de la imagen. Entre el abanico de dichos métodos este documento se centrará principalmente en dos: Los *histogramas* y las *integrales proyectivas*:

2.7.1 *Histogramas*

El histograma de una imagen [179] es un método que se centra en realizar un conteo del número de píxeles que tiene un determinado nivel de intensidad. Aunque el funcionamiento del método es bastante sencillo, a través de dicho conteo se pueden realizar operaciones similares a las vistas en la sección 2.4 referentes a la mejora del contraste en una imagen [77, 111, 198]. Formalmente el cálculo del histograma queda representado por la Ecuación 2.41:

$$\mathrm{Hist}(I) = \{H(v) | v = [0, \ldots, L-1]\} \qquad H(v) = n(v) \tag{2.41}$$

Donde:

$\mathrm{Hist}(I)$: Resultado del cálculo del histograma para la imagen I.

$H(v)$: Resultado del histograma para el valor de intensidad v.

$L-1$: Nivel de intensidad máximo (256 *en escala de grises para una profundidad de 8-bits*).

$n(v)$: Función que contabiliza el número de ocurrencias del valor v en la imagen.

Aunque es posible emplearlo en imágenes en color, para simplificar el resultado de la gráfica que se puede obtener del histograma, en la Figura 2.29 aparece representado el resultado del cálculo del histograma para la imagen monocromática en escala de grises de la Figura 2.4 que tiene una resolución de 2048x1376 píxeles. Para imágenes con un número de canales mayor, habría que calcular un histograma separado para cada canal.

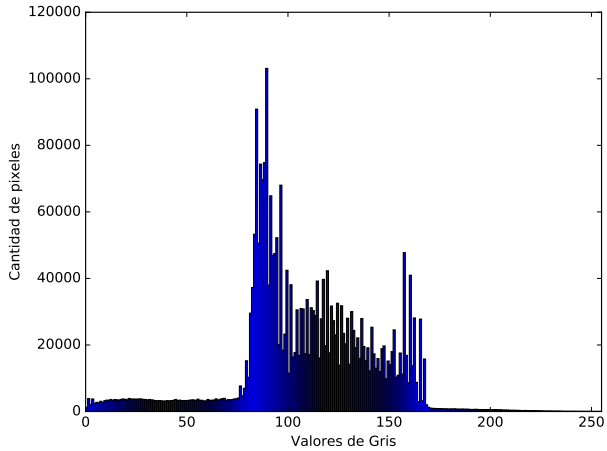


Figura 2.29
Resultado del cálculo del histograma

2.7.2 *Integrales proyectivas*

Las integrales proyectivas [63, 96], también conocidas como proyecciones, se trata de una técnica que extrae información relativa a las acumulaciones de valores en un ángulo determinado. Estos valores acumulados, nos permiten detectar en la imagen dónde se encuentra por ejemplo, los valores máximos y mínimos de intensidad globales en una imagen. Aunque se pueden emplear a lo largo de cualquier ángulo, normalmente se suelen emplear dos tipos, las integrales proyectivas horizontales, y las integrales proyectivas verticales, que corresponden con el sentido paralelo del eje "X" y del eje "Y" respectivamente.

Las integrales proyectivas verticales extraen la información del acumulado de valores de intensidad por columnas en una imagen tal y como muestra la Ecuación 2.42.

$$\Phi(I_j) = \sum_{i=0}^{C-1} I(i,j); \forall j = 0, ..., R-1 \tag{2.42}$$

Donde:

$\Phi(I_j)$: Resultado del cálculo de la integral proyectiva vertical para la imagen I.

C: Número máximo de columnas en la Imagen I.

R: Número máximo de filas en la imagen I.

i: Iterador para las filas.

j: Iterador para las columnas.

De forma análoga podemos deducir que las integrales proyectivas horizontales es el caso opuesto, obteniendo el acumulado de los valores de intensidad por filas tal y como muestra la Ecuación 2.43.

$$\Theta(I_i) = \sum_{j=0}^{R-1} I(i,j); \forall i = 0, ..., C-1 \tag{2.43}$$

Donde:

$\Theta(I_i)$: Resultado del cálculo de la integral proyectiva horizontal para la imagen I.

C: Número máximo de columnas en la Imagen I.

R: Número máximo de filas en la imagen I.

i: Iterador para las filas.

j: Iterador para las columnas.

Del mismo modo que sucedía con el histograma, las integrales proyectivas se pueden emplear en cualquier tipo de imágenes, pero se ha decidido por simplicidad mostrar en la Figura 2.30 el resultado de las integrales horizontales y verticales para la Imagen en escala de grises de la Figura 2.4.

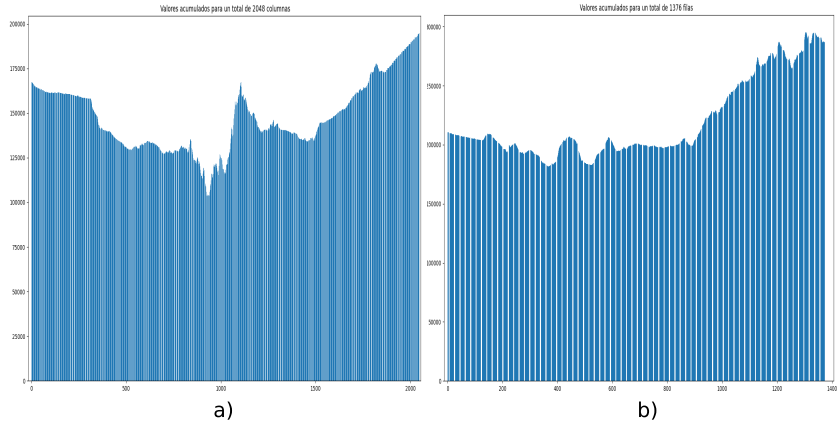


Figura 2.30
Resultado integrales proyectivas verticales y horizontales. a) Integral proyectiva vertical. b) Integral proyectiva horizontal.

2.8 RESUMEN DEL CAPÍTULO

Este capítulo ha presentado los conceptos básico del procesamiento de imágenes partiendo de cómo se almacenan en una computadora las imágenes, hasta cómo extraer información relevante de las mismas.

En lo referente a los modelos y espacios del color se ha mostrado que la forma de representar una imagen no es única y no necesariamente debe de ser siempre en *RGB*. En realidad, tal y como se ha mostrado en el resto de secciones, por ejemplo, en la extracción de características, debido a su simplicidad por procesar un único canal, impera la representación en escala de grises.

Respecto a los algoritmos detallados a lo largo del capítulo, su utilidad depende del ámbito de aplicación, pero como se ha podido observar, hay métodos como los filtros de mejora que nos ayudan a suavizar las imágenes para poder por ejemplo extraer bordes de ellas sin que existan falsos positivos o disminuir el número de píxeles diferentes que existen en la imagen.

También se ha podido observar que existen métodos como los operadores morfológicos que trabajan a un nivel mucho más bajo que la escala de grises mediante imágenes binarias (*en la que sólo se manejan dos tipos de valores* 0 *ó* 255) modificando la forma de los diferentes objetos de la imagen, y que es posible obtener dichas imágenes binarias a través de los métodos de segmentación basados en umbrales de valores.

Además, se ha podido observar la existencia de otros métodos que no trabajan a nivel local mejorando las imágenes, sino que nos permiten obtener información estadística del número de píxeles totales que tienen un determinado valor de intensidad (*histogramas*), o la información acumulada de intensidades por filas o por columnas de la imagen (*como en el caso de las integrales proyectivas*).

Por último, queda aclarar que no todos los métodos expuestos en esta sección se emplearán en la presente Tesis Doctoral, pero sí que todos los métodos empleados en ella están incluidos en esta sección (*véase la Sección 5.2*). El resto de métodos se han introducido para dotar al lector de una vista general de las alternativas que existen y que son empleados por otros autores en la temática de la presente Tesis Doctoral de detección de defectos en el pavimento como se puede observar en el Capítulo 4.

3 PRELIMINARES EN APRENDIZAJE AUTOMÁTICO

La inteligencia es la capacidad de adaptarse al cambio.
Stephen William Hawking [1]

ÍNDICE

1 Stephen William Hawking (1942 - 2018) de origen británico, fue físico teórico y divulgador científico diagnosticado de esclerosis lateral amiotrófica a la temprana edad de 21 años, y que pese a sus limitaciones motrices era un claro ejemplo de que el conocimiento nunca debe de ser una limitación. Galardonado con más de diez doctorados *honoris causa* y premios como el *Premio Príncipe de Asturias de la Concordia, la Medalla Copley y la Medalla de la Libertad* entre otros.

3.1 INTRODUCCIÓN

¿Alguna vez se ha parado a pensar cómo es posible que cuando hacemos una consulta en un buscador web (*ej. google*) nos aparezcan los resultados justamente de lo que estamos buscando? o ¿cómo es posible que nuestro teléfono se capaz de categorizar automáticamente en tipos las fotos que realizamos? o incluso, ¿cómo nuestro correo electrónico es capaz de separar el correo normal del correo que se considera SPAM? Todas estas cuestiones y muchas más son problemas cotidianos que pasamos desapercibidos y que se delegan a una *Inteligencia Artificial*.

SPAM: *Correo no deseado*

La inteligencia artificial (*IA*) no es más que un concepto que se centra en hacer posible que las máquinas y los dispositivos que solemos utilizar, lleven a cabo tareas de forma inteligente al nivel que podría realizarlas un ser humano o incluso mejor. Sin embargo, este concepto necesita de herramientas para llevarse a la práctica, y uno de los ejes fundamentales de la inteligencia artificial, es el aprendizaje automático o APRENDIZAJE DE MÁQUINA.

APRENDIZAJE DE MÁQUINA: *En inglés "Machine Learning"*

El aprendizaje automático engloba a todos aquellos métodos y algoritmos que habilitan a una máquina o sistema a realizar tareas de forma inteligente a través del autoaprendizaje sin la necesidad de haber programado previamente dichos sistemas para llevar a cabo esas tareas. A su vez, tal y como se muestra en la Figura 3.1, en el aprendizaje automático se distinguen diferentes enfoques. Nótese que existen más enfoques que no serán introducidos en esta Tesis Doctoral.

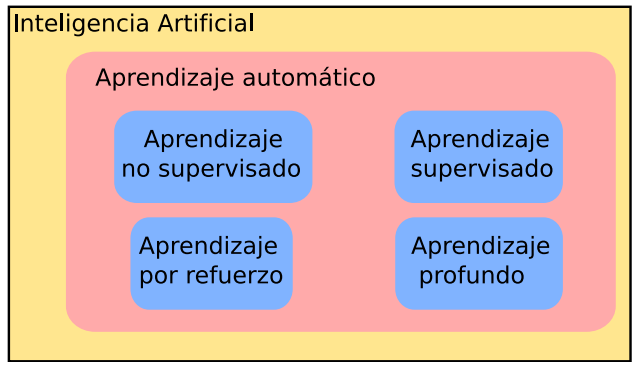


Figura 3.1
Clasificación de inteligencia artificial y aprendizaje automático.

El *aprendizaje no supervisado* [86, 110] consiste en aquellos métodos en los que el aprendizaje se lleva a cabo de forma autodidacta y sin un experto detrás que indique el camino a seguir o la etiqueta de la clase a la que corresponde un dato.

El punto opuesto del aprendizaje no supervisado es el *aprendizaje supervisado* [120, 123], en el que un experto categoriza cada uno de los datos en diferentes clases o tipos, y los métodos tienen que ser capaces de descubrir las relaciones entre dichos datos para poder diferenciarlos al igual que haría un experto.

En el caso del *aprendizaje por refuerzo* [7, 124] la filosofía es diferente, centrándose en maximizar el beneficio de un agente (*ente que toma las decisiones*) para dar solución a un escenario dado (*problema a resolver*). En realidad, se puede ver un símil del funcionamiento de este tipo de aprendizaje en el juego del ajedrez sin ir más lejos, en el que el agente correspondería con el algoritmo de inteligencia artificial que debe de decidir qué acciones realizar y obtener una recompensa asociada a las mismas e ir aprendiendo mediante la experimentación en vez de la imitación como en los casos del aprendizaje supervisado o no supervisado, debido a que no existen datos previos.

En lo referente al *aprendizaje profundo* [46, 112], su filosofía consiste en una similitud a cómo el cerebro humano es capaz de aprender nuevos problemas. Para ello, emplean diferentes niveles de aprendizaje mediante datos que no existen en el conjunto inicial con los que el algoritmo aprende. Su estructura es muy similar a uno de los algoritmos de aprendizaje supervisado (*en concreto las Redes Neuronales, véase la Sección 3.3.7*) y en muchas ocasiones se suele considerar como un subtipo de estos. No obstante, este tipo de aprendizaje puede ser tanto supervisado [83] como no supervisado [115], una mezcla de ambos [15] o incluso estar basados en aprendizaje por refuerzo [12].

Teniendo todo lo anterior en cuenta, sería imposible abarcar toda la casuística de métodos existentes, y por ello, en las siguientes secciones se mostrarán diferentes algoritmos de aprendizaje automático prestando especial importancia a los algoritmos no supervisados y haciendo un mayor énfasis en los algoritmos supervisados debido a que son los métodos utilizados en la presente tesis. En cuanto a los algoritmos de aprendizaje profundo se detallará el funcionamiento general necesario para entender en que se basan las propuestas de otros autores (*véase el Capítulo 4*), y el aprendizaje por refuerzo queda fuera del ámbito de la presente Tesis Doctoral. Por otro lado, de forma adicional dado que los algoritmos están estrechamente relacionados a los datos, este capítulo proporciona las herramientas necesarias para llevar a cabo el diseño de experimentos, así como las métricas utilizadas para evaluar el rendimiento de los algoritmos.

Antes de proseguir y para evitar confusiones en las siguientes secciones y capítulos, es necesario dejar clara la siguiente terminología empleada a lo largo del documento:

- *Clase o etiqueta de clase*: Representan los diferentes tipos de elementos a distinguir de nuestro problema (*ej. el tipo de cada una de las imágenes de la Figura 1.1*).
- *Patrón o instancia*: Elemento compuesto por varios datos que puede estar etiquetado o sin etiquetar.
- *Atributos o características*: Diferentes datos que definen a cada patrón. Pueden ser nominales o numéricos.
- *Modelo*: Conjunto de reglas o funciones para realizar la clasificación de los patrones.

3.2 ALGORITMOS NO SUPERVISADOS

Los algoritmos y métodos no supervisados son un área del aprendizaje automático, que se centran en extraer relaciones existentes entre los datos, sin que haya un conocimiento experto previo que les guíe en su proceso de descubrimiento. Debido a esta naturaleza en su funcionamiento, a continuación se pueden destacar algunos de los problemas a los que da solución el aprendizaje no supervisado:

- Agrupación de datos [167].
- Extracción de prototipos [132].
- Detección de VALORES ATÍPICOS [90].
- Compresión o disminución del espacio de características [29].

VALORES ATÍPICOS: *En inglés: "outliers"*

Los tres primeros problemas se pueden resolver mediante los ALGORITMOS DE AGRUPAMIENTO, y en el último problema aunque también se pueden emplear estos algoritmos, es más usual utilizar los algoritmos de compresión no supervisados. Como el objetivo de la presente Tesis Doctoral no está centrado en el uso de algoritmos de agrupamiento, sólo se explicará de forma genérica en qué consiste cada uno de los tipos sin profundizar en la explicación de algoritmos concretos, y en cuanto a los algoritmos de compresión sí se detallará con más detenimiento uno de los métodos más usuales por ser utilizado para su comparación (*véase la Sección 6.5*).

ALGORITMOS DE AGRUPAMIENTO: *Más conocidos por su término en inglés como "clustering algorithms"*

3.2.1 *Métodos de agrupamiento*

Los métodos de agrupamiento se centran en la unión o separación de los datos basándose en varias estrategias de similitud. Existen diferentes tipos de métodos, pero en general se pueden resumir teniendo en cuenta su estrategia de unión/separación en tres grupos [148, 152]:

Métodos basados en particiones

El funcionamiento general de este tipo de métodos es realizar K divisiones de un conjunto de datos X. Estas divisiones o particiones deben de cumplir las siguientes reglas:

- Cada instancia debe de ser asignado a una partición k_i.
- Cada partición debe de contener al menos un elemento $n(k_i) > 0$, donde $n()$ representa la cardinalidad de un conjunto de datos.
- Cada partición representa un grupo de instancias del conjunto de datos original $n(X) = \sum n(K_i)$.

Para cumplir estas reglas, los métodos de particionado suelen recolocar las particiones de forma iterativa. De esta forma, en cada iteración se refinan los elementos que componen cada partición k_i como se puede observar en la Figura 3.2. El criterio de parada de las iteraciones es diferente para cada algoritmo, pero por lo general suele ser cuando dejan de existir nuevas asignaciones de los patrones a cada partición. Hay

que notar que el número de particiones por lo general es un parámetro que debe de ser establecido de antemano y que la posición espacial de cada partición puede pertenecer a uno de los datos que lo componen, o bien ser por ejemplo, una media de ellos. Algunos de los algoritmos que se engloban dentro de esta metodología son *K-means* [155] o *K-Medoid* [157] entre otros.

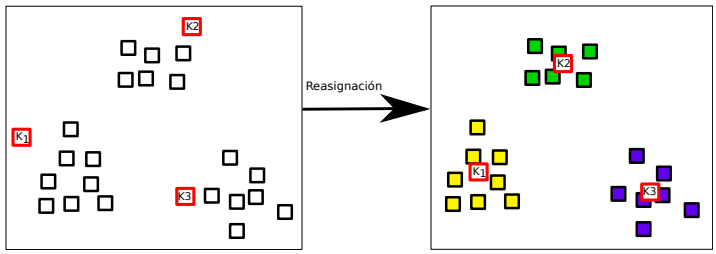


Figura 3.2
Ejemplo de reasignación de particiones.

Métodos jerárquicos

En los métodos jerárquicos [150], a diferencia de los métodos particionales, el número de grupos oscila en el rango [1-N] donde N es el número de instancias que existen en el conjunto de datos con el cual el algoritmo trabaja. Teniendo en cuenta este rango, se pueden considerar dos modos de agrupamiento jerárquico: *aglomerativos* y *divisivos*.

Los métodos aglomerativos generan un grupo por cada instancia en el conjunto de datos y realiza agrupamientos basados en la similitud entre parejas de instancias hasta que todas son cubiertas por dos grupos.

El funcionamiento opuesto se denomina agrupamiento divisivo, en el que se empieza con un único grupo que contiene todos los patrones, y se divide de forma iterativa utilizando la instancia más alejada del resto para formar un nuevo grupo.

Cómo se puede observar ambos métodos están basados en distancias, y con ellas se genera una matriz de proximidad donde las filas y las columnas de la matriz representan los patrones del conjunto de datos con el que entrenan los algoritmos. La diagonal de dicha matriz tendría distancia "0" al ser la distancia hasta el propio elemento. Como métrica de distancia es usual emplear la distancia euclídea, aunque se puede usar cualquier métrica de distancia para el cálculo. Sin embargo, la métrica de la distancia sólo proporciona el valor numérico. La definición de la distancia entre grupos, la proporciona el tipo de enlace que se emplea para juntar o separar los diferentes grupos en cada iteración. Estos enlaces se suelen llevar a cabo mediante uno de los tres criterios siguientes [103]:

- *Enlace simple*: donde la distancia entre dos grupos se define como la distancia más corta entre dos puntos que componen dichos grupos.
- *Enlace promedio*: donde la distancia entre dos grupos se define como la distancia más larga entre dos puntos que componen dichos grupos.
- *Enlace completo*: donde la definición de distancia entre grupos corresponde con el promedio de las distancias entre de cada punto de un grupo con todos los puntos del otro grupo.

DENDROGRAMA: *Representación gráfica en forma de árbol que divide datos en categorías*

Una vez seleccionada la métrica de distancia y el método de enlace, con cada iteración del algoritmo se construye poco a poco un DENDROGRAMA que finaliza cuando se hayan agrupado todos los datos. En la Figura 3.3 se muestra un ejemplo de método aglomerativo con distancia euclídea y enlace simple.

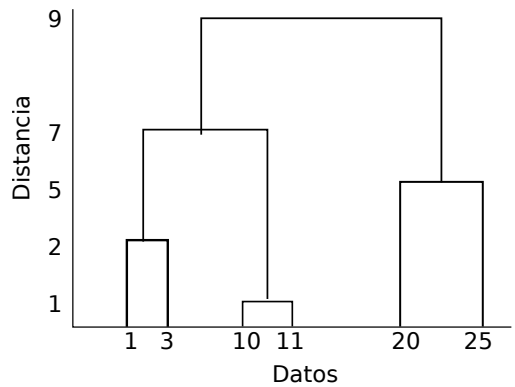


Figura 3.3
Ejemplo de agrupamiento jerárquico.

Métodos basados en densidades

Los métodos basados en densidades [166] se centran en detectar agrupaciones de datos que no son lineales y que por tanto, contienen una forma arbitraria. Al estar basados en la densidad de los datos, estos métodos no necesitan conocer de antemano el número de agrupaciones, pero necesitan conocer dos parámetros, el radio de vecindad (R) y el número mínimo de datos (k) dentro de ese radio de vecindad. Por tanto, se distinguen tres tipos diferentes de datos tal y como se puede observar en la Figura 3.4:

- *Datos atípicos*: son valores que se encuentran fuera de la vecindad, y no superan el número mínimo k de vecinos dentro de su radio de vecindad para considerarse como una densidad que conformen un nuevo grupo.
- *Datos de núcleo*: son valores que dentro de su radio de vecindad superan el número mínimo de vecinos k y por lo tanto forman parte de un grupo.

- *Datos de borde*: son valores que aún teniendo vecinos de un núcleo de densidad dentro de su radio de vecindad R, no superan el número mínimo para considerarse núcleos, y por tanto se consideran como un borde de la densidad de dichos núcleos.

El número de grupos se determina una vez que se han clasificado todos los datos en una de las tres categorías anteriores, teniendo en cuenta que todos los datos núcleos y bordes forman parte del mismo grupo si son alcanzables desde al menos un punto cualquiera que forme parte de ellos teniendo en cuenta el radio R. Algunos de los algoritmos que se basan en la agrupación mediante densidades son *DBSCAN* [54], *OPTICS* [10], entre otros.

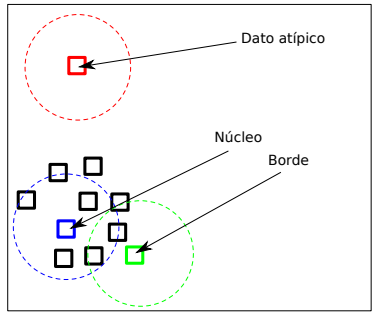


Figura 3.4
Tipos de datos en agrupamiento basado en densidades.

3.2.2 *Métodos de compresión*

Los métodos de compresión de datos se centran en reducir la dimensionalidad de los datos de los que se dispone. Esta reducción puede ser en la cantidad de datos, usando alguno de los métodos anteriores de agrupamiento (*agrupamiento particional, jerárquico o basado en densidades*) para extraer de esta forma prototipos y eliminar gran parte de las instancias de una base de datos, o pueden ser una reducción del número de características que definen a una instancia. A continuación se detallará el funcionamiento de uno de los métodos más extendido para la reducción de datos, el *análisis de componentes principales* (PCA) [37, 87, 125, 190], aunque existen otros como la *factorización de matrices no negativas* (NMF) [94], o la *descomposición de valores singulares* (SVD) [199].

PCA: *Del inglés "Principal component analysis"*

El análisis de componentes principales [2] es un método que se centra en la búsqueda de relaciones en los datos que componen una base de datos. Esta búsqueda permite reducir el número de atributos necesarios para definir a un dato eliminando la información menos relevante o innecesaria mediante métodos estadísticos y operaciones algebraicas.

El conjunto de operadores estadísticos y algebraicos utilizados en el análisis de componentes principales se pueden resumir en cuatro pasos:

- *Cálculo de la media y la matriz de covarianzas*: El primer paso es el cálculo de la media de cada una de las características o atributos de la base de datos. Para ello, los datos se representan mediante una matriz donde las columnas corresponden con los atributos y las filas a cada uno de los patrones de nuestra base de datos. Una vez calculada la media de cada una de las características, la covarianza queda expresada mediante la Ecuación 3.2:

$$C = \frac{1}{n-1}((X-\bar{x})^T(X-\bar{x})) \tag{3.1}$$

Donde:
C: Matriz de covarianzas.

X: Matriz compuesta por cada una de las instancias o patrones de la base de datos.

$\bar{x}$: Vector con la media de las característica.

n: Número de elementos en la base de datos.

- *Cálculo de autovalores y autovectores*: Una vez se ha calculado la matriz de covarianzas, es posible obtener mediante la Ecuación 3.2, los autovalores y autovectores.

$$(C-\lambda\cdot I)\cdot E = 0 \tag{3.2}$$

Donde:
C: Matriz de covarianzas.

λ: Autovalores.

I: Matriz Identidad.

E: Autovectores.

- *Selección del número de atributos*: El siguiente paso es ordenar los autovalores de forma descendente, y seleccionar el porcentaje de representación de variabilidad de los datos. En realidad los autovalores proporcionan información acumulativa respecto a la variabilidad de los datos que pueden representar los autovectores asociados a dicho autovalor. Una vez se ha seleccionado cuantos autovalores proporcionan la variabilidad deseada se construye una matriz S formada por los autovectores asociadas a los autovalores escogidos.

- *Generación de los nuevos datos*: Una vez que se ha seleccionado el número de atributos, se puede realizar una conversión de los datos a un espacio de características reducido (Y) usando la Ecuación 3.3:

$$Y = X\cdot S \tag{3.3}$$

3.3 ALGORITMOS SUPERVISADOS

Otro de los grandes pilares del aprendizaje automático son los métodos supervisados. En general, estos métodos usan los atributos de las instancias como entradas y los procesan para producir una predicción a sus salidas de valores discretos o continuos dependiendo del problema a tratar. Debido a esto, los algoritmos supervisados se dividen principalmente en dos tipos: *métodos de clasificación* y *métodos de análisis de regresión*.

En los métodos de aprendizaje supervisado orientados a la *clasificación* [184], existe un conocimiento experto detrás de los datos que previamente ha analizado cada una de las instancias que componen la base de datos y les ha asignado un tipo determinado mediante una etiqueta. Un ejemplo sencillo de datos etiquetados puede observarse en la Tabla 3.1, donde se ha determinado en base a experiencias pasadas qué días se ha podido jugar al baloncesto. En dicha tabla, los atributos serían cada una de las columnas y la última columna sería la etiqueta de la clase que correspondería con los posibles valores [sí,no], para predecir si es posible jugar o no. Por otro lado, en este tipo de métodos se distinguen dos fases como se puede observar en la Figura 3.5: *la etapa de entrenamiento*, y *la etapa de generalización*.

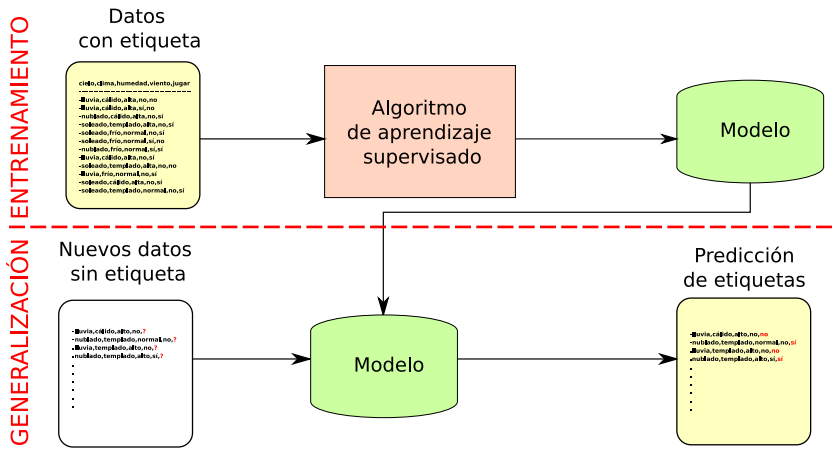


Figura 3.5
Tipos de datos en agrupamiento basado en densidades.

La etapa de entrenamiento emplea datos que han sido etiquetados y se utilizan sus atributos y su etiqueta de clase como entradas de un algoritmo de aprendizaje automático obteniendo las posibles relaciones entre los datos y la etiqueta de la clase. Una vez el algoritmo ha sido entrenado con los datos etiquetados, se ha generado un *modelo de clasificación* que tiene como objetivo ser capaz de etiquetar los datos sin clasificar tal y como lo haría un experto basándose en el entrenamiento que ha recibido, correspondiendo dicha fase a la *etapa de generalización*.

Los métodos de *análisis de regresión* [181] están orientados a analizar la relación existentes entre los atributos de los datos y proporcionar

Tabla 3.1
Base de datos de ejemplo.

cielo	clima	humedad	viento	jugar
lluvia	calido	alta	no	**no**
lluvia	calido	alta	si	**no**
nublado	calido	alta	no	**si**
soleado	templado	alta	no	**si**
soleado	frio	normal	no	**si**
soleado	frio	normal	si	**no**
nublado	frio	normal	si	**si**
lluvia	templado	alta	no	**no**
lluvia	frio	normal	no	**si**
soleado	templado	normal	no	**si**

un valor numérico continuo en vez de una etiqueta. En realidad, estos métodos intentan obtener funciones que sean capaces de predecir un dato numérico como rectas, curvas, etc. Ejemplos de este tipo de algoritmos son la regresión lineal y la regresión logística entre otros. Este tipo de métodos no serán cubiertos en detalle en la presente tesis debido a que quedan fuera de los objetivos que persigue la misma, aunque sí que serán utilizados internamente por algunos métodos de clasificación como uno de sus pasos y se explicarán los conceptos necesarios para entender su funcionamiento dentro del contexto de clasificación.

3.3.1 *Método de los K vecinos más cercanos*

KNN: *En inglés "k nearest neighbor"*

El algoritmo de los *K* vecinos más cercanos (KNN) [18], se considera uno de los métodos más básicos del aprendizaje supervisado. A este algoritmo le suele acompañar el sobrenombre de *perezoso* porque a diferencia de lo que se representaba en la Figura 3.5 para los algoritmos de clasificación supervisados, este no genera un modelo, sino que etiqueta los nuevos patrones en base a los datos de entrenamiento existentes. De este modo cada vez que se va a realizar una predicción se vuelven a explorar todos los datos de entrenamiento.

Los pasos que realiza este algoritmo son los siguientes:

- Selección del número de vecinos k.
- Cálculo de la distancia del nuevo dato a todos los patrones de entrenamiento.
- Ordenar las distancias de menor a mayor y seleccionar como vecinos el número determinado por el parámetro k.
- Escoger como etiqueta para el nuevo dato la clase de la mayoría de los k vecinos.

Tal y como puede observarse de los pasos de este algoritmo, el funcionamiento del mismo es muy similar a los métodos no supervisa-

dos basados en particiones, en concreto al algoritmo de particionado *k-means* [155]. Además, en lo referente a la métrica de distancia al igual que ocurría con los algoritmos no supervisados, se puede emplear la distancia euclídea u otra diferente dependiendo de las características del problema a analizar.

3.3.2 *Método Naïve Bayes*

El método de clasificación NAÏVE Bayes clásico [69], es un método de clasificación probabilístico que se centra en la etiquetación de los datos mediante la extracción de la información que se conoce de los datos disponibles en la etapa de entrenamiento. En realidad, la primera parte de su nombre (*Naïve*) proviene de las siguientes suposiciones en su funcionamiento:

NAÏVE: *Término francés traducido como "inocente/ingenuo o incluso sencillo" en español*

- Se consideran que todos los atributos que definen a cada instancia en la base de datos son estadísticamente independientes entre sí, y que los valores de cada uno no afectan al valor de los demás.
- Se considera que todos los atributos tienen la misma importancia en la representación de la instancia, no habiendo atributos que tengan una relevancia mayor que otros.

La segunda parte de su nombre (*Bayes*) proviene del método empleado para la extracción de información estadística de los datos, *el teorema de Bayes* [106]. Empleando dicho teorema y las suposiciones comentadas anteriormente, es posible realizar el cálculo de la probabilidad de pertenencia a una clase de una instancia nueva no perteneciente al conjunto de entrenamiento, tal y como se expresa en la Ecuación 3.4.

$$P(y_j|x_1...x_n) = P(y_j) \cdot \prod_{i=1}^{n} P(x_i|y_j) \tag{3.4}$$

Donde:

$P(y_j|x_1...x_n)$: Probabilidad a posteriori de pertenencia a la clase y dada las características $x_1...x_n$.

j: Diferentes valores de la clase y

n: Número de características.

$P(y_j)$: Probabilidad del valor j de la clase y_j.

$P(x_i|y)$: Probabilidad condicionada del atributo x_i dado el valor de la clase y_j.

Una vez que se han calculado las probabilidades de pertenencia a cada clase, la etiqueta definitiva se asigna con el valor de la clase que tiene una probabilidad mayor. De este modo, y al igual que sucedía con el método de la sección anterior, no se genera un modelo de clasificación como tal, sino que se genera una regla de decisión para la clasificación de un nuevo patrón.

Se puede observar que si se lleva a cabo el cálculo de las probabilidades de la Tabla 3.1 existen probabilidades que son igual a "0", y que en la Ecuación 3.4 provocarían un resultado igual a "0" al realizar el producto. Por ello, se suele emplear *el método de corrección laplaciano*, que consiste en añadir un valor mínimo (*generalmente "1"*) a todas las características cuando existe alguna probabilidad igual a "0". En la Tabla 3.2 se puede observar el cálculo de las frecuencias relativas usando la corrección laplaciana del ejemplo de base de datos mostrada en la Tabla 3.1 necesarias para aplicar la Ecuación 3.4 y poder predecir la pertenencia de un nuevo patrón.

Tabla 3.2
Frecuencias relativas.

x_{Cielo}	P(Sí)	P(No)
lluvia	$\frac{2}{9}$	$\frac{4}{7}$
nublado	$\frac{3}{9}$	$\frac{1}{7}$
soleado	$\frac{4}{9}$	$\frac{2}{7}$

x_{Clima}	P(Sí)	P(No)
cálido	$\frac{2}{9}$	$\frac{3}{7}$
templado	$\frac{3}{9}$	$\frac{2}{7}$
frío	$\frac{4}{9}$	$\frac{2}{7}$

$x_{Humedad}$	P(Sí)	P(No)
alta	$\frac{3}{8}$	$\frac{4}{6}$
normal	$\frac{5}{8}$	$\frac{2}{6}$

x_{Viento}	P(Sí)	P(No)
sí	$\frac{2}{8}$	$\frac{3}{6}$
no	$\frac{6}{8}$	$\frac{3}{6}$

P(Sí)	P(No)
$\frac{2}{8}$	$\frac{3}{6}$

3.3.3 *Árbol de decisión primero el mejor*

BFTree: *Procedente de su nombre en inglés "Best-First tree"*

El método de clasificación mediante árboles de decisión primero el mejor (BFTree) [58, 160] sigue la filosofía de los *métodos de divide y vencerás* [170] para generar un árbol de clasificación. Este método, construye un modelo basado en particiones binarias con una arquitectura de arriba hacia abajo, donde *los nodos* del árbol son los atributos que separan las instancias en las diferentes clases, *las ramas* son las reglas de decisión para tomar un camino hacia los nodos, y *las hojas* corresponden con la clasificación final de los datos.

En este método tal y como su propio nombre indica, cada división del árbol (*y de los patrones de la base de datos*) en nodos, se realiza tomando el atributo que mejor es capaz de dividir los patrones empleando para ello la métrica de *la impureza*. La impureza es la que representa cómo de mezclados se encuentran los datos y por tanto, cómo de deseable es un atributo para separar los datos en dos clases teniendo en cuenta lo siguiente:

- El valor de la impureza es mayor cuando los datos se dividen de forma uniforme entre los atributos.
- La impureza debería ser igual a 0 cuando todos los datos pertenecen a la misma clases.

Las métricas que se suelen emplear para medir la impureza son el *índice de Gini* [25] que se puede calcular con la Ecuación 3.5, y *la en-*

tropía [8] que se verá con más detalle en la siguiente sección (*véase la Sección 3.3.4*).

$$\text{Gini} = 1 - \sum_{i=1}^{n} (p_i)^2 \tag{3.5}$$

Donde:

n: Corresponde con el número de elementos en la partición.

p_i: Es la probabilidad del elemento i de pertenecer a una clase específica.

Como se ha comentado anteriormente, este método realiza las divisiones de los nodos basándose en la impureza, y construye de este modo un árbol con particiones binarias. En cada partición, los patrones del conjunto de entrenamiento son separados en nuevos conjuntos y se calcula de nuevo la impureza a cada nuevo subconjunto, para realizar una nueva división hasta que se alcanza un criterio de parada. El criterio de parada para dejar de subdividir un nodo puede ser uno de los siguientes:

- Mediante un parámetro M que determina el número de patrones mínimo que se considera en las hojas.
- Cuando la pureza de un nodo es máxima, es decir, cuando todos los patrones pertenecen a la misma clase.

Una vez se ha construido el árbol de decisión, con el objetivo de simplificar la complejidad del árbol generado y evitar que sobreaprenda los datos de entrenamiento, es posible reducir su número de ramas realizando una poda del árbol. Para ello se distingue dos métodos diferentes: *la poda a priori* y *la poda a posteriori*.

En el primer tipo se realiza al mismo tiempo que se construye el árbol, y detiene su crecimiento cuando no hay una asociación estadísticamente significativa entre un atributo y la clase en un nodo particular [53]. Por otro lado, en la poda a posteriori, se deja crecer el árbol completamente y luego se simplifica eliminando partes de las ramas o sustituyéndolas [136].

En la Figura 3.6 se puede observar el modelo resultante en forma de árbol de clasificación usando el índice de *Gini* y no considerando poda del árbol para la base de datos de ejemplo expuesta en la Tabla 3.1.

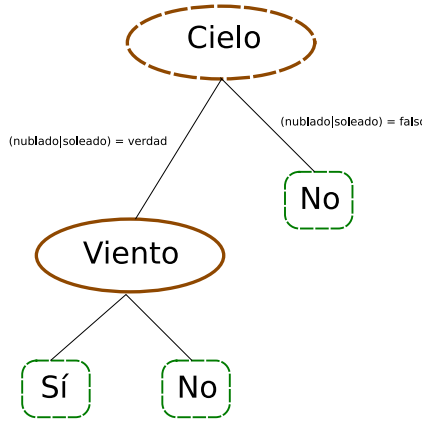


Figura 3.6
Modelo de clasificación con método BFTree.

3.3.4 *Árbol de decisión C4.5*

El árbol de decisión C4.5 [143], al igual que el algoritmo BFTree, se compone de nodos que dividen el árbol, ramas y nodos hojas que representen la clase. Sin embargo, a diferencia de los árboles primero el mejor, C4.5 es un método en el que las ramificaciones en un nodo no tienen porque ser obligatoriamente binarias, sino que pueden tomar todos los posibles valores del atributo.

Aunque el funcionamiento sea similar a los árboles primero el mejor, C4.5 no emplea el *índice de gini* para realizar la selección del atributo que se convertirá en un nodo divisorio, sino que utiliza *la ganancia de información (GI)* [143] que aporta un atributo al ser seleccionado como un nodo, ayudándose de *la entropía (E)* [142] tal y como se muestra en la Ecuación 3.6:

$$\begin{aligned} E(C) &= \sum_{i=1}^{m} -p_i \cdot \log_2 p_i \\ GI(C,A) &= E(C) - \sum_{i=1}^{n} p_{A_i} \cdot E(C_{A_i}) \end{aligned} \tag{3.6}$$

Donde:

$E(C)$: Entropía del conjunto C.

m: Número de clases.

p_i: Frecuencia de la clase i en el conjunto c.

$GI(C,A)$: Ganancia de información de C tras dividir el conjunto empleando el atributo A.

n: Número de valores para el atributo A en el conjunto C.

p_{A_i}: Frecuencia de los valores i del atributo A dentro del conjunto C.

$E(C_{A_i})$: Entropía del subconjunto de C que tiene los valores i.

Una vez conocida la métrica que permite realizar la separación de nodos, los pasos que sigue este algoritmo para construir el árbol se detallan a continuación:

1. Se calcula para cada atributo la ganancia de información que aporta.
2. Se selecciona el mejor atributo que corresponde con aquel que mayor ganancia de información aporta y se crea un nodo cuyas ramas de decisión son los valores del atributo.
3. Se separan todos los ejemplos basándose en los valores de cada rama.
4. Si todos los ejemplos pertenecen a la misma clase se genera un nodo hoja con la etiqueta de la clases en cuestión. En caso contrario se vuelven a repetir los pasos anteriores.

Hay que destacar que C4.5 no puede trabajar con valores continuos y por lo tanto en los casos en los que los atributos tengan esta característica, el algoritmo discretiza dichos valores en rangos para eliminar así su característica continua y tratarlo como si fueran variables nominales.

En este algoritmo, al igual que sucedía con los árboles primero el mejor, se puede realizar tanto una poda a priori como una poda a posteriori. No obstante, las implementaciones más usuales de este algoritmo, suelen usar una poda a posteriori basadas en reemplazar partes del árbol mediante *una poda pesimista*, en la que si el error combinado para un conjunto de ramas es mayor que el del nodo padre, todas esas subramas se podan. Dicho error se calcula atendiendo al número mínimo de instancias por hoja y a un factor de confianza c dado, tal y como puede observarse en la Ecuación 3.7.

$$e = \frac{f + \frac{z^2}{2N} + z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}} \tag{3.7}$$

Donde:

e: Error estimado para un nodo.

z: Valor derivado del factor de confianza c, obtenido mediante una distribución normal (*Ej. c=25 % z=0.69*).

N: Número de patrones cubiertos en el nodo hoja.

La Figura 3.7 muestra el modelo generado por el método C4.5 para los valores de ejemplo de la Tabla 3.1, donde no se ha utilizado poda por tratarse de un ejemplo sencillo, ya que tal y como se puede observar, el árbol generado tiene un único nodo (*nodo* RAÍZ) del cual cuelgan directamente nodos hojas.

RAÍZ: *Primer nodo del árbol*

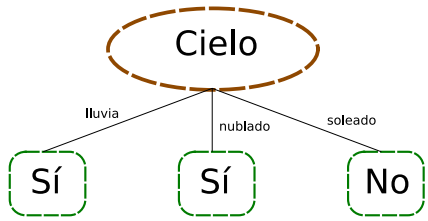


Figura 3.7
Modelo de clasificación con método C4.5.

3.3.5 *Árboles de decisión basados en modelos logísticos*

LMT: *Del inglés "Logistic model trees"*

Los árboles de modelos logísticos (LMT) [97], son un tipo de árbol especial que combina los árboles de inducción (*como los vistos anteriormente* junto con modelos de regresión (*similares a los métodos de análisis de regresión que se introdujeron en la Sección 3.3*). En concreto, la estrategia de separación entre nodos para dividir los datos es similar al algoritmo C4.5, y se vuelve a construir un árbol con nodos hojas y bifurcaciones con ramas. Sin embargo, en este método a diferencia de los algoritmos anteriores, las hojas no están compuestas por una hoja que representa la etiqueta de la clase, sino que las hojas son modelos de regresión que se emplean para decidir la clase a la que corresponde cada uno de los datos.

El árbol de modelos de regresión logística se construye empleando el algoritmo LogitBoost [97], el cual es un algoritmo iterativo, en el que el número de particiones se halla mediante una validación cruzada [176] (*Véase más información sobre la validación cruzada en la Sección 3.6.3*). Los pasos de este método se pueden resumir en los siguientes:

1. El nodo raíz divide los datos en dos grupos empleando el algoritmo LogitBoost.
2. Los hijos del nodo raíz se siguen dividiendo empleando el algoritmo LogitBoost hasta que se alcanza uno de los siguientes criterios de parada:
   - Si un nodo contiene menos instancias que el valor de un parámetro M, se deja de dividir. Generalmente este número suele ser *quince*, siendo este valor lo suficientemente grande para asegurar que los modelos en las hojas tengan suficientes datos para realizar el ajuste.
   - Sólo se realiza una división de un nodo en dos nuevos conjuntos de datos, si cada subconjunto tiene al menos dos datos dentro de cada uno, y sólo si se alcanza una ganancia de información mínima que se establecía para el algoritmo C4.5.
   - Sólo se realiza una división si un nodo contiene un mínimo de cinco patrones. Este número viene impuesto por el algoritmo LogitBoost para poder realizar la validación cruzada.

3. Una vez que se ha construido el árbol, se le aplica una poda a posteriori empleando para ello la métrica CART [25], eliminando de esta forma las partes innecesarias del árbol.

En el ejemplo de la base de datos de cuándo jugar al baloncesto de la Tabla 3.1, todos los atributos son nominales por lo que es necesario llevar a cabo su transformación a valores numéricos para poder realizar la regresión logística del algoritmo LogitBoost. Como resultado del método LMT, no llega a generar un árbol debido a la baja cantidad de patrones y que no se necesita realizar una división en nodos, dando lugar a un árbol especial el cual se compone de un único nodo hoja que contiene las funciones de regresión logística para la clasificación entre cada uno de los valores de la clase *jugar*. Estas funciones de regresión se pueden observar en la Ecuación 3.8:

$$\begin{aligned} Clase_{No} = -0.56 \cdot (cielo = lluvia) \cdot 1.17 + \\ (humedad = normal) \cdot -0.92 + \\ (viento = si) \cdot 1.08 \\ \\ Clase_{Si} = 0.56 \cdot (cielo = lluvia) \cdot -1.17 + \\ (humedad = normal) \cdot 0.92 + \\ (viento = si) \cdot -1.08 \end{aligned} \tag{3.8}$$

3.3.6 *Poda incremental repetida para la reducción de errores*

El método de poda incremental para la reducción de errores (RIPPER) [41], a diferencia de los métodos supervisados que se han detallado anteriormente y pese a que en su nombre incorpora el término de *poda*, no tiene como objetivo la construcción de un árbol de decisión, sino que se trata de un método de inducción de reglas de decisión [98]. Este método al igual que los árboles de decisión de las secciones anteriores se basa en la metodología de *divide y vencerás* realizando particiones del conjunto de entrenamiento para construir reglas de forma iterativa. Una peculiaridad de este método a la hora de construir las reglas, es que se centra en realizar una inducción de reglas desde las clases menos frecuente (*y por tanto la que menos ejemplos contienen*) a las más frecuentes.

RIPPER: *Del inglés "Repeated Incremental Pruning to Produce Error Reduction"*

El funcionamiento del algoritmo RIPPER, se puede resumir en los siguientes pasos:

1. *Etapa de búsqueda*: Se realiza una búsqueda para encontrar la clase minoritaria (*cuyos ejemplos pasan a ser de la clase positiva y el resto la clase negativa*).

2. *Etapa de construcción*: Este paso se compone de dos tareas, el *crecimiento* y la *poda* que se repiten hasta que se alcanza uno de los siguientes criterios:
   - Que no existan más ejemplos positivos que cubrir con la regla.

- Que el error al separar los ejemplos positivos y negativos sea mayor del 50 %.
- Que el valor de la métrica de longitud de la descripción (DL) [41] sea 64 bits mayor que la longitud de la descripción más pequeña encontrada hasta el momento.

DL: *Del inglés "Description Length"*

El crecimiento se encarga de agregar de forma progresiva condiciones a la regla atendiendo al valor que proporciona la ganancia de información al igual que se llevaba a cabo en la separación de nodos para el algoritmo C4.5.

La poda se encarga de eliminar de forma inmediata cualquier condición que se haya agregado a la regla atendiendo a métrica de poda [140] (MP) expresada en la Ecuación 3.9:

$$MP = 2 \cdot \frac{p}{(p+n)} \tag{3.9}$$

Donde:

p: Corresponde con ejemplos positivos y pertenecientes a la clase que busca la regla.

n: Corresponde con ejemplos negativos no pertenecientes a la clase que detalla la regla.

3. *Etapa de optimización*: En esta etapa se generan dos variantes para cada regla tomando características de forma aleatoria. La primera variante se genera realizando una regla completamente nueva y la segunda agregando atributos a la regla existente. Para este par de reglas nuevas, se calcula la métrica DL, determinándose cual regla es más larga de las analizadas, y se selecciona aquella con el valor mínimo en dicha métrica. Al final de esta etapa, si en el conjunto de crecimiento queda algún ejemplo positivo, se vuelve a repetir de nuevo *la Etapa de construcción* para generar otra regla que cubra estos ejemplos.

4. *Etapa de limpieza*: Una vez que se han hallado todas las reglas, la métrica DL se calcula de nuevo para todas ellas y todas las reglas que aumentan el valor de esta métrica son eliminadas del conjunto de reglas. Una vez que se han podado las reglas, ya se dispone de las instrucciones para la clasificación de los patrones entre las diferentes clases.

5. Si existen más de dos clases, se vuelve a repetir el proceso desde la fase de búsqueda con la siguiente clase minoritaria para generar nuevas reglas para el resto de clases.

Si aplicamos los pasos del algoritmo a la base de datos de la Tabla 3.1, se obtendrían las reglas de decisión que muestran a continuación:

$$\text{(cielo=lluvia)} \rightarrow \text{Clase}_{\text{No}}$$

$$\text{En cualquier otro caso} \rightarrow \text{Clase}_{\text{Si}}$$

3.3.7 *Redes neuronales*

Otro de los algoritmos supervisados que más se suelen emplear en el aprendizaje automático son las Redes Neuronales Artificiales (ANN) [51, 138] o simplemente Redes Neuronales (NN). Las redes neuronales intentan simular el comportamiento del cerebro humano para llevar a cabo los procesos de clasificación. Biológicamente, uno de los elementos que forman parte del cerebro humano son las neuronas, las cuales se componen de tres elementos (*dendritas, el soma y el axión*) tal y como muestra la Figura 3.8 a). Las dendritas es la parte de la neurona que reacciona a los impulsos eléctricos y en el soma es donde los impulsos son recogidos y tratados por la neurona. Una vez que el soma ha tratado dichos impulsos, transmite otro impulso por el axión a otras neuronas y formar así una red interconectada de neuronas.

ANN: *Del inglés "Artificial Neural Networks"*

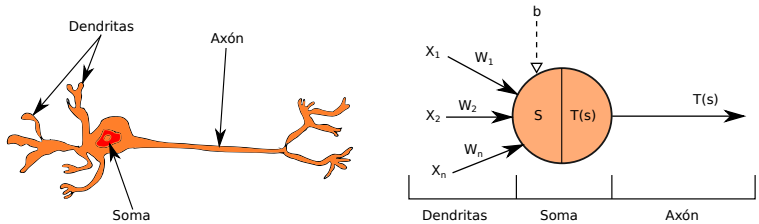


Figura 3.8
Representación de una neurona. a) Partes de una neurona del cerebro humano. b) Representación matemática de una neurona.

En una computadora no tenemos estos elementos biológicos, pero sí que se puede realizar una adaptación de dicho proceso de funcionamiento mediante variables de entradas $(x_1, x_2, .., x_n)$, que corresponderían las dendritas, y una función de transferencia que haría la función del soma (T(S)) tal y como se puede observar en la Figura 3.8 b). Además, en esta figura podemos observar que existen una series de pesos (W_i) conectados entre las variables de entradas y la neurona que sirven para representar la importancia de la variable en la neurona. En la parte de la neurona podemos observar que tenemos una entrada especial que se denomina *bias o sesgo* (b) que se trata como un peso adicional que se añade a la función de suma S y representa el desfase de los pesos de las variables de entradas cuando vayan a emplearse en la función de activación o transferencia (T(S)). La función de suma y la función de activación se encuentran reflejadas en la Ecuación 3.10.

$$\begin{aligned} S &= b + \sum_{i=1}^{n} W_i \cdot X_i \\ T(S) &= \frac{1}{1 + e^{-S}} \end{aligned} \tag{3.10}$$

Donde:

S: Función de suma de pesos y entradas a la neurona.

b: Sesgo de la función de suma.

n: Número de conexiones con la neurona.

W_i: Pesos de las conexiones con la neurona.

X_i: Valor de un atributo de entrada o salida de otra neurona.

$T(S)$: Función de activación basada en sigmoides. Esta función suele ser la más utilizada aunque existen otras [48].

Los elementos anteriores son los que intervienen en una única neurona, pero al igual que el cerebro no está compuesto de una sola neurona, una red neuronal en una computadora por lo general tampoco lo está (*aunque esto depende de la complejidad del problema y del diseño de la topología de la red*). Por ello, como se observa en la Figura 3.9, se distinguen diferentes partes. *La capa de entrada* que se compone de los valores de los diferentes atributos de las instancias de la base de datos. *Las capas ocultas* que se componen de un número determinado de niveles con un número específico de neuronas para cada nivel interconectadas entre sí. Y por último *la capa de salida* que proporciona el nivel de afinidad para cada una de las clases de un problema de clasificación.

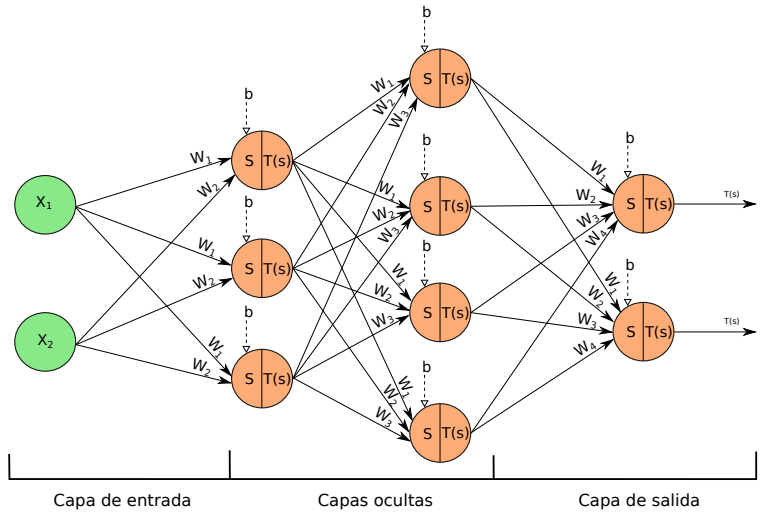


Figura 3.9
Partes principales de una red neuronal.

Tal y como se puede apreciar, la importancia de los pesos W_i y el sesgo b en la red juegan un rol fundamental, y es por tanto el objetivo del aprendizaje de las redes neuronales. De este modo, los pesos se inicializarán en una etapa inicial con valores aleatorios usualmente comprendidos en el rango [0-1] y se actualizarán de forma iterativa (*también denominadas épocas*) en el proceso de aprendizaje del algoritmo. En este sentido existen diferentes propuestas para la actualización de dichos elementos, aunque la más extendida y utilizada es el mecanismo de *retropropagación del error* [40, 153].

La retropropagación del error consiste en la actualización de los pesos de la red neuronal de forma iterativa, de tal forma que como se conocen los errores cometidos por la red en las capas de salida porque los datos están etiquetados, es posible calcular el error empleando la

Ecuación 3.11 y propagar dicho error por cada una de las neuronas de la red hacia las capas de entradas.

$$E = \frac{1}{2} \sum_{p=1}^{P} \sum_{j=1}^{N_L} (t_j - a_j)^2 \tag{3.11}$$

Donde:

E: Es el error cometido.

P: Número de patrones en el conjunto de entrenamiento.

N_L: Número de neuronas en la capa de salida.

t_j: Valor deseado de activación de la neurona j.

a_j: Valor de activación realmente obtenido en la neurona. j

La actualización de los pesos en cada iteración del algoritmo se puede llevar a cabo mediante la Ecuación 3.12 donde cada peso se actualiza atendiendo al valor que tenía en la iteración anterior, y al error que se ha propagado hasta la neurona. Además, se pueden observar dos parámetros (α, μ) [13] que multiplican a cada uno de los elementos de la ecuación que modifican la importancia del valor anterior y la del cambio. *El momento* (α) controla la influencia del valor anterior del peso y *el ratio de aprendizaje* (μ) expresa la importancia que se le va a aplicar al cambio tras propagar el error.

$$w_j(t) = \alpha \cdot w_j(t-1) - \mu \cdot \frac{\partial E(t)}{\partial w_j(t)} \tag{3.12}$$

Donde:

$w_j(t)$: Peso de conexión con una neurona en la iteración o época actual .

$w_j(t-1)$: Peso de la conexión con la neurona en la época anterior.

α: Valor del momento con un rango entre [0-1].

μ: Valor del factor de aprendizaje con un rango comprendido entre [0-1].

En la Figura 3.10 se puede observar el resultado de entrenar una red neuronal, con una capa oculta, una neurona en capa oculta, aplicando 100 iteraciones para el ajuste de los pesos, un valor del momento igual a 0.2 y ratio de aprendizaje igual a 0.3, para el ejemplo de la Tabla 3.1. Nótese, que el uso de una única neurona en capa oculta es poco usual, pero dado que el ejemplo es tan sencillo, al igual que sucedía con los árboles de clasificación tan pequeños obtenidos en las secciones anteriores, no es necesario emplear una arquitectura más compleja para poder clasificar los ejemplos. Además, como se puede observar en dicha figura, las variables de entradas se han transformado a valores binarios.

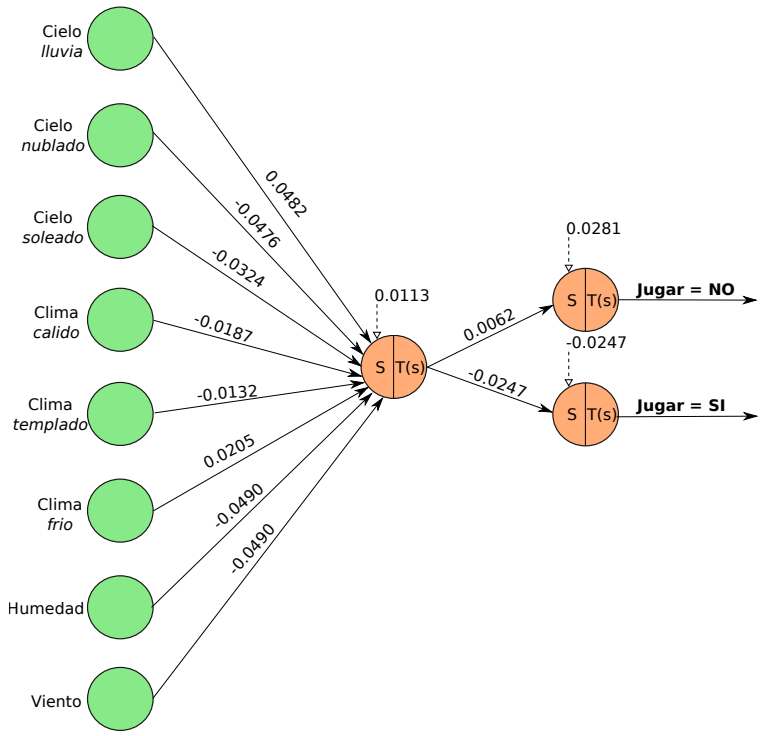


Figura 3.10
Red neuronal entrenada para ejemplo de jugar al baloncesto.

3.3.8 *Máquinas de vectores soporte*

SVM: *Del inglés: "Support Vector Machine*

Las máquinas de vectores soporte (SVM) [72] son un tipo de algoritmo supervisado que se pueden emplear con fines de clasificación o regresión. Su funcionamiento en términos de clasificación se centra en una metodología discriminatoria entre dos clases. Para una clasificación de múltiples clases, se suele confrontar una con todas las demás, extendiendo así el uso de clasificación de dos clases a múltiples de ellas [34].

Para explicar el funcionamiento de las máquinas vectores soportes, vamos a tomar un caso simplificado donde los datos son linealmente separables como se muestra en la Figura 3.11. En esta figura, se puede observar un espacio de características de los datos de dos dimensiones (x_1, x_2) y que los patrones de color rojo y los de color verde son linealmente separables mediante un hiperplano. Sin embargo, la cantidad de valores diferentes que se pueden estimar para los parámetros W y b con el objetivo de separar estas dos clases son diversos. Por ello, el primer requisito, es maximizar la distancia entre los hiperplanos $W^T \cdot X - b = 1$ y $W^T \cdot X - b = -1$ (*siendo X los atributos de entradas de una instancia de la base de datos*) que actúan como bandas separadoras entre las clases. El segundo requisito, es conseguir que no existan datos dentro de la zona entre las bandas, aunque en muchas ocasiones dada la naturaleza de los datos se vuelve una tarea complicada y se suelen añadir dos variables de tal forma que se permita asumir un cierto error (E) y parametrizar la importancia de esos errores C per-

mitiendo así cierta holgura. Por consiguiente, tenemos un problema de optimización en base a restricciones tal y como se observa en la Ecuación 3.13.

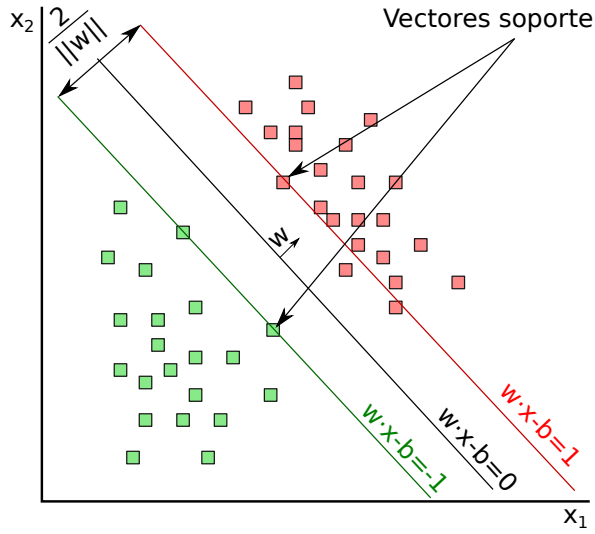


Figura 3.11
Representación de vectores soporte.

$$\min_{W,b} \frac{||W||}{2} + C \sum_{i=0}^{N} E_i \qquad (3.13)$$

Sujeto a: $y_i \cdot (W^T \cdot X_i + b) \geqslant 1 - E_i$

Donde:

y_i: Clase correspondiente. Recordemos que SVM clasifica en clase negativa y positiva [-1,1].

W^T: Traspuesta del vector de pesos que es necesario hallar.

X_i: Atributos de cada uno de los datos i del conjunto de entrenamiento.

b: Sesgo que al igual que el vector de pesos W es necesario hallar.

C: Parámetro de regularización del error.

E_i: Error de cada dato, siendo $E_i \geqslant 0$.

N: Número de ejemplos en el conjunto de entrenamiento.

Tal y como se ha expresado la Ecuación 3.13 es lo que se conoce como problema *primal*, y existe la posibilidad de realizar el cálculo de una forma más eficiente transformándolo a un problema de tipo *dual* [43] haciendo uso de *la función de lagrange* y de *las condiciones de Kuhn-Tucker*, obteniéndose la Ecuación 3.14 [43].

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \sum_{i,j=1}^{N} \left(y_i \alpha_i x_i^T x_j y_j \alpha_j \right) \qquad (3.14)$$

Sujeto a: $0 \leqslant \alpha_i \leqslant C$

Donde se mantienen el mismo significado de los parámetros y los nuevos son:

α_i, α_j: Multiplicadores de lagrange a estimar.

i, j: Patrones del conjunto de entrenamiento.

A partir de la Ecuación 3.14, y de las derivadas parciales de la función de lagrange así como de las condiciones de *Kuhn-Tucker* [43], es posible obtener los valores de W y de los sesgos b sabiendo que los vectores soporte son aquellos datos que tiene un valor $\alpha > 0$ tal y como muestra la Ecuación 3.15.

$$\begin{aligned} W &= \sum_{i=1}^{N} \alpha_i y_i x_i \\ b &= y_i - W^T x_i \end{aligned} \tag{3.15}$$

Una vez hallados los parámetros W y b, basta con emplear la ecuación del hiperplano $W^T \cdot x + b$, y reemplazar x por los atributos del dato que sea desea clasificar, y comprobar el signo del valor que devuelve dicha ecuación. Recordemos que SVM era un clasificador que discriminaba dos clases, con lo que los valores negativos de la ecuación del hiperplano pertenecen a la clase negativa y los valores positivos a la clase positiva.

Para el caso de datos que no sean linealmente separables, SVM aplica lo que se denomina el truco del núcleo, que consiste en aplicar al vector de atributos X_i de la Ecuación 3.13, una función ($K(X_i)$) suponiendo que al transformar y aumentar la dimensionalidad del espacio de características en uno nuevo, los datos sean separables [121].

A continuación, se puede observar el hiperplano resultante así como los parámetros W y b de emplear el algoritmo SVM con un núcleo lineal para los ejemplos de la Tabla 3.1, donde las variables nominales han sido previamente transformadas a una codificación binaria para poder emplear el algoritmo al igual que ocurría con las Redes Neuronales:

```
w = [1.00000 -0.87520 -0.12480 0.31240 0.12480 -0.18760
    -0.31240 0.81240]

b=-0.43747

clase=signo(1.00000*(cielo=lluvia)-0.87520*(cielo=nublado)
-0.12480*(cielo=soleado)+0.31240*(clima=cálido)
-0.12480*(clima=templado)-0.18760*(clima=frío)
-0.31240(Humedad)+0.81240*Viento -0.43747)
```

3.3.9 *Metaclasificadores: Ensembles de modelos*

Los ensembles de modelos también conocidos como metaclasificadores, no son un algoritmo o método que generen un modelo de clasificación como se ha visto en las secciones anteriores, sino que generan un modelo compuesto por modelos basados en un comité de decisión [202], donde las decisiones de etiquetar un dato en una clase recae en la decisión de todos los miembros del comité (*los modelos individuales*). De este modo, podemos entender un ensemble de modelos como un moderador y los modelos de los algoritmos de clasificación que incorpora como los expertos. Por lo tanto, si el moderador lanza una pregunta (*clasificar un nuevo dato*) a los expertos, estos utilizando sus metodologías para hallar la respuesta y sus conocimientos, responden la clase que creen más adecuada. Una vez todos los expertos han tomado la decisión, el moderador recoge todas las respuestas de los expertos, las combina, y toma la decisión final. La respuesta en base a las decisiones de los expertos que puede tomar el moderador suele estar basada en la mayoría simple de los votos de los expertos, aunque existen otros mecanismos [202]. Por consiguiente, se puede observar claramente que la ventaja de los ensembles de modelos recae en la tolerancia a fallos, debido a que si algunos de los modelos de clasificación que componen el ensemble cometen un fallo en su predicción, pero la mayoría de los otros modelos restantes predicen la etiqueta del dato correctamente, el ensemble puede proporcionar una respuesta correcta.

Al tratarse los ensembles de una mixtura de modelos, no existe una metodología única para construir uno [66, 154], aunque se pueden distinguir dos tipos fundamentales, *los ensembles heterogéneos* y *los ensembles homogéneos*:

- *Ensembles heterogéneos*: Los modelos que lo componen son de diferente naturaleza, por ejemplo modelos basados en redes neuronales y árboles de clasificación, proporcionando así una mayor diversidad de metodologías en la toma de decisiones individual de cada modelo.
- *Ensembles homogéneos*: Todos los modelos que componen el ensemble son basados en el mismo algoritmo de clasificación. De este modo, cada modelo se ha entrenado con un mismo algoritmo (*ej: Naïve Bayes*) pero con datos diferentes, proporcionando así una mayor versatilidad en la clasificación basada en un tipo de algoritmo.

Tanto si el ensemble que se decida construir es homogéneo como heterogéneo, si se basa en el voto de la mayoría y se considera que la probabilidad de cometer un error o un acierto es independiente para los diferentes modelos, se puede expresar matemáticamente el rendimiento del ensemble mediante la Ecuación 3.16 [27]:

$$\texttt{Confianza} = \sum_{x > \frac{m+1}{2}}^{m} \frac{m!}{(m-x)! \cdot x!} \cdot p^x \cdot (1-p)^{(m-x)} \tag{3.16}$$

Donde:

p: Es la probabilidad independiente de uno de los modelos de tomar la decisión correcta.

m: Es el número de algoritmos de clasificación que componen el ensemble.

x: Es el iterador del sumatorio.

La Ecuación 3.16 se puede ver más claramente si se representa de forma gráfica como se muestra en la Figura 3.12, donde se ha variado la probabilidad p con valores desde "0" hasta "1" en intervalos de "0.1" en el eje de abscisas y la confianza devuelta por dicha ecuación en el eje de ordenadas. Se puede observar que a diferencia de los resultados que puede obtener un único modelo de clasificación cuando la probabilidad de tomar la opción correcta por los modelos que componen el ensemble es mayor de 0.5 ($p > 0.5$), la confianza del ensemble es mayor que la de un único modelo, y que por el contrario si es menor, el ensemble se comporta peor que un modelo de clasificación individual. Sin embargo, tener en modelos de clasificación un porcentaje de acierto igual a "0.5" es similar a tirar una moneda al aire en la toma de decisiones y generalmente se buscan modelos para componer el ensemble cuyas probabilidades estén por encima de "0.5". Además, en la misma figura se puede observar cómo al aumentar el número de modelos en el ensemble hace que se alcancen mejores valores de predicción cuando la probabilidad de tomar la elección correcta es menor que en la de un único modelo justificando así las ventajas de este tipo de enfoque.

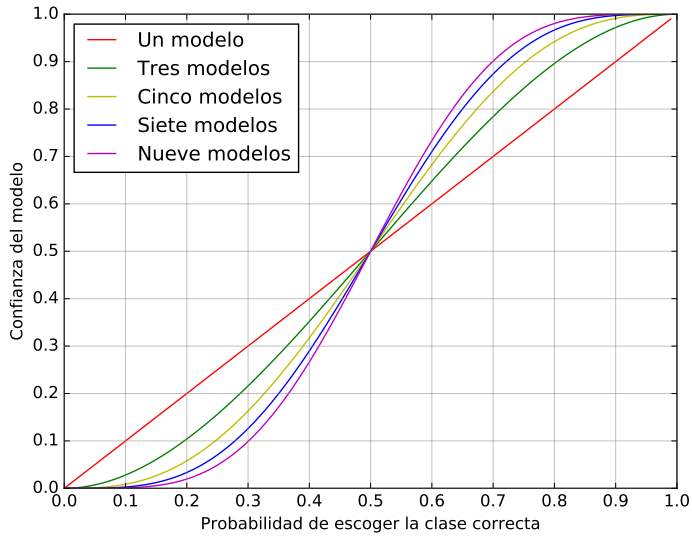


Figura 3.12
Confianza y probabilidad de acierto variando el número de modelos.

3.4 APRENDIZAJE PROFUNDO

EL APRENDIZAJE PROFUNDO [62, 137] es una de las vertientes del aprendizaje automático, en el que la idea básica es emplear un procesamiento jerárquico utilizando muchas capas de una arquitectura similar a la redes neuronales (*véase la Sección 3.3.7*). Sin embargo, a diferencia de lo que sucede en las redes neuronales, las cuales no son capaces de manejar un número elevado de características de entrada, y donde las entradas son características procesadas de un patrón de un problema en concreto (*como el resultado de la extracción de bordes de la Figura 2.23*), en el aprendizaje profundo, los datos se emplean directamente tal cual se encuentran en el entorno natural del problema sin extraer ningún tipo de características previas o procesamiento (*como la Figura 2.1*), siendo la red y las capas que la componen las encargadas de extraer estas características de forma automática.

EL APRENDIZAJE PROFUNDO: *Del inglés: "Deep Learning"*

En este sentido, al necesitar extraer de forma automática las características y a diferencia de los algoritmos de aprendizaje automático, el aprendizaje profundo necesita de una cantidad de datos para realizar su entrenamiento mucho más elevada, siendo un requisito bastante complejo de alcanzar en algunos problemas. Por ello una técnica bastante usual es aumentar los datos originales de entrada aplicándoles rotaciones, y desplazamientos en el caso concreto de las imágenes [3, 162, 168].

Dos de los tipos de arquitecturas de aprendizaje profundo más conocidos que utilizan imágenes como datos de entradas, son *las redes neuronales convolucionadas* y *los auto-codificadores*.

3.4.1 *Redes neuronales convolucionadas*

Las redes neuronales convolucionadas (CNN) [4, 93] suelen ser empleadas para el caso específico de disponer de imágenes como datos de entrada del modelo. Esta característica las hace diferente de las tradicionales redes neuronales (*véase la Sección 3.3.7*) debido a que estas últimas emplean datos procesados para realizar una clasificación (*generalmente un vector de datos*), y en las redes neuronales convolucionadas se suelen utilizar las imágenes sin ningún procesamiento previo. Sin embargo, aunque los datos de partida sean diferentes, este tipo de arquitectura guarda una estrecha relación con las redes con retropropagación de los errores hacia las entradas, tal y como se puede observar en la Figura 3.13 donde se muestra un esquema general de las diferentes componentes de este tipo de redes.

CNN: *Del inglés: "Convolutional Neural Networks"*

Este tipo de redes funciona modelando consecutivamente pequeñas piezas de información y combinándolas más profundamente en la red. Las partes de las que se componen estas redes se detallan a continuación:

- La primera parte y de la cual deriva el nombre de este tipo de redes, es la capa de convolución, la cual se encarga de tomar la

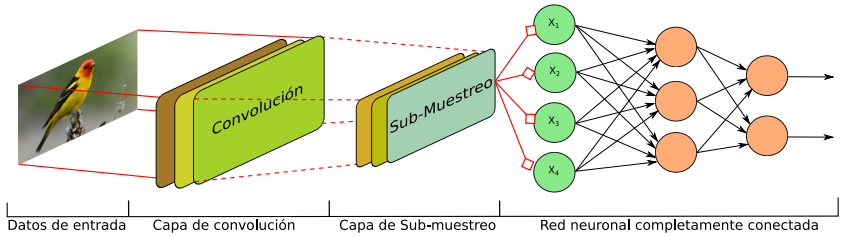


Figura 3.13
Partes principales de una red neuronal convolucionada.

imagen de entrada y extraer características con un proceso similar al que se mostraba en la Figura 2.12. En este paso hay que ajustar tres parámetros:

- *Profundidad*: Define el número exacto de filtros necesarios para la operación de convolución.
- *Avance*: Define la cantidad de píxeles necesarios para deslizar la matriz del kernel de convolución sobre la imagen de entrada.
- *Relleno*: Define si los bordes de las imágenes y las convoluciones de etapas posteriores a la primera, se rellenan con 0 para aplicar el kernel de convolución.

- La rectificación lineal es una etapa posterior a la convolución que suele ser aplicada para la eliminación de la linealidad de los mapas de características obtenidos por la convolución. Uno de los operadores más usuales suele ser la unidad lineal rectificada (RELU) [122] en el que se sustituyen con cero todos los valores negativos del mapa de características de todas las convoluciones.

RELU: *Del inglés: "Rectified Linear Unit"*

- La parte de Sub-muestreo se emplea para reducir el tamaño de los datos de entrada de la siguiente capa de convolución o de la capa de entrada de una red neuronal totalmente conectada. Una técnica usual es aplicar un kernel en el que se escoja el valor máximo de los valores que quedan dentro del kernel.

- Por último una vez que se han obtenido todas las características y se han reducido, éstas se utilizan como entradas de una red neuronal en la que todas las neuronas están totalmente conectadas entre sí con un esquema similar al de la Figura 3.9.

3.4.2 *Auto-codificadores*

AUTO-CODIFICADORES: *En inglés "autoencoders"*

Los AUTO-CODIFICADORES [16, 113] son un tipo de redes neuronales profundas que se componen de capas de entradas, capas ocultas y capas de salida al igual que las redes neuronales tradicionales. Sin embargo, la principal diferencia erradica en que el número de neuronas en la capa de salida, es el mismo que el de la capa de entrada, y en vez de realizar una predicción de valores en la capa de salida, lo que pretenden es predecir los valores de la capa de entrada como se muestra en el esquema de la Figura 3.14.

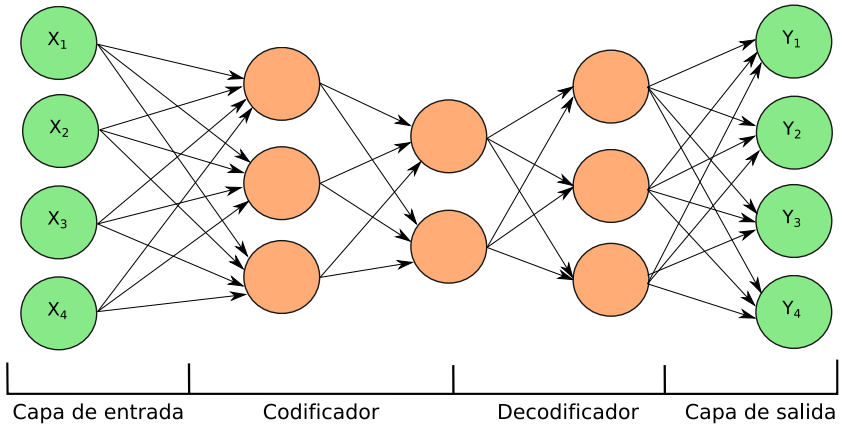


Figura 3.14
Esquema general de auto-codificador.

El funcionamiento general es similar al de las redes neuronales con retropropagación del error (*véase la Sección 3.3.7*):

- Realizar una iteración alimentando las entradas de todas las neuronas que componen la red para calcular su función de activación.
- Calcular el error entre los valores calculados y las entradas.
- Propagar el error hacia las entradas para actualizar los pesos.
- Repetir los puntos anteriores hasta obtener los pesos satisfactorios en todas las neuronas.

Además, se suele utilizar un número menor de nodos en capa oculta que en capa de entrada o de salida (*ambas son iguales en número*) considerándose de esta forma que la última capa oculta de la red es una representación comprimida de las entradas que toma.

Entre las diferentes metodologías de auto-codificadores se pueden destacar las siguientes:

- *Regularización dispersa de Auto-codificadores* [81]: En el caso de que la capa oculta sea mayor que la capa de entrada o la de salida, se aplica una técnica de regularización para evitar que el auto-codificador utilice todos los nodos a la vez.
- *Regularización mediante eliminación* [185]: En este enfoque se toma los datos de entradas modificados convirtiendo aleatoriamente algunos de los valores de entrada a "0".
- *Apilamiento de Auto-codificadores* [161]: Cuando se añaden otras capas ocultas, obteniendo varios codificadores apilados y un único decodificador en la salida.

3.5 MÉTRICAS DE EVALUACIÓN

Una vez que se ha generado un modelo de clasificación con alguno de los algoritmos detallados en las secciones anteriores u otros diferentes, el modelo generado puede ser un clasificador que etiqueta perfectamente cada uno de los patrones que le proporcionamos para predecirlo, o por el contrario incurrir en errores etiquetando ciertos patrones (*el caso más habitual*). Por ello, es de suma importancia saber cómo se comporta realmente el algoritmo con los patrones, y dependiendo de nuestros requisitos, saber si nuestros modelo es o no "*aceptable*" para el problema que estemos tratando. Dichos errores y aciertos es lo que permite crear la denominada *matriz de confusión*.

Tabla 3.3
Matriz de confusión.

		Etiqueta predicha	
		Clase A	Clase B
Etiqueta real	Clase A	TP	FN
	Clase B	FP	TN

		Etiqueta predicha	
		Clase A	Clase B
Etiqueta real	Clase A	50	7
	Clase B	10	78

Una matriz de confusión o matriz de contingencia [68] está compuesta de un conteo de los patrones que han sido clasificados correctamente e incorrectamente y qué etiqueta se le han dado a cada uno de los patrones. Imaginemos un caso sencillo con dos clases (*clase A y clase B*) en las que nuestro modelo etiqueta cada uno de los patrones tal y como se muestra en la Tabla 3.3. Los elementos que componen la matriz de confusión en este caso, son los siguientes:

FP: *En inglés "False positives"*

- *Falsos positivos* (FP): patrones pertenecientes a la clase B que el modelo ha etiquetado como patrones de la clase A.

TN: *En inglés "True negatives"*

- *Verdaderos negativos* (TN): patrones pertenecientes a la clase B que el modelo ha etiquetado correctamente perteneciendo a dicha clase.

TP: *En inglés "True positives"*

- *Verdaderos positivos* (TP): patrones pertenecientes a la clase A que el modelo ha etiquetado correctamente perteneciendo a dicha clase.

FN: *En inglés "False negatives"*

- *Falsos negativos* (FN): patrones pertenecientes a la clase A que el modelo ha etiquetado como patrones de la clase B.

La Tabla 3.3 muestra en el lado izquierdo el caso general para dos clases y en la parte derecha un ejemplo con valores numéricos. Para un problema con un mayor número de clases la matriz se construiría de forma análoga [177].

Sin embargo, aunque la matriz de confusión muestra toda la información relativa a aciertos y fallos por clases de un modelo, es poco útil cuando queremos proporcionar una información de rendimiento resumida. Por ello, la matriz de confusión se considera el paso precursor de las *métricas de evaluación*.

Las métricas de evaluación permiten obtener una unidad tangible para analizar el comportamiento de un modelo y poder comparar los diferentes algoritmos entre sí, aunque sean de naturaleza distinta (*ej. métodos de inducción de reglas, métodos probabilísticos, etc.*). Algunas de las principales métricas de evaluación se detallan a continuación:

3.5.1 *Accuracy*

El ACCURACY [177] expresa el porcentaje de patrones que el clasificador ha sido capaz de etiquetar correctamente en las clases correctas de entre todos los patrones etiquetados. Esta métrica se puede obtener como la suma de la diagonal de la matriz de confusión dividida entre el número total de patrones tal y como muestra la Ecuación 3.17.

ACCURACY: *En español "Precisión"*

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.17}$$

Siguiendo el ejemplo mostrado en la parte derecha de la Tabla 3.3 se obtendría un valor de 0.8827. Esta métrica al igual que el resto suelen aparecer representadas en tanto por ciento, con lo que bastaría con multiplicar el valor por 100.

3.5.2 *Tasa de error*

Si la métrica anterior contaba el número de patrones bien clasificados o etiquetados, la tasa de error [108] detalla justo lo contrario, el número de patrones mal clasificados. Esta métrica se puede obtener al igual que la precisión de la matriz de confusión tal y como muestra la Ecuación 3.18. Sin embargo, dado que el error es el complementario de la precisión, se podría obtener directamente como "$1 - \text{Accuracy}$". En el ejemplo de la Tabla 3.3 obtendríamos un error igual a 0.1172.

$$\text{Error} = \frac{FP + FN}{TP + TN + FP + FN} \rightarrow (1 - \text{Accuracy}) \tag{3.18}$$

Tanto el *accuracy* como el error, son métricas que se utilizan generalmente para analizar el rendimiento global de un clasificador. Sin embargo, tienen serios inconvenientes cuando las bases de datos que intentamos etiquetar están DESBALANCEADAS. En estos casos carece de sentido utilizar únicamente estas métricas debido a que enmascararían el rendimiento del clasificador, proporcionando un buen resultado debido a que la clase con más patrones se clasifica casi a la perfección. Por ello se suelen utilizar dos métricas que informen del rendimiento del clasificador para cada clase como se puede observar en los siguientes apartados.

DESBALANCEADAS: *Base de datos que tiene clases mayoritarias (muchos patrones de una clase) y minoritarias (pocos patrones de una clase).*

3.5.3 *Precision*

La PRECISION [31] indica que porcentaje de los patrones etiquetados como una clase realmente corresponden con dicha clase, es decir, expresa

PRECISION: *En español "Exactitud".*

cómo de exacto es nuestro modelo al clasificar los patrones de una clase. Tomando como referencia la matriz de confusión de la Tabla 3.3 podemos traducir la definición de esta métrica en la Ecuación 3.19 y observar claramente que se trata de una métrica que penaliza especialmente falsos positivos.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.19}$$

Aplicando la Ecuación 3.19 a nuestro ejemplo de la Tabla 3.3, podemos observar que obtendríamos un valor igual a 0.8333 para la clase A y un valor de 0.9176 para la clase B y corroborar fácilmente el efecto de los falsos positivos en ambas clases (*nótese que para el cálculo del valor de la clase B, los elementos FP, TP, FN y TN, de la matriz de confusión estarían invertidos*).

3.5.4 *Recall*

RECALL: *En español "Exhaustividad".*

El RECALL [23] corresponde con la proporción de patrones de una clase que el modelo ha sido capaz de etiquetar correctamente. La Ecuación 3.20 muestra el cálculo de dicha métrica.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.20}$$

A diferencia de la métrica anterior, se puede deducir del ejemplo que tenemos en la parte derecha de la Tabla 3.3 que esta métrica penaliza los falsos negativos en vez de los falsos positivos, proporcionando un valor de 0.8771 para la clase A y un valor de 0.8863 para la clase B (*nótese que para el cálculo del valor de la clase B, los elementos FP, TP, FN y TN, de la matriz de confusión estarían invertidos*).

3.5.5 *F-Score*

Hasta este punto se ha observado que se pueden utilizar métricas para valorar el rendimiento como la *precision* y el *recall* que mitigan los problemas frente al *accuracy* o el *error*. Sin embargo, si queremos proporcionar toda la información posible, tenemos que emplear dos métricas en vez de una sola para evaluar la robustez del método respecto a una clase. En este contexto, existe una métrica llamada F-SCORE [172] que permite aunar los resultados de *precision* y *recall* en un único valor.

F-SCORE: *En español "Valor F" o "Medida F"*

Esta métrica no es tan simple como una media aritmética entre los valores de *precision* y *recall*, sino que además de disponer de los valores de dichas métricas se emplea un parámetro extra denominado α. El parámetro α permite modificar la importancia que se la va a dar al valor de *precision* o al *recall*, o en otras palabras, hacer más énfasis en falsos positivos o no.

De este modo, la Ecuación 3.21 detalla cómo calcular el valor de la métrica *F-Score* de forma general. En el caso de que los falsos positivos

y los falsos negativos tuvieran la misma importancia ($\alpha = 1$), la Ecuación 3.22 representaría la MEDIA ARMÓNICA entre la *precision* y el *recall*, que en nuestro ejemplo de la Tabla 3.3 tendría un valor igual a $0,8547$.

MEDIA ARMÓNICA: *División del número total de valores entre la suma del inverso de los valores. No confundir con la media aritmética (promedio de los valores).*

$$F_\alpha - Score = (1 + \alpha^2) \cdot \frac{precision \cdot recall}{(\alpha^2 \cdot precision) + recall} \tag{3.21}$$

$$F_1 - Score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{3.22}$$

Como se ha comentado antes, esta métrica combina los resultados *Precision* y *Recall* ajustando la importancia con el parámetro α. Sin embargo, no es obligatorio calcular previamente estos valores sino que se puede expresar el valor del $F_\alpha - Score$ y $F_1 - Score$ en términos de la matriz de confusión de la Tabla 3.3 tal y como muestra la Ecuación 3.23 y Ecuación 3.24 respectivamente.

$$F_\alpha - Score = \frac{(1 + \alpha^2) \cdot TP}{(1 + \alpha^2) \cdot TP + \alpha^2 \cdot FP + FN} \tag{3.23}$$

$$F_1 - Score = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{3.24}$$

3.5.6 *Promedio ponderado de métricas*

Como se ha podido observar las métricas anteriormente detalladas están centradas en proporcionar el rendimiento de un clasificador para una clase. El problema viene cuando el número de clases es elevado o simplemente queremos realizar una comparación que recoja a todas las clases. Además, se puede apreciar que la cantidad de patrones influye enormemente en el valor de los resultados. Debido a esto el promedio de los valores obtenidos por cada clase debe de ir ponderado al número de patrones de cada clase si se desea obtener un valor general de las métricas [108] tal y como muestra la Ecuación 3.25:

$$\sum_{i}^{numClass} \frac{n_i}{n_{total}} \cdot M_i \tag{3.25}$$

Donde:

i: Iterador para las clases.

$numClass$: Número total de clases.

n_{total}: Número total de patrones de la base de datos.

M_i: Métrica para la clase i (*ej. Recall*).

3.6 TÉCNICAS DE EVALUACIÓN

Tal y como se ha podido ver en las secciones anteriores los datos juegan un papel fundamental en la generación de los modelos de clasificación y en las métricas de evaluación, dado que dependen de la cantidad de aciertos o fallos al etiquetar los mismos. Por ello es esencial emplear técnicas que nos permitan generar y evaluar los modelos con los datos que disponemos. Llegados a este punto es necesario explicar que los datos se pueden utilizar en los siguientes casos [193]:

- *Entrenamiento*: patrones que se utilizan para la generación de los modelos de clasificación.
- *Validación*: patrones que se utilizan para el ajuste de los parámetros de los modelos de clasificación.
- *Comprobación* (*test*): patrones que se utilizan para comprobar el rendimiento de un modelo con datos no vistos anteriormente.
- *Predicción*: patrones que se utilizan para asignarles una etiqueta de una de las clases que el modelo es capaz de asignar.

Puesto que no hay una única forma de utilizar los patrones de una base de datos, los siguientes apartados muestran los enfoques más usuales que permiten emplear dichos datos para la generación y comprobación de los diferentes modelos.

3.6.1 *Utilización de los datos completos*

El primer enfoque es utilizar todos los patrones que tenemos en nuestra base de datos para entrenar nuestro modelo. Esta acción en sí no es usual debido a que en muchas ocasiones la bases de datos empleadas son difíciles de obtener. Además, existe el inconveniente añadido de que se necesitaría una base de datos nueva para poder obtener las métricas de evaluación que analicen la robustez del modelo generado. Lo que nunca se debe hacer bajo ningún concepto es utilizar los mismos datos con los que se ha entrenado un modelo para obtener métricas sobre su rendimiento, dado a que el modelo claramente conoce dichos patrones y no se obtendrían resultados veraces de cómo se comporta con nuevos datos.

3.6.2 *Método de retención*

MÉTODO DE RETENCIÓN: *Conocido en inglés como "Holdout"*

El MÉTODO DE RETENCIÓN [128] es una mejora del enfoque anterior en el que tal y como muestra la Figura 3.15 un porcentaje de los datos son empleados para entrenar y generar el modelo y los restantes se emplean para analizar su comportamiento con datos no vistos anteriormente por el modelo mediante las métricas de evaluación.

La Figura 3.15 muestra un ejemplo donde el 60 % de los datos que componen la base de datos se emplean para entrenar el modelo, y el 40 % restante para analizar su comportamiento. Estos porcentajes no son fijos, sino que dependiendo de las necesidades y del tamaño de la

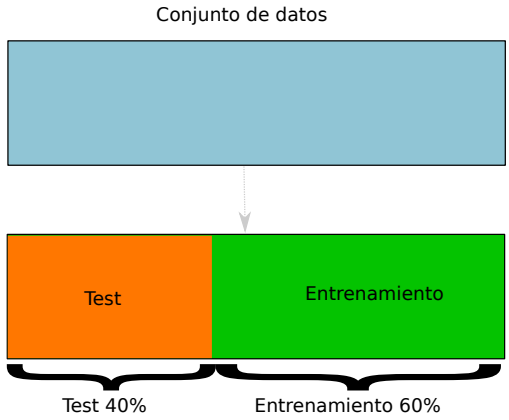


Figura 3.15
Ejemplo de particionado 60 %-40 %.

base de datos interesará destinar más datos para entrenar o más datos para analizar.

3.6.3 *Método de validación cruzada con iteraciones*

La VALIDACIÓN CRUZADA MEDIANTE K ITERACIONES [194] tiene un objetivo similar al método de retención de separar un porcentaje de los datos para entrenamiento y otro para *test*. Sin embargo, existen diferencias al método anterior, dado que se trata de un método iterativo en el que se generan más de un modelo de clasificación tal y como se muestra en la Figura 3.16.

VALIDACIÓN CRUZADA MEDIANTE K ITERACIONES: *En inglés "Cross-validation K-fold"*

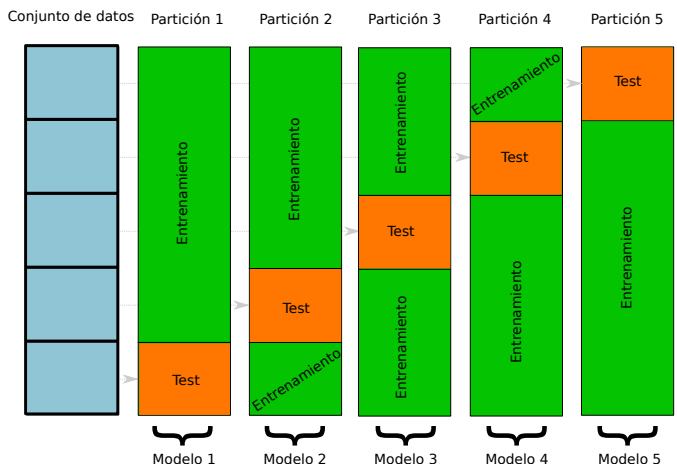


Figura 3.16
Ejemplo de validación cruzada con cinco particiones.

Imaginemos que se desea hacer una validación cruzada con cinco particiones. El primer paso sería dividir el conjunto de datos en cinco partes. A continuación, se escoge la primera parte como datos para realizar la fase de *test* y el resto se utilizan para entrenar y generar el modelo. Tras la generación de este modelo, se coge otra partición de las cinco que se

hicieron al principio, y se emplea para una nueva fase de *test* y el resto para entrenar y generar un nuevo modelo. Al final se generarán tantos modelos como particiones se hayan realizado tal y como muestra la Figura 3.16.

Además, en esta figura se observa un ejemplo en el que se han realizado cinco particiones (*5-fold*), pero se podrían haber realizado un número diferente de ellas. En realidad, en la literatura científica se ha tomado como un estándar el uso de diez particiones (*10-fold*) para bases de datos que tienen un tamaño considerable, y cinco particiones para bases de datos pequeñas.

3.6.4 *Método de validación cruzada dejando uno fuera*

VALIDACIÓN CRUZADA DEJANDO UNO FUERA: *En inglés "Cross-validation K-fold leave one out (LOOCV)"*

El enfoque de VALIDACIÓN CRUZADA DEJANDO UNO FUERA [70] se puede considerar como una caso especial del método de validación mediante k particiones debido a su modo de funcionamiento. Al igual que sucedía con el método de las k particiones, en este caso se vuelven a realizar particiones de entrenamiento y de *test*, pero el tamaño de cada una es fijo. En concreto el tamaño de la partición de *test* es de un patrón y el de entrenamiento de $n-1$ patrones donde n es el número de patrones. En este caso, tal y como muestra la Figura 3.17, se generan siempre un número de modelos igual al total de patrones existentes en la base de datos.

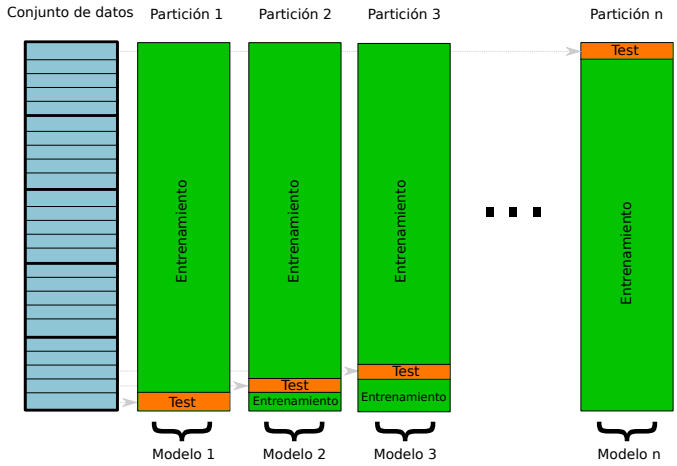


Figura 3.17
Ejemplo de validación cruzada dejando un dato fuera.

La ventaja de este método está en que se tiene un mayor número de comprobaciones independientes considerándose como una validación cruzada completa [91]. Sin embargo, el problema es el alto coste computacional que requiere dado que se generan como se ha comentado anteriormente n modelos, y si estos modelos son computacionalmente pesados de entrenar y además, se tienen muchos patrones, el método puede ser inviable en algunas circunstancias.

3.6.5 *Particiones estratificadas*

Este enfoque a diferencia de los anteriormente descritos no es un método propiamente dicho, sino que es una opción extra a la hora de particionar los datos [139]. Tanto en el método de retención como en el de validación cruzada con k particiones, los datos que se reparten entre el conjunto de entrenamiento y *test* suelen ser escogidos al azar. Sin embargo, esto genera un problema cuando tratamos con bases de datos no balanceadas en las que escoger datos al azar puede hacer que se seleccionen muy pocos datos o incluso ninguno de las clases minoritarias, con lo que la robustez del método frente a fallos estaría comprometida. Para solucionar esto se deben realizar las particiones teniendo en cuenta la cantidad de patrones existentes en cada una de las clases y realizar los conjuntos de entrenamiento y de *test* manteniendo una proporción equivalente al conjunto de datos original.

3.7 RESUMEN DEL CAPÍTULO

Este capítulo ha introducido el concepto de inteligencia artificial y explicado uno de su principales pilares, el aprendizaje automático, mostrando un resumen de los diferentes tipos de algoritmos que existen desde métodos no supervisados básicos hasta etapas más complejas como los procesos de clasificación.

En lo referente a los algoritmos no supervisados se han introducido las principales metodologías que existen y se ha podido observar que las ideas subyacentes de algunos de estos métodos, son la base de funcionamiento de métodos supervisados como el algoritmo de los *K vecinos más cercanos (KNN)* y que estos métodos pueden ser usados para realizar una selección de prototipos o una reducción del espacio de características como un paso previo a las tareas de clasificación como es el caso del *análisis de componentes principales*.

En cuanto al aprendizaje supervisado se ha podido observar la principal diferencia respecto a los métodos que son no supervisados, encontrándonos con la necesidad de que cada patrón tiene que estar previamente etiquetado para lograr conseguir un modelo que sea capaz de predecir nuevos datos sin etiquetar. De este tipo de algoritmos se han proporcionado diferentes metodologías, partiendo desde las más sencillas hasta algunas más complejas como las *Redes Neuronales (NN)*. Además, se ha detallado el mecanismo principal de los metaclasificadores que en realidad son algoritmos supervisados pero que debido a su funcionamiento difieren del estándar de la creación de modelos computacionales para clasificación y que serán analizados en la presente Tesis Doctoral.

Adicionalmente, se han explicado los conceptos fundamentales de los algoritmos basados en aprendizaje profundo por ser una tendencia actual y empleada por otros autores como se puede observar en el Capítulo 4, fuera del ámbito de la clasificación de patrones.

Para finalizar se ha detallado cómo se evalúa el rendimiento de los modelos de clasificación generados, para que pueda existir una comparación cuantitativa entre los diferentes métodos, así como el diseño de experimentos con los patrones disponibles en las bases de datos.

4

REVISIÓN DE PROPUESTAS EXISTENTES

Investigar es ver lo que todo el mundo ha visto,
y pensar lo que nadie más ha pensado
Albert Szent-Györgyi [1]

ÍNDICE

1 Albert Szent-Györgyi (1893 - 1986) bioquímico húngaro descubridor del ácido ascórbico (comúnmente denominado Vitamina C) y Premio Nobel de Fisiología y Medicina en 1937.

4.1 INTRODUCCIÓN

Tras observar el abanico de posibilidades de los diferentes métodos de procesamiento de las imágenes y de los enfoques de la inteligencia artificial, este capítulo realiza un análisis de las diferentes propuestas existentes en la literatura científica. También expone como se emplean algunos de los métodos citados en las secciones anteriores en la temática de la Tesis Doctoral correspondientes a la detección y clasificación de grietas en pavimentos.

Realizando una exploración de las propuestas más recientes se puede observar que el análisis de defectos de las grietas en pavimentos se divide principalmente en dos tareas:

- *Segmentación de grietas*: Tarea que se centra en la discriminación entre las zonas de la imagen que corresponden con el pavimento dañado y las que corresponden con el pavimento sano o en condiciones óptimas.
- *Clasificación de grietas*: Tarea que se centra en dividir en diferentes categorías los defectos de los pavimentos.

De este modo, los autores de la literatura científica han propuesto diferentes metodologías para cada una de estas tareas donde muchos trabajos se centran únicamente en una de ellas (*ej. segmentación de grietas*). Por ello, las diferentes propuestas de los autores se dividen en tres grupos, que cubren las principales metodologías utilizadas para dar solución a las tareas anteriores.

La primera Sección "4.2: Segmentación de grietas" expone todas aquellas propuestas que se han centrado exclusivamente en la tarea de detectar defectos, empleando para ello algoritmos de procesamiento de imágenes y algoritmos de inteligencia artificial, sin llegar a categorizar dichos defectos.

La segunda Sección "4.3: Clasificación de grietas mono-modelo" cubre aquellos trabajos que han realizado un procesamiento de imágenes similar al de la primera tarea, pero que su objetivo final es categorizar los diferentes tipos de defectos utilizando algoritmos de aprendizaje automático en los que únicamente se lleva a cabo la clasificación haciendo uso de un solo modelo.

La tercera Sección "4.4: Clasificación de grietas multi-modelo" detalla aquellas propuestas que persiguen un objetivo similar a los de la segunda sección, pero en las que los autores se han aventurado a realizar clasificaciones con múltiples modelos de clasificación.

4.2 SEGMENTACIÓN DE GRIETAS

La segmentación de grietas es un proceso que ha sido estudiado por varios autores en la literatura científica [119], los cuales emplean algoritmos de procesamiento de imágenes e incluso algoritmos de aprendizaje automático con el objetivo de obtener una división entre el fondo del pavimento y la parte que representa el daño. Para ello, suelen emplear una imagen binaria donde generalmente las grietas suelen ser los píxeles de interés y por tanto los valores que están a 1 (*en imágenes de profundidad de 8-bit el valor correspondiente sería 255*).

En el contexto del uso de métodos de procesamiento de imágenes, *Wang et al.* [188] han propuesto un mecanismo que denominan *componentes potenciales de grietas* (PCRCS) para realizar una segmentación binaria de las grietas transversales y longitudinales. Este mecanismo se lleva a cabo mediante un filtrado de los niveles de intensidad entre el pavimento y las grietas, donde se eliminan todas las componentes que no están conectadas con otras. Una vez han eliminado todas las componentes inconexas, las grietas quedan definida por los elementos que no han sido eliminados. Para realizar una distinción entre el pavimento y la grieta, los autores utilizan lo que ellos mismos denominan *métrica de forma* (SM) obteniendo así una menor tasa de falsos positivos que otros métodos basados en aprendizaje automático como se puede observar en su estudio.

PCRCS: *En inglés "Potential Crack Components"*

SM: *En inglés "Shape Metric"*

Un enfoque diferente lo encontramos en la propuesta de *Amhaz et al.* [9] donde los autores desarrollan un método denominado *selección de la ruta mínima* (MPS) que se basa en la suposición de que el trayecto mínimo de una grieta es el que alcanza una función de coste mínimo de cualquier camino que se pueda encontrar en la imagen. Para ello realizan dos fases, la primera es hallar los píxeles que se consideran como significativos de inicio y final de los caminos dentro de las grietas y detectar todos los caminos entre pares de puntos y su coste mínimo. La segunda etapa consiste en el post-procesamiento eliminando aquellos caminos aislados y con una longitud pequeña que corresponden con ARTEFACTOS en la imagen. Además, los autores estiman el ancho de la grieta empleando el nivel de intensidad de los píxeles vecinos de los caminos hallados.

MPS: *En inglés: "Minimal Path Selection"*

ARTEFACTOS: *Forma alternativa de denominar a aquellos elementos en la imagen que se consideran outliers o ruido con una forma mayor de un único pixel*

Otra propuesta de técnicas de procesamiento de imágenes se puede observar en el trabajo de *Lins et al.* [104] los cuales persiguen el objetivo de desarrollar un sistema capaz de trabajar en tiempo real y que sea capaz de detectar las grietas haciendo uso de un algoritmo de seguimiento de objetos conocido como *filtrado de partículas* [189]. Los autores modifican este algoritmo para poder adaptarlo al seguimiento de píxeles de las grietas en el espacio de color RGB (*véase la Sección 2.2.1*), y mediante un proceso iterativo se almacena la posición de dichos elementos para poder reconstruir la grieta y estimar así su longitud.

Adicionalmente a los métodos de procesamiento de imágenes, algunos autores han empleado junto con ellos algoritmos de aprendizaje auto-

mático supervisado como es el caso del trabajo de *Ai et al.* [6]. Estos autores realizan una segmentación de las grietas empleando dos mapas de probabilidad para determinar si un píxel de la imagen es un componente de una grieta o no. El primer mapa se genera a partir del nivel de intensidad de cada píxel, y el segundo mediante la información de los vecinos de cada píxel a diferente escalas junto a máquinas de vectores soporte (*véase la Sección 3.3.8*).

Otro ejemplo de algoritmo de aprendizaje automático supervisado se encuentra en el trabajo de *Wang et al.* [186], donde los autores emplean un mecanismo de segmentación de la imagen basado en máquinas de aprendizaje extremas (ELM). El uso de este método les permite realizar una inicialización y selección de los parámetros de los nodos de una capa oculta de la red neuronal sin requerir ningún tipo de conocimiento experto y mejorando el tiempo de procesamiento en la etapa de aprendizaje frente a las redes neuronales tradicionales.

ELM: *En inglés: "Extreme Learning Machine"*

Otros autores se han centrado en usar aquellos métodos de la inteligencia artificial enfocados al aprendizaje profundo, como es el caso de *Cha et al. [32]*. Estos autores han propuesto el uso de una red neuronal convolucionada compuesta de ocho capas ocultas las cuales cubren los pasos de convolución, agrupación, rectificación lineal y activación Softmax. Aunque a diferencia de otros autores, ellos emplean la red en grietas de pavimentos de cemento y no realizan una segmentación binaria, sino que extraen diferentes regiones de interés de la imagen donde se encuentra localizadas las diferentes grietas.

Softmax: *Nombre que se le da a la función exponencial normalizada*

Un enfoque similar del uso de redes neuronales convolucionadas se observa en el trabajo de *Cheng et al.* [39] donde emplean una arquitectura denominada U–NET [151]. En esta red, los nodos de capa oculta se encuentran totalmente conectados de tal forma que la mitad de la red se comporta como un codificador de la información de las variables de entrada y transformación de los datos, y la otra mitad un decodificador que obtiene una imagen binaria que representa con valores positivos aquellos píxeles que corresponden con las grietas.

U–NET: *Llamada así por su forma de "U" al representar gráficamente la conexión entre las diferentes capas*

Una solución alternativa al empleo de redes neuronales convolucionadas ha sido propuesto por *Bang et al.* [17] quienes emplean una red residual profunda con transferencia de aprendizaje, compuesta por 152 capas ocultas. De este modo logran realizar la tarea de clasificar a nivel de píxeles si cada uno de ellos corresponden con una grieta o con el pavimento sano, al igual que el trabajo de *Ai et al.* [6].

Además de las propuestas que emplean imágenes visuales bidimensionales, existen autores que han analizado este problema desde un punto de vista diferente utilizando Fusión de datos. El trabajo de *Xu et al.* [192] es un claro ejemplo de ello, donde los autores emplean una combinación de imágenes visuales junto con la información que les proporcionan cuatro láseres lineales. Esto permite a los autores obtener una información espacial del pavimento y proporcionar así una reconstrucción con información de la distancia a los defectos con una precisión milimétrica.

Fusión de datos: *Método que consiste en compactar datos de diferente naturaleza para crear uno nuevo*

4.3 CLASIFICACIÓN DE GRIETAS MONO-MODELO

En la tarea de clasificar los defectos en pavimentos, empleando un único algoritmo de aprendizaje automático orientado a la clasificación, se pueden encontrar algunas propuestas actuales en las que no existe un consenso en el tipo de defectos a analizar. En realidad, las clases objetivo que persiguen los diferentes autores de la literatura científica cubren desde problemas sencillos como la etiquetación binaria entre pavimento sano y pavimento con grietas, hasta la clasificación de diferentes tipos de grietas.

Se pueden encontrar propuestas como el trabajo de *Hoang y Nguyen* [75] quienes llevan a cabo un análisis del comportamiento de diferentes algoritmos supervisados de clasificación. Su objetivo es descubrir el algoritmo más robusto para la discriminación de grietas longitudinales, transversales, malla y aquellos casos donde el pavimento está sano. Para ello emplean una etapa de procesamiento de imagen para obtener una imagen binaria empleando filtros gaussianos, la segunda derivada de la imagen y el cálculo de las integrales proyectivas. En la etapa de clasificación llevan a cabo pruebas con redes neuronales artificiales, máquinas de vectores soporte y un algoritmo denominado RANDOM FOREST [24]. Tras realizar todas las pruebas los autores llegan a la conclusión de que la mejor forma para clasificar este tipo de grietas se consigue mediante las máquinas de vectores soporte.

RANDOM FOREST: *Un método basado en múltiples árboles de decisión*

Otros autores que realizan una clasificación empleando un único modelo de clasificación son *Ibrahim et al.* [79] quienes realizan una comparación entre el algoritmo de *los k vecinos más cercanos* (*véase la Sección 3.3.1*) y su versión *difusa*, que se trata de un método de otra de las ramas de la inteligencia artificial, la lógica difusa. La propuesta de estos autores sólo se centra en la clasificación de grietas longitudinales y transversales realizando un procesamiento de las imágenes para obtener una imagen binaria empleando umbrales seleccionados de forma manual para cada grieta. Al obtener la imagen binaria estos autores realizan una reducción de los atributos de las imágenes a lo que los autores denominan `delta_x` y `delta_y` que son la diferencia entre el punto con el valor máximo en la coordenada de abscisas y el mínimo, para el valor de `delta_x` y de forma análoga, empleando la coordenada de ordenadas, se halla el valor de `delta_y`.

De forma similar a la propuesta anterior, *Ahmadi et al.* [5] desarrolla un sistema con un procesamiento de las imágenes basado en segmentación mediante umbrales de varias etapas, matrices de intensidad y el modelo de color LAB (*véase la Sección 2.2.5*) para la eliminación del ruido y la obtención de la imagen binaria. Tras la obtención de esta imagen, los autores realizan una reducción del espacio de características utilizando datos procedentes de las matrices de intensidad empleadas en la obtención de la imagen binaria, e información estadística obtenida de esta última. De este modo, los autores consiguen un total de ocho atributos principales y catorce adicionales para llevar a cabo la clasificación de grietas transversales y de tipo bloque. En la etapa de

clasificación se utilizan algoritmos de máquinas de vectores soporte, árboles de decisión y *los k vecinos más cercanos*, siendo este último el que mayor rendimiento proporciona.

Por otro lado se encuentran muy pocas propuestas que se basen en la clasificación de los defectos de pavimentos utilizando métodos de aprendizaje profundo, quizás debido a la limitación de estos en cuanto a la necesidad de una enorme cantidad de datos. Un ejemplo se encuentra en el trabajo de *Kim et al.* [88], donde los autores los autores se centran en desarrollar un sistema que sea capaz de distinguir si una imagen, corresponde con un pavimento sano o por el contrario tiene defectos. Para ello emplean dos fases, la primera una segmentación de la imagen original del pavimento mediante el cálculo de *regiones de grietas en cemento* (CCR) en el que se realiza una distinción entre el pavimento de la imagen y las grietas, asumiendo que las zonas claras corresponden al pavimento sano y las zonas más oscuras corresponden a los defectos. Y en la segunda fase correspondiente a la clasificación, los autores realizan una comparación de entre el método SURF [76] y redes neuronales convolucionadas.

CCR: *En inglés: "Concrete Crack Regions"*

SURF: *En inglés: "Speeded Up Robust Features"*

También se pueden encontrar algunas propuestas como el trabajo de *Li et al.* [102] donde los autores emplean imágenes de naturaleza diferente a las tradicionales en dos dimensiones al igual que el trabajo analizado de *Xu et al.* [192] en la sección anterior. En este caso, *Li et al.* emplean un sistema de detección de luz de baja altitud y alcance con vehículos aéreos no tripulados (UAV LiDAR) obteniendo de ellos una nube de puntos en un espacio tridimensional que son empleados como los atributos de entradas de diferentes algoritmos de clasificación entre los que se destacan máquinas de vectores soporte, *random forest*, y un método de clasificación basado en la máxima verosimilitud (MLC) [169]. Con el uso de estos modelos de clasificación llevan a cabo una comparativa del comportamiento de los diferentes algoritmos para la clasificación de defectos tales como grietas, baches, problemas de hundimiento y de desgaste del pavimento. Los autores terminan concluyendo que para la naturaleza de los datos que emplean el algoritmo de clasificación que mejor se comporta es *random forest*.

UAV LiDAR: *En inglés: "Low altitude Unmanned Aerial Vehicle Light Detection And Ranging"*

MLC: *Maximum Likelihood Classification*

Otro ejemplo con datos tridimensionales y el uso de los métodos de aprendizaje automático basados en aprendizaje profundo ha sido propuesto por *Li et al.* [100] donde al igual que en el trabajo de *Kim et al.* [88] se emplean redes neuronales convolucionadas para detectar grietas de tipo longitudinales, transversales, de bloque y de malla. Para ello los autores utilizan las imágenes 3D obtenidas del sistema *PaveVision 3D* [105] en las que se dividen las imágenes en regiones de 512×512 píxeles y son empleadas directamente como entradas de la red neuronal.

4.4 CLASIFICACIÓN DE GRIETAS MULTI-MODELO

En la tarea de clasificar las imágenes segmentadas o imágenes binarias en diferentes tipos de grietas, no se encuentran muchos autores en la literatura científica que hayan realizado propuestas con más de un modelo de clasificación, aunque los pocos existentes muestran una metodología muy interesante.

Uno de los trabajos que emplean varios modelos para la clasificación de las grietas en diferentes tipos, lo encontramos en la propuesta de *Li et al.* [101]. En su investigación los autores han desarrollado un sistema para realizar una clasificación de los defectos en dos tipos, grietas de tipo malla y grietas lineales en las que los autores identifican dos tipos diferentes de estas últimas, las grietas longitudinales y las grietas transversales. Para la fase de detección emplean un algoritmo para corregir la iluminación de la imagen y obtener una segmentación basada en los valores de intensidad de la misma. Tras obtener la imagen binaria, los autores realizan un proceso para reducir los datos de la imagen buscando las componentes conectadas más grandes en la imagen binaria y tomando la componente de área mayor como la representativa del defecto en al imagen. Tras extraer la región mayor, esta queda definida por dos atributos, el porcentaje de píxeles que compone la región y el porcentaje de píxeles respecto al tamaño de la imagen. Estos nuevos atributos se utilizan para alimentar una red neuronal con propagación de los errores hacia las entradas (*véase la Sección 3.3.7*) discriminado de esta forma en una primera fase aquellos defectos que corresponden con grietas lineales o de tipo malla. Para la distinción de las grietas de tipo transversal y tipo longitudinal, los autores emplean una regla de decisión utilizando un tercer parámetro obtenido de la región que representa el ángulo de inclinación respecto al eje de abscisas. Sin embargo, estos autores no contemplan que la superficie de la calzada alquitranada pueda encontrarse libre de grietas, que es el caso más común. Además, exceptuando la regla de decisión emplean en la discriminación entre grietas transversal y longitudinal un modelo de red neuronal que es poco interpretable visualmente.

Otro ejemplo de propuestas que emplean dos modelos para la etapa de la clasificación se encuentra en el trabajo de *Cubero-Fernández et al.* [42], donde los autores han desarrollado un sistema que realiza la clasificación automática del pavimento sano, y del pavimento con grietas longitudinales, transversales y de tipo malla. Para ello emplean algoritmos de procesamiento de imágenes para corregir la iluminación y extraer las integrales proyectivas de la imagen binara de las grietas. Mediante las integrales proyectivas los autores emplean dos algoritmos de clasificación basados en árboles de inducción, en concreto el algoritmo C4.5 (*véase la Sección 3.3.4*). De este modo se genera un sistema de clasificación en dos etapas, empleando un árbol C4.5 para la discriminación de grietas transversales y pavimento sano, y otro árbol para la clasificación de grietas longitudinales y pavimento sano. Una vez que se ha realizado la discriminación en estos dos tipos, la distinción de grietas de tipo malla se lleva a cabo empleando las salidas de los dos modelos

anteriores, utilizando una regla de decisión en la que si la salida de los dos modelos proporcionan como resultado simultáneamente longitudinal y transversal, el resultado es una grieta de tipo malla. Sin embargo, el uso de los dos modelos de clasificación empleados de forma individual, muestran que la tolerancia a fallos no es muy alta obteniendo unos resultados de exactitud (*véase la Sección 3.5*) que se encuentran en torno al 72 % y 67 % para las grietas de tipo transversal y longitudinal respectivamente.

4.5 RESUMEN Y ANÁLISIS DEL CAPÍTULO

Una vez revisadas las propuestas de otros autores de la literatura científica se puede observar que para la clasificación de grietas en pavimentos, existen dos tareas diferentes, la segmentación de las grietas y la clasificación de las mismas. Además, se puede observar que en el caso de la clasificación de las grietas la mayoría de las propuestas realizan una parte de detección previa generando una imagen binaria, que es el objetivo de la primera tarea.

En lo referente a las propuestas orientadas a detección de las grietas, los autores emplean diferentes metodologías desde el uso de algoritmos de procesamiento de imágenes y algoritmos de aprendizaje automático, hasta la combinación de ambos. Se pude observar que también existe un cierto interés en la literatura científica de emplear los métodos de aprendizaje automático para desarrollar esta tarea e incluso emplear métodos de aprendizaje profundo sin que haya que realizar un procesamiento de imágenes previo.

En cuanto a la tarea de clasificación se observa que la mayoría de los autores realizan diversas comparaciones entre algoritmos como máquinas de vectores soporte y redes neuronales y que suelen emplear por lo general un único modelo de aprendizaje automático para realizar esta labor, pero no se encuentran muchas propuestas en las que los algoritmos que se consideran como los mejores para una tarea coincidan en las demás propuestas. Además, algunos autores han intentado tomar la ventaja de utilizar varios modelos de clasificación de forma individual o por etapas para realizar una clasificación de los diferentes tipos de grietas.

Por otro lado, se puede observar en la literatura existente, que el tipo de elemento visual que impera en esta temática corresponde con imágenes en dos dimensiones. Además, algunos autores han intentado reducir el número de características o atributos empleados como entradas para los diferentes algoritmos de aprendizaje, pero estas características están lejos de ser fácilmente interpretables o permitir realizar una clasificación de más de dos tipos de grietas diferente.

Por lo tanto, se puede considerar que las propuestas actuales que se encuentran en la literatura actual:

- Generalmente emplean algoritmos como redes neuronales o máquinas de vectores soporte que convierten a los clasificadores en

CAJAS NEGRAS poco o nada interpretables a diferencia de los árboles de decisión y métodos basados en reglas de inducción, los cuales podrían proporcionar unos resultados igualmente aceptables siendo más interpretables.

CAJAS NEGRAS: *Término empleado para definir una función, modelo o método que toma unas entradas y realiza un proceso con ellas sin que tengamos conocimiento de que se realiza internamente y proporciona unas salidas.*

- No aprovechan la ventaja que podría proporcionar el uso varios algoritmos de aprendizaje automático supervisado empleados al mismo tiempo para realizar una toma de decisión en cuanto a la clasificación de las diferentes grietas, como demuestra la escasez en el número de trabajos que se han podido encontrar y describir en la Sección 4.4.

- No se proporciona una reducción del número de atributos que definen a una grieta que sean interpretables y permitan realizar una clasificación de varios defectos de forma simultánea.

- Por lo general, no se proporciona información por parte de los autores para considerar que las propuestas puedan ser susceptibles de trabajar en sistemas computacionales de recursos limitados en el que se puedan llevar a cabo las tareas de clasificación y detección de defectos.

Por todo ello, los siguientes capítulos de la tesis se centrarán en proponer una metodología para hacer frente a estos puntos ciegos de la literatura científica y demostrar su rendimiento analizando los resultados obtenidos de los experimentos llevados a cabo con diferentes imágenes.

5 METODOLOGÍA

Toda la ciencia no es más que
un refinamiento del pensamiento cotidiano
Albert Einstein [1]

ÍNDICE

1 Albert Einstein (1879 - 1955) físico alemán consagrado como el mayor científico del siglo XX, conocido por ser el padre de la teoría de la relatividad y el teorema de equivalencia entre masa y energía ($E = mc^2$).

5.1 INTRODUCCIÓN

Este capítulo presenta la metodología propuesta para la realización de un *sistema automático para la detección y clasificación de grietas en pavimentos*. Por ello, una vez revisados los principales antecedentes en procesamiento de imágenes y aprendizaje automático, así como las propuestas existentes en la literatura científica en la detección y clasificación de grietas, se plantea un flujo de ejecución secuencial tal y como se puede observar en la Figura 5.1. Nótese que este enfoque de funcionamiento aúna las tareas que se comentaron en la Sección 4.1 para proporcionar un sistema completo.

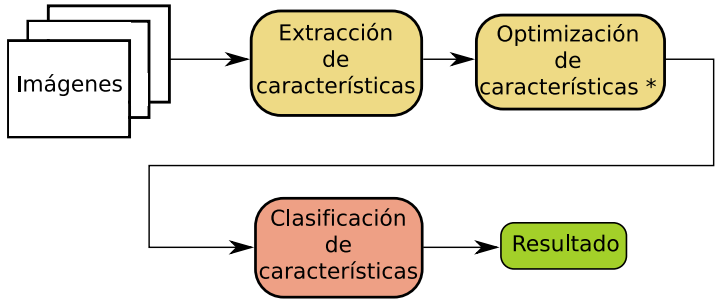


Figura 5.1
Esquema general de la metodología propuesta.

IN-SITU: *Que se lleva a cabo en el mismo sitio donde se realiza la acción. En este caso en el mismo dispositivo donde se capturan las imágenes*

Tal y como se observa en el esquema de la figura anterior, se pueden distinguir dos bloques funcionales, la extracción de características y la clasificación. La primera de ellas, toma como entrada los datos obtenidos desde una cámara (*en el caso de un procesamiento* IN-SITU) o imágenes procedentes de una base de datos (*en un procesamiento externo, fuera del entorno de captura de imágenes*) y se manipulan para obtener una serie de atributos que representen los posibles defectos. En estas imágenes se contemplan dos requisitos. El primer requisito consiste en que las imágenes deben de estar representadas en el modelo de color RGB y de forma matricial, tal y como se contemplaba en la Sección 2.2.1 y en la Expresión 2.1 respectivamente. El segundo requisito es que la cámara debe de estar posicionada de forma cenital al pavimento para poder enfocar correctamente los defectos.

Una vez extraídas las características, éstas serán empleadas a continuación como las entradas del bloque de clasificación que proporcionará un resultado etiquetando la imagen en uno de los cuatros tipos contemplados en la Figura 1.1 al inicio de este documento (*véase la Sección 1.1*).

Además, se puede observar un bloque intermedio entre la extracción de características y la clasificación de las grietas, correspondiente a una etapa cuyo objetivo es la minimización del número de características de tal forma que se mantenga cierto grado de interpretabilidad de los datos.

Por consiguiente, las siguientes secciones de este capítulo detallan los mecanismos empleados en cada uno de los bloques de la Figura 5.1.

5.2 EXTRACCIÓN DE CARACTERÍSTICAS

Considerando que las imágenes son proporcionadas individualmente en el espacio de color RGB, la primera parte del sistema consiste dotar al mismo de una serie de mecanismos que se aplican de forma secuencial para obtener una serie de atributos que representen a un defecto en el caso de que lo haya, sin necesidad de emplear toda la imagen para ello. Además, la manipulación de las imágenes se debe realizar en un tiempo prudencial para poder permitir su ejecución en aquellos casos que se requiera de esta limitación.

Teniendo esto en cuenta, los pasos que ha de seguir el bloque de extracción de características, se pueden representar en forma de pseudocódigo tal y como se muestra en el Algoritmo 1.

Algoritmo 1: Extracción de características

Datos de entrada: $imagen_{RGB}$, K, σ_K, σ_r, α, β
$imagen_{GS} \leftarrow$ **Transformación del Color** ($imagen_{RGB}$);
$imagen_{Mejorada} \leftarrow$ **Mejora de la imagen** ($imagen_{GS}$,K, σ_K, σ_r, α, β);
$imagen_{Bordes} \leftarrow$ **Detección de bordes** ($imagen_{Mejorada}$,K);
$imagen_{Modificada} \leftarrow$ **Modificación morfológica** ($imagen_{Bordes}$,K);
$[\Phi, \Theta] \leftarrow$ **Integrales proyectivas** ($imagen_{Modificada}$);
Resultado: $[\Phi, \Theta]$

Los parámetros de entradas, así como la salidas y el cometido de cada una de las funciones que se encarga de efectuar el Algoritmo 1 se detallan en las siguientes subsecciones las cuales corresponden con cada una de las funciones de dicho algoritmo.

5.2.1 *Trasformación del espacio de color*

Una vez se disponen de las imágenes en formato RGB (*denominadas $imagen_{RGB}$ en el Algoritmo 1*) tal y como se puede observar en la Figura 5.2 a) comienza la cadena de procesamiento del sistema. Si observarnos detenidamente esta figura (*la cual corresponde con una grieta cuyo defecto es de tipo transversal*), o las imágenes que se proporcionaban en la Figura 1.1, se puede detectar que existen elementos cromáticos tales como variación en la iluminación o elementos del entorno como las ramas u hojas dentro de los defectos. Este tipo de elementos realmente carecen de interés para el problema en cuestión que intenta resolver la presente Tesis Doctoral, y por lo tanto no es necesario que se tenga en cuenta esta información. Por ello, la información de la imagen se puede comprimir en un único canal de información empleando una representación en escala de grises (*denominada $imagen_{GS}$ en el Algoritmo 1*) empleando la Ecuación 2.3.

Esta nueva representación tal y como se puede observar en la Figura 5.2 b), no modifica en exceso la percepción visual de la imagen dada la naturaleza de las imágenes, puesto que suelen tener por lo general tonos grises en el pavimento, y permite como se ha comentado antes

mimetizar la información de los elementos como la iluminación o suciedad con el entorno. Además, puesto que se pretende que el sistema pueda emplearse en un hardware con recursos computacionales limitados, la utilización de un único canal divide prácticamente entre tres, el tiempo de procesamiento utilizado en la mayoría de los algoritmos que serán empleados en las sucesivas secciones.

Figura 5.2
Resultado de aplicar una transformación del color a una grieta transversal. a) Imagen de grieta transversal en RGB. b) Imagen de grieta transversal en escala de grises.

5.2.2 *Mejora de la imagen*

Este paso del algoritmo toma como entrada la imagen monocromática en escala de grises tras su conversión desde un modelo RGB de la sección anterior, y aplica varios filtros de forma consecutiva a la imagen con el objetivo de mejorar su visualización enfatizando las características correspondientes a las grietas.

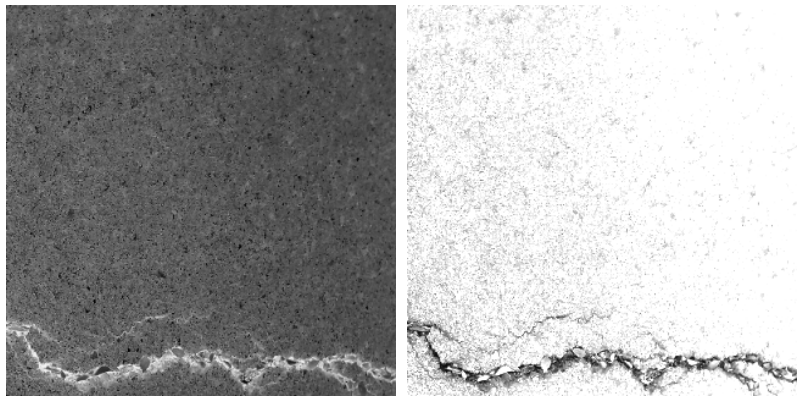


Figura 5.3
Resultado de aplicar la transformada logarítmica a una grieta transversal. a) Negativo de la imagen de la grieta. b) Resultado de la transformada logarítmica.

Por tanto, como pueden existir variaciones en la iluminación a la hora de tomar las capturas del asfalto, el primer algoritmo que se utiliza para mitigar este efecto enfatizando las zonas oscuras, es la transformada logarítmica. Para su cálculo, tal y como se detalló en la Sección 2.4.4, es necesario obtener el negativo de la imagen y luego emplear la Ecuación 2.27 teniendo en cuenta los parámetros, α, β y K que toma como entrada esta función en el Algoritmo 1. Estos parámetros, son los mismos que se detallaron en la Ecuación 2.27. En la Figura 5.3 a) y b), se puede observar el cálculo del negativo de la imagen y el resultado de la transformada logarítmica respectivamente para la grieta transversal de la Figura 5.2 b). Hay que destacar que para que quede clara su interpretación, el negativo y el logaritmo se han realizado como dos pasos independientes, pero que para optimizar el tiempo de cómputo empleado por la transformada logarítmica, el negativo se puede calcular al mismo tiempo que se calcula el valor del logaritmo para un píxel dado tal y como se detalló en la Ecuación 2.27. Además, puesto que el cálculo de la imagen en escala de grises de la sección anterior no implica píxeles vecinos, es posible realizar su cálculo al mismo tiempo que se calcula el negativo. Sin embargo, no se ha detallado así en la metodología, para que se puedan entender conceptualmente los pasos que sigue el método propuesto, aunque si se ha realizado de manera conjunta en la implementación del sistema.

Una vez se ha llevado a cabo la transformada logarítmica, el resto de parámetros que se emplean en esta función tal y como se puede observar en el Algoritmo 1, corresponden por su nomenclatura al filtrado bilateral (*Véase la Sección 2.16*). El uso de este método de mejora de la imagen, nos permite realizar un difuminado del fondo de la imagen (*el pavimento*) y mantener intactos los bordes (*las grietas*) tal y como puede observarse en la Figura 5.4 b). Además, como se aprecia en dicha figura, este método permite eliminar o difuminar la mayoría de los píxeles que pertenecen a la capa asfáltica, los cuáles, no son útiles para la adquisición de las grietas. De este modo, se evita tener que emplear para ello métodos de de eliminación de artefactos como emplean las propuestas de otros autores [9, 101] citadas en el Capítulo 4.

5.2.3 *Detección de bordes*

Tras obtener la $imagen_{Mejorada}$ con los métodos de la sección anterior, ésta es usada para detectar los bordes de las grietas y compactar aún más la información de la imagen. Para ello, y con el objetivo de detectar dichos bordes, se emplea el algoritmo de Canny (*véase la Sección 2.5.5*).

No obstante, el método de Canny necesita dos umbrales (th_1, th_2) para su correcto funcionamiento, y tal y como se puede observar en el Algoritmo 1 la función de esta parte del procesamiento no recibe ninguno de estos parámetros. Esto se debe a que estos parámetros no son sencillos de ajustar manualmente para cada imagen y además, es posible seleccionar de forma dinámica dichos parámetros sin necesidad de fijarlos previamente. Para ello se emplea la Ecuación 2.21 correspondiente al método de Otsu que permite determinar de forma automática

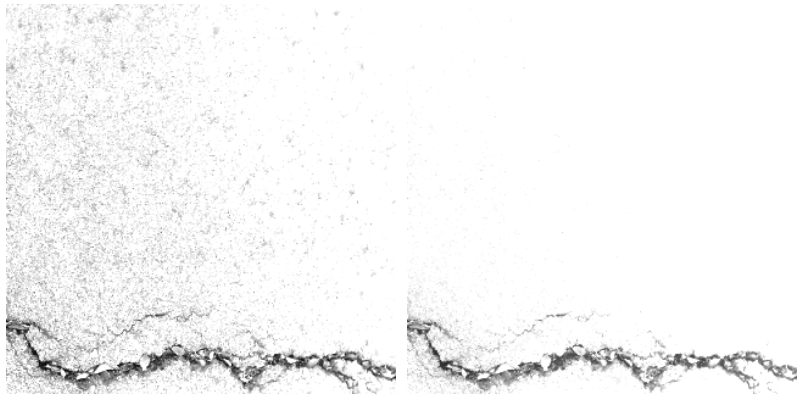


Figura 5.4
Resultado de aplicar el filtrado bilateral a la imagen devuelta por la transformada logarítmica. a) Imagen de la transformada logarítmica. b) Imagen filtrada.

un valor de umbral (t) para llevar a cabo la segmentación de una imagen monocromática.

Una vez calculado el umbral mediante el método de Otsu, los valores de los umbrales de Canny pueden seleccionarse como $th_1 = 0.5 \cdot t$ y $th_2 = t$. De este modo, se puede obtener un resultado similar al de la Figura 5.5 b) en la que se pueden observar los bordes extraídos tanto del exterior de las grietas como de su interior.

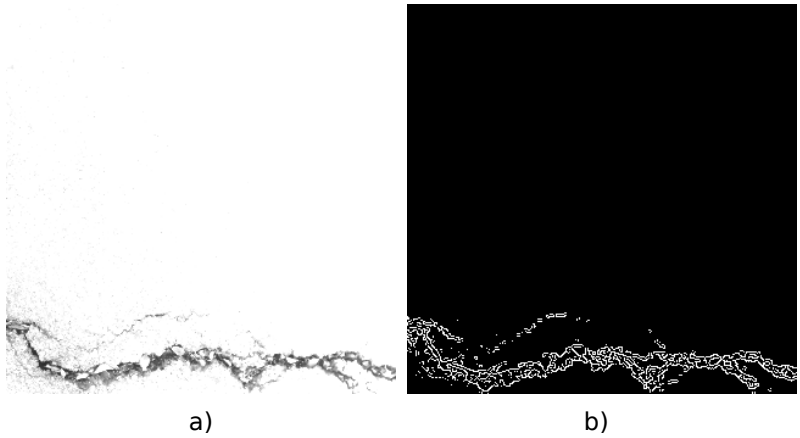


Figura 5.5
Resultado de aplicar el método de Canny a una grieta. a) Imagen procedente del filtrado bilateral. b) Imagen resultante al aplicar el método de Canny a la imagen del filtrado bilateral.

5.2.4 *Modificación morfológica*

Obtenidos todos los bordes de las grietas, se puede observar que existen algunas partes de ellas que no están rellenas en la Figura 5.6 a). Por ello, y con el fin de rellenar estos "huecos" al igual que sucedía con el escudo de informática de la Figura 2.28, en esta parte del Algoritmo 1

se emplea el operador morfológico de cierre. El efecto de aplicar este operador se puede observar en la Figura 5.6 b).

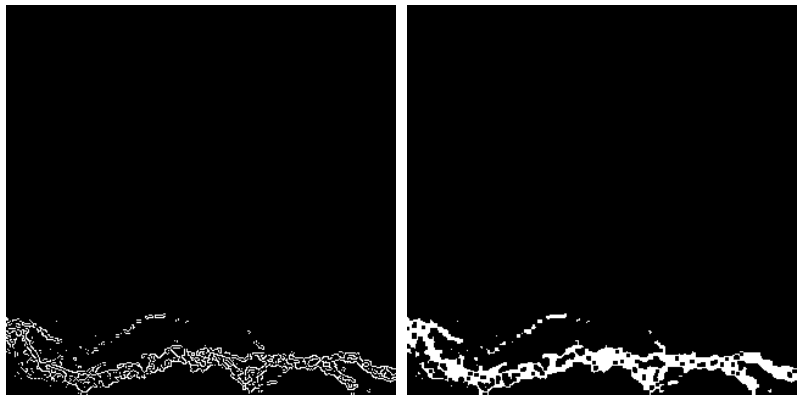


Figura 5.6
Resultado de aplicar operador morfológico de cierre al resultado de Canny en una grieta transversal. a) Resultado devuelto por el método de Canny. b) Resultado tras aplicar el operador morfológico de cierre.

5.2.5 *Integrales proyectivas*

El último paso que debe de realizar el bloque de extracción de características mostrado en el Algoritmo 1 es reducir la imagen obtenida de todos los pasos anteriores ($imagen_{Modificada}$) a una nueva representación donde se destaque la información global de la imagen. Para ello, se calculan las integrales proyectivas horizontales y verticales (*véase la sección 2.7.2*), las cuales analizan la información de la información de la imagen binaria por filas y por columnas como se muestra en la Figura 5.7.

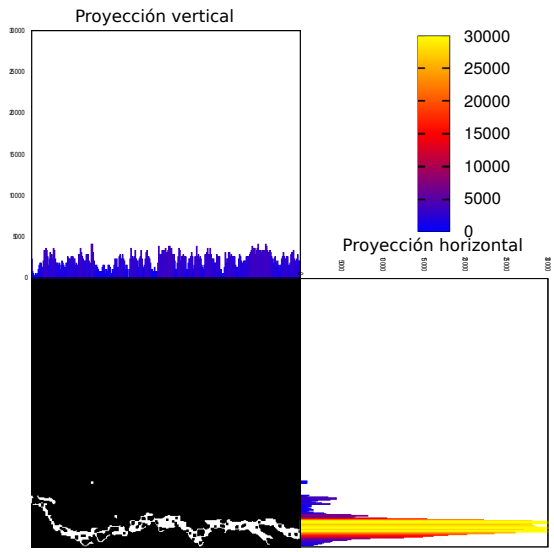


Figura 5.7
Integrales proyectivas verticales y horizontales para una grieta transversal.

Tras el cálculo de las integrales proyectivas, el número de atributos que originalmente componían una imagen ($imagen_{RGB}$) $\mathrm{N}(\mathtt{columnas}) \cdot \mathrm{N}(\mathtt{filas}) \cdot 3$ (*siendo* $\mathrm{N}()$ *el número de elementos y* 3 *el número de canales*), se ha compactado a un número menor pasando a ser igual a $\mathrm{N}(\mathtt{columnas}) + \mathrm{N}(\mathtt{filas})$. Además tal y como se muestra en el Algoritmo 1 la nueva representación ya no corresponde con una imagen, sino con una estructura compuesta por los elementos de ambas integrales proyectivas ($[\Phi, \Theta]$).

5.3 OPTIMIZACIÓN DE CARACTERÍSTICAS

Mediante el cálculo de las integrales proyectivas del bloque anterior, se podría llevar a cabo una clasificación de los diferentes tipos de grietas. Sin embargo, aunque los valores de estas proyecciones como se observa en la Figura 5.8 sean más interpretables que los valores locales de los píxeles de forma individual, ¿qué se puede interpretar de una imagen que tenga un valor en la proyección horizontal 100 igual a 3200? ¿Y un valor igual a 4560 en su integral vertical 200?. Evidentemente dada la definición de las integrales proyectivas la respuesta es clara, debido a que corresponde con el valor acumulado. No obstante, para definir completamente un defecto habría que considerar todos los valores en ambas integrales.

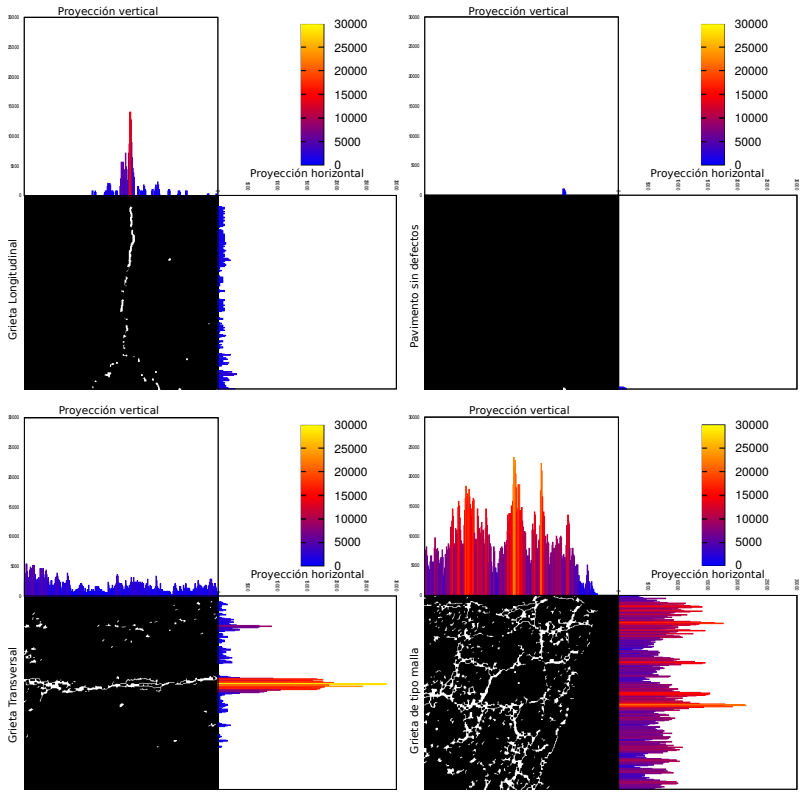


Figura 5.8
Efecto de las integrales proyectivas para los diferentes tipos de clases analizadas en este trabajo.

Debido a esto, y que el número de valores depende del tamaño de la imagen, se podría considerar realizar una reducción de datos por ejemplo mediante *el método de análisis de componentes principales* detallado en la Sección 3.2.2. Esto permitiría considerar un número de atributos reducidos manteniendo la variabilidad de los datos. No obstante, este método es dependiente del tamaño de la imagen y proporciona una reducción de características diferente dependiendo del tamaño. Para solucionar esto se podría fijar el número de características, pero nos encontramos con dos limitaciones. La primera limitación es que para diferentes tamaños, el fijar un número específico de características no garantiza la misma variabilidad en los datos. Y segundo y más importante, es que los datos reducidos generan nuevos atributos dependiente de los originales mediante una transformación del espacio en el que dejan de ser interpretables.

Por ello, es necesario buscar una reducción alternativa que permita la interpretabilidad de los datos al mismo tiempo que se elimina la componente de de la localidad del defecto y sea empleable para imágenes de cualquier tamaño. De este modo, si se observa en la Figura 5.8, se detecta que en las grietas longitudinales y transversales, existe un valor máximo acumulado en las integrales proyectivas verticales y horizontales respectivamente. Y en el pavimento sano apenas existen valores al contrario de lo que sucede en las grietas de tipo malla donde se encuentran muchos valores altos en ambas proyecciones. Por lo tanto, el valor máximo acumulado para ambas integrales es un atributo fundamental para la interpretación de los defectos (*maxV, maxH*), aunque por sí solos no son suficientes debido a que la existencia de los valores altos en las grietas de tipo malla podrían acarrear un conflicto. Por este motivo además de emplear los valores máximos se calculan la diferencia de los mismos respecto a la media de los valores (*dV,dH*), como una medida de magnitud que cuantifique si realmente el valor máximo es representativo de la información del defecto o si por el contrario es muy similar a los valores que están por debajo de él. Con esta reducción mediante el máximo y su diferencia respecto a la media, el número de atributos para clasificar una imagen se simplifica a cuatro (*[maxV, dV, maxH, dH]*).

Con este cuarteto de atributos se ha conseguido la interpretabilidad de los datos, reducir a un número de atributos inferior respecto a las integrales proyectivas, y anular el efecto de la posición de los defectos. No obstante, aún permanece el problema de imágenes de diferente tamaño, con lo que es necesario un paso más que consiste en normalizar los datos al rango $[0-1]$. Mediante la normalización, un mismo modelo de clasificación se puede emplear para una imagen de un tamaño arbitrario sin necesidad de llevar a cabo un entrenamiento de los datos específico para este nuevo tamaño de imágenes.

Los pasos para el cálculo de los valores máximos y las desviaciones que debe de seguir este bloque de optimización se detallan en el Algoritmo 2, donde *N()* corresponde con el número de elementos:

Algoritmo 2: Optimización de características

Datos de entrada: $[\Phi, \Theta]$
maxV=Φ_0;
mediaV=0;
maxH=Θ_0;
mediaH=0;
for *i=1: N(Φ)* **do**
    mediaV= mediaV + Φ_i;
    if Φ_i>*maxV* **then**
        maxV=Φ_i;
mediaV = $\frac{\text{mediaV}}{N(\Phi)}$;
dV=maxV-mediaV;
for *i=1: N(Θ)* **do**
    mediaH= mediaH + Θ_i;
    if Θ_i>*maxH* **then**
        maxH=Θ_i;
    mediaH = $\frac{\text{mediaH}}{N(\Theta)}$;
    dH=maxH-mediaH;
Resultado: $[\frac{\text{maxV}}{255 \cdot N(\Theta)}, \frac{\text{dV}}{255 \cdot N(\Theta)}, \frac{\text{maxH}}{255 \cdot N(\Phi)}, \frac{\text{dH}}{255 \cdot N(\Phi)}]$

Cabe destacar que los pasos el Algoritmo 2 están separados del cálculo de las integrales proyectivas para que conceptualmente se comprenda el origen de los parámetros reducidos. Sin embargo, el cálculo de los valores máximos, así como el acumulado de la media, se pueden realizar al mismo tiempo que se calculan las integrales proyectivas, optimizando así el tiempo empleado en su cálculo.

5.4 CLASIFICACIÓN DE GRIETAS

El último bloque del sistema consiste en la clasificación de las grietas en cada uno de los tipos contemplados en la Figura 1.1. Para realizar esta tarea, se propone realizar la clasificación mediante un ensemble de modelos (*véase la Sección 3.3.9*) dado que es un enfoque que como se observó en el Capítulo 4 no ha sido muy empleado en esta temática y puede presentar unos resultados eficaces tal y como lo han sido en otros ámbitos de la literatura científica [19, 56, 60, 195] fuera de la temática de la presente Tesis Doctoral.

Sin embargo, tal y como se detalló en la Sección 3.3.9, la confianza de un ensemble de modelos basado en el voto de la mayoría, depende del número de modelos empleados, y en este trabajo se busca que la clasificación al igual que la extracción de características que se detallaba en la Sección 5.2, pueda llevarse a cabo en sistemas de recursos computacionales limitados. Por ello, el número de modelos empleados importa dado que el número de operaciones (*si los modelos tiene alguna componente de regresión*) o consultas de las reglas generadas, lleva asociado un consumo de recursos.

En este sentido, podría llegarse a la conclusión de que el número mínimo para componer un ensemble de modelos basado en el voto de la mayoría es dos. Sin embargo, eso daría lugar a conclusión errónea porque aunque emplear dos modelos sigue la filosofía de los ensembles de usar varios modelos de clasificación, este número, en el voto de la mayoría es insuficiente debido a que no es posible alcanzar dicha mayoría en la toma de la decisión. Para visualizar esta justificación de forma gráfica basta con representar de nuevo la Ecuación 3.16 para el caso de un único modelo, dos modelos y tres modelos como se observa en la Figura 5.9. En esta figura, en el comportamiento de la línea discontinua azul, que corresponde con dos modelos de clasificación, sólo se alcanzaría una buena clasificación en el caso ideal de tener modelos individuales que sean clasificadores perfectos y sin fallos, y en tal situación el uso de un ensemble de modelos carecería de sentido. Por lo tanto, queda justificado que el número mínimo para componer un ensemble mediante el voto por la mayoría debe de ser necesariamente tres.

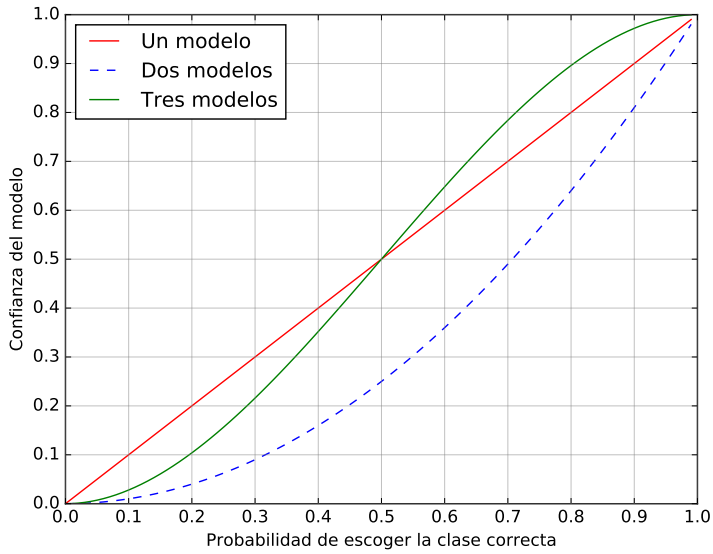


Figura 5.9
Confianza del ensemble para un modelo, dos modelos y tres modelos.

Los algoritmos concretos que compondrán el ensemble serán determinados en el siguiente capítulo cuando se determine el comportamiento de los modelos individuales. De entre los algoritmos que se comprobarán se escogerán aquellos que mejores resultados proporcionen individualmente. Debido a que tenemos cuatro clases (*pavimento sano, grietas longitudinales, grietas transversales, y grietas de tipo malla*) cabe la posibilidad de que los tres modelos escojan clases diferentes entre sí, y por lo tanto sería imposible tomar una decisión por mayoría. Por ello, en este caso especial, se escogerá la clase que indique el algoritmo que individualmente proporciona mejores resultados de clasificación. Para el resto de casos, se podría realizar la votación mediante una codificación similar al del Algoritmo 3, donde se considera que las etiquetas de las

clases que proporcionan los modelos individuales (*R_A*) son valores numéricos (*o:pavimento sano, 1:grietas longitudinales, 2:grietas transversales, y 3:grietas de tipo malla*).

Algoritmo 3: Voto de la mayoría

Datos de entrada: *Resultados de los algoritmos de aprendizaje (R_A)*

```
R ← 0;
i ← 0;
Num_Algoritmos ← 3;
Num_Clases ← 4;
Votos_Clase_i;
while i < Num_Clases do
    for j=0: Num_Algoritmos do
        if R_A_j Igual a i then
            Votos_Clase_i ← Votos_Clase_i + 1;
Num_votos ← Votos_Clase_0;
i ← 1;
while i < Num_Clases do
    if Num_votos < Votos_Clase_i then
        R ← i;
        Num_votos ← Votos_Clase_i;
```

Resultado: R;

5.5 RESUMEN DEL CAPÍTULO

Este capítulo ha mostrado la composición de un sistema para la detección y clasificación de grietas en pavimentos considerando tres tipos de defectos así como cuando el pavimento está en buenas condiciones y no tiene ninguna grieta apreciable. Por ello, el sistema propuesto se divide en tres módulos: extracción de características, optimización de características y la clasificación de características.

El primer módulo extrae las características mediante algoritmos de procesamiento de imágenes desde una perspectiva basada en la localidad de los píxeles hasta la adquisición global de atributos que permitan realizar una clasificación de los diferentes tipos de imágenes. Durante los algoritmos de procesamiento de imágenes se ha podido observar cómo se iba eliminando la información superflua de los posibles defectos y qué métodos o pasos se pueden llevar a cabo al mismo tiempo con el objetivo de minimizar el tiempo de cómputo.

El segundo módulo ha proporcionado una representación de los atributos que les otorga la característica de interpretabilidad considerando sólo cuatro valores a diferencia de los que tenía la imagen originalmente o los adquiridos en el primer módulo mediante las integrales proyectivas. Además, esta representación tal y como está definida, permite si se quisiese, emplear imágenes de cualquier tamaño dado que se elimina la relación espacial de los defectos.

Por último, en el módulo de clasificación, se ha propuesto un enfoque basado en ensembles de modelos considerando sólo tres modelos. Esto puede permitir aprovechar las ventajas frente a modelos de clasificación individuales o en múltiples etapas como los que se detallaron en el Capítulo 4.

De este modo, el siguiente capítulo de la presente Tesis Doctoral (*véase el Capítulo 6*), analizará el rendimiento de la metodología propuesta, y definirá qué modelos concretos componen el ensemble, basándose en los modelos individuales que mejor se comporten individualmente en la clasificación de diferentes imágenes.

6 EXPERIMENTACIÓN Y DISCUSIÓN DE RESULTADOS

Toda verdad es fácil de comprender una vez que ha sido descubierta, el problema es descubrirla.
Galileo Galilei [1]

ÍNDICE

1 Galileo Galilei (1564 - 1642) físico y astrónomo italiano considerado uno de los padres de la revolución científica, así como el inventor del telescopio y descubridor de las leyes que rigen el movimiento de los péndulos.

6.1 INTRODUCCIÓN

Este capítulo muestra y analiza el rendimiento de las propuestas realizadas en el capítulo anterior con el objetivo de conseguir un *sistema automático para la detección y clasificación de las grietas en los pavimentos*. De este modo, y dado que el sistema se compone de varias fases o tareas (*véase la Figura 5.1*), este capítulo se divide en las siguientes secciones:

- *Recursos utilizados*: En esta sección se detalla el hardware de recursos limitados empleado para realizar los experimentos así como los datos con los que se han llevado a cabo los mismos.
- *Extracción de características*: En esta sección se comprueba el funcionamiento de los algoritmos de procesamiento de imágenes para la extracción de características.
- *Clasificación de características*: Una vez extraídas las características, se realizan pruebas de los diferentes algoritmos explicados la Sección 3.3, y se comparan sus resultados frente a un ensemble de tres modelos.
- *Clasificación mediante características optimizadas*: Esta sección se centra en el análisis de los resultados de la reducción de características conseguida mediante el análisis de componentes principales (*véase la Sección 3.2.2*) y la reducción propuesta (*véase la Sección 5.3*) empleando el mejor modelo obtenido en la sección de *clasificación de características*.
- *Comparación con propuestas existentes*: Esta última sección del capítulo analiza las diferencias existentes de emplear la metodología propuesta en esta Tesis Doctoral frente a otras propuestas similares en el campo de la clasificación de grietas en los pavimentos.

6.2 RECURSOS UTILIZADOS

Para llevar a cabo el análisis de los experimentos de la metodología propuesta en el Capítulo 5, se hace uso tanto de un sistema de recursos computacionales limitados, así como de una base de datos.

SoC: *En inglés: "System on Chip"*

En cuanto al hardware usado, se ha escogido el SoC *NVIDIA Jetson Nano* [130]. El uso de esta plataforma hardware permite analizar si la metodología propuesta en el Capítulo 5, la cual se componía de las tareas de procesamiento y clasificación de imágenes, puede aplicarse en dispositivos de este tipo. Las principales características del *SoC Jetson Nano*, son las siguientes:

- Procesador: ARM A57 (cuatro núcleos) [11] a una frecuencia de 1,43 GHz.
- Memoria de acceso aleatorio: 4 GB LPDDR4.
- GPU: Arquitectura NVIDIA Maxwell [129] con 128 núcleos. 4 USB 3.0

- Tamaño (largo x ancho x alto): 10 cm x 8 cm x 2,9 cm.
- 4 x USB y 2 x conexiones MIPI CSI-2 [116].

En lo referente a la base de datos utilizada, como se ha podido observar en el Capítulo 4 existen dos tareas, la detección y la clasificación, y los autores emplean bases de datos para validar sus metodologías. Sin embargo, en los trabajos de detección, los autores no proporcionan la etiqueta de clase, y en la tarea de clasificación, los autores no suelen poner a disposición del público (*al menos a fecha de la escritura de este documento*) las bases de datos con las etiquetas de clase, además de no incorporar todos los tipos de imágenes analizados en esta Tesis Doctoral. Debido a este motivo, se propone utilizar las siguientes bases de datos (*véase la Tabla 6.1*) para analizar la metodología expuesta en el Capítulo 5:

- *Base de datos A*: Esta base de datos está compuesta por las imágenes de la propuesta de Cubero-Fernandez et al. [42] (*véase la Sección 4.4*), donde sí se proporciona una etiqueta de clase para cada imagen. Esta base de datos está balanceada, y se compone de patrones de tipo longitudinal, malla, transversal y superficie sin defectos como se observa en la *Tabla 6.1*. Además, las imágenes tienen una resolución de 320x320, con una iluminación uniforme de la superficie del asfalto, una anchura de las grietas similar en todas las imágenes, y una superficie libre de elementos externos como arena u hojas.
- *Base de datos B*: La segunda base de datos consiste en la fusión de la *base de datos A* junto con una nueva base de datos. Esta nueva base de datos se compone de 1856 imágenes recolectadas de diferentes tramos de calzadas alquitranadas de las ciudades de *Córdoba* (*España*) y *Oslo* (*Noruega*), y etiquetadas manualmente por el autor de la presente tesis. Las características más detalladas de estas nuevas imágenes se listan a continuación:
  - Las imágenes de las superficies alquitranadas presentan una iluminación y tonalidades diferentes del pavimento, con el objetivo de comprobar la invariabilidad de la metodología propuesta frente a estas características.
  - Se han incorporado grietas con humedad, arena, vegetación incrustada, o incluso pequeñas ramas y hojas para analizar el comportamiento del sistema propuesto con elementos del entorno real.
  - En las imágenes que contienen grietas, la anchura y longitud de las mismas, es aleatorio y no uniforme tal y como se encuentran en un escenario real.
  - En concordancia con las imágenes de la *base de datos A*, las imágenes de la nueva base de datos tienen una resolución de 320x320, proporcionando detalles suficientes para observar los defectos y los elementos del entorno. Además, las imágenes han sido adquiridas a una altura de aproximadamente 1.5 metros de la superficie de pavimento en posición cenital al mismo.

  - Esta base de datos se encuentra disponible de forma libre a disposición de la comunidad científica [178].

- *Base de datos C*: La última base de datos consiste en el aumento del número de patrones de la *base de datos B* de forma artificial empleando métodos de aumento de datos [162] usados normalmente en el aprendizaje profundo para la adquisición de patrones (*véase la Sección 3.4*). En concreto, se han realizado ocho desplazamientos de 30 píxeles en cada imagen. Los cuatro primeros desplazamientos en sentido positivo del eje "X" y los cuatro restantes en sentido positivo del eje "Y". Esto nos permite analizar el comportamiento de los clasificadores y la metodología propuesta con la presencia de una base de muchos patrones, desbalanceada y defectos localizados espacialmente en posiciones diferentes de la imagen.

Tabla 6.1
Número de imágenes que componen las diferentes bases de datos utilizadas.

Base de Datos	Grietas de tipo Malla	Grietas de tipo Longitudinal	Grietas de tipo Transversal	Pavimento sin Grietas	Total
A	100	200	200	100	600
B	380	590	558	928	2456
C	3040	4720	4464	7424	19648

6.3 EXTRACCIÓN DE CARACTERÍSTICAS

El primer paso del sistema propuesto tal y como se detalló en el Capítulo 5, consistía en la eliminación de todos los elementos que no aportan información relevante de las grietas y extraer al mismo tiempo las características que definan a las mismas en caso de que existan. Para ello, como se detalló en el Algoritmo 1, se emplean diferentes métodos de procesamiento de imágenes algunos de los cuales necesitan de una serie de parámetros para poder realizar su labor. Entre los métodos que necesitan estos parámetros se destaca la *transformada logarítmica*, el *filtrado bilateral* y el *operador morfológico de cierre*.

La selección de los parámetros para estos algoritmos se ha llevado a cabo de forma empírica mediante prueba y error, empleando aproximadamente unas 50 iteraciones hasta determinar la combinación óptima de los valores. El listado definitivo de cada uno de los parámetros se encuentra detallado en la siguiente lista:

- Transformada logarítmica: K=$3x3$, $\alpha = 0.9$, $\beta = 1.1$
- Filtrado bilateral: K=$3x3$, $\sigma_K = 35$, $\sigma_r = 16$
- Operador de cierre: K=$3x3$

Con el objetivo de comprobar el tiempo de ejecución y analizar si es posible ejecutar el método de procesamiento de las grietas *in-situ* en

el dispositivo de recursos limitados, se ha obtenido el tiempo medio de cada uno de los pasos del Algoritmo 1 tal y como se muestra en la Tabla 6.2, para un sub-conjunto de 200 imágenes de la *base de datos B*. Los resultados de la segunda columna en esta tabla, arrojan que el método propuesto podría procesar cada imagen en 0.0912 segundos, o lo que es equivalente a una tasa de 10 fps. Además, se puede observar en la misma columna que el paso que más tiempo consume (*marcado en rojo*) es el filtrado bilateral. Sin embargo, tal y como hemos definido este paso del filtrado bilateral en la metodología, es posible optimizar su tiempo de cómputo aprovechando las características del dispositivo utilizada. Dicha optimización consiste en migrar la ejecución de la CPU, a la GPU. Realizando este cambio de contexto el método propuesto es capaz de trabajar a aproximadamente 40 fps.

Tabla 6.2
Resultados de tiempos para la extracción de características en SoC NVIDIA Jetson Nano.

Método	Tiempo (s)	Tiempo optimizado (s)
Cambio a gris, Negativo, Logaritmo	0.0162	0.0162
Filtrado Bilateral	**0.0734**	0.0074
Método de Otsu	0.0004	0.0004
Método de Canny	0.0006	0.0006
Operador de cierre	0.0003	0.0003
Integrales proyectivas	0.0003	0.0003
Tiempo total	**0.0912**	**0.0252**

En lo referente a los resultados visuales del procesamiento, la Figura 6.1, la Figura 6.3, la Figura 6.2 y la Figura 6.4 muestran la imagen final de la extracción de características para las grietas longitudinales, transversales, pavimento sin defecto y el pavimento con grietas de tipo malla respectivamente (*las integrales proyectivas aunque se pueden representar de forma gráfica se han omitido para facilitar la interpretación visual en diferentes imágenes del mismo tipo*).

Se puede observar en la Figura 6.1 y la Figura 6.3 que existen algunos patrones en los que las grietas tienen presencia de vegetación o de tierra, y que con los pasos del método propuesto esa información se descarta o se mimetiza con la forma de la grieta. Del mismo modo, se aprecian pavimentos en los que la capa asfáltica está compuesta de dos materiales diferentes o colores y que el método no tiene problemas en detectar las grietas a través de ellos. Además, en las grietas de tipo malla que aparecen en la Figura 6.4 se observan imágenes con una iluminación diferente entre los diferentes patrones y en algunos casos las grietas son visualmente poco apreciables, aunque el método es capaz de obtener las grietas de forma aceptable en estos casos. Sin embargo, de forma general, se aprecia que se han detectado pequeños artefactos o elementos que corresponden con el pavimento y no con un defecto, los cuales como se verá en las secciones siguientes no tienen un gran impacto en la clasificación, pero que bien se podrían eliminar con pasos adicionales y su respectivo impacto en la velocidad de procesamiento.

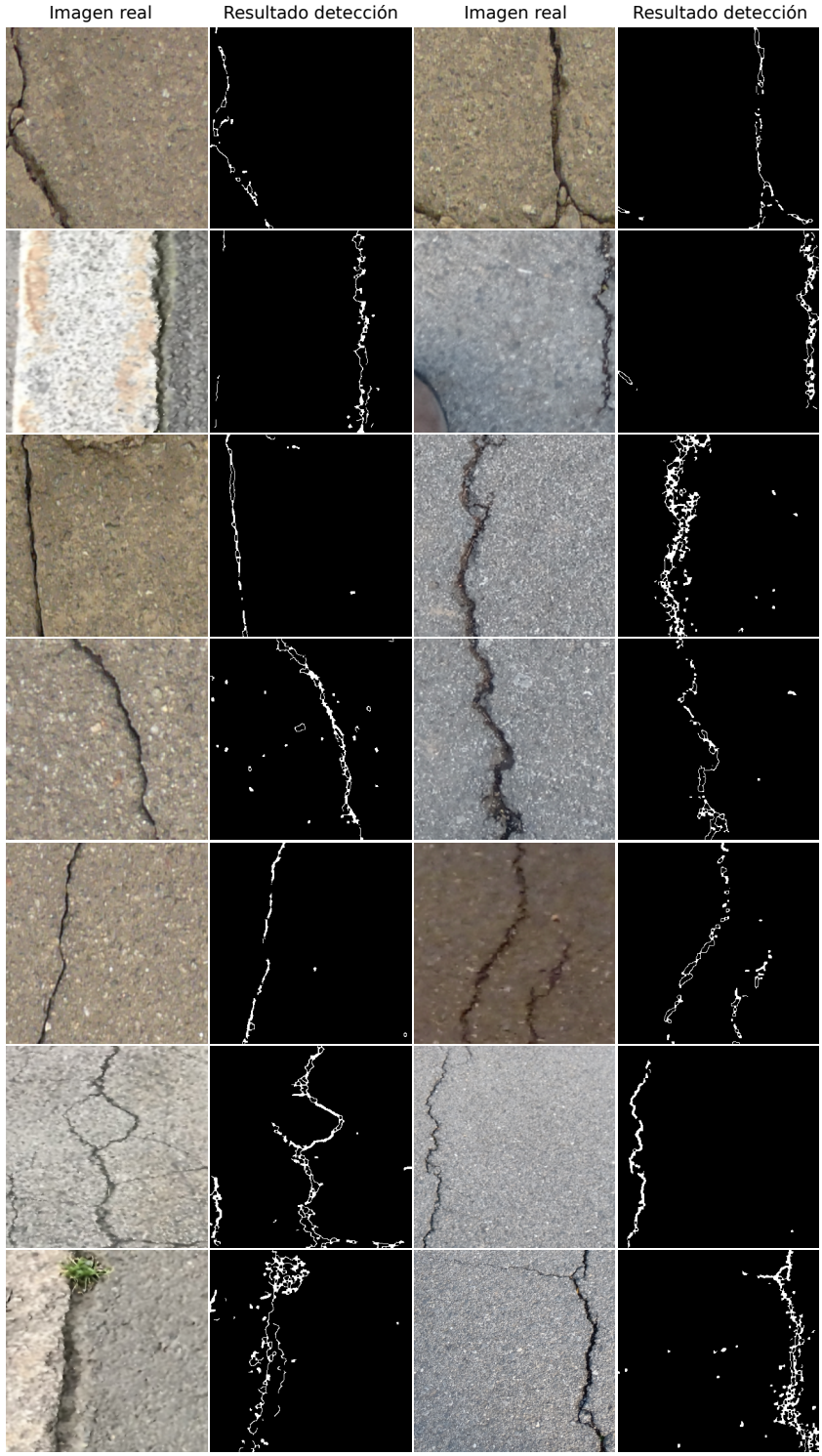


Figura 6.1
Resultados extracción de imagen binaria de varias grietas longitudinales.

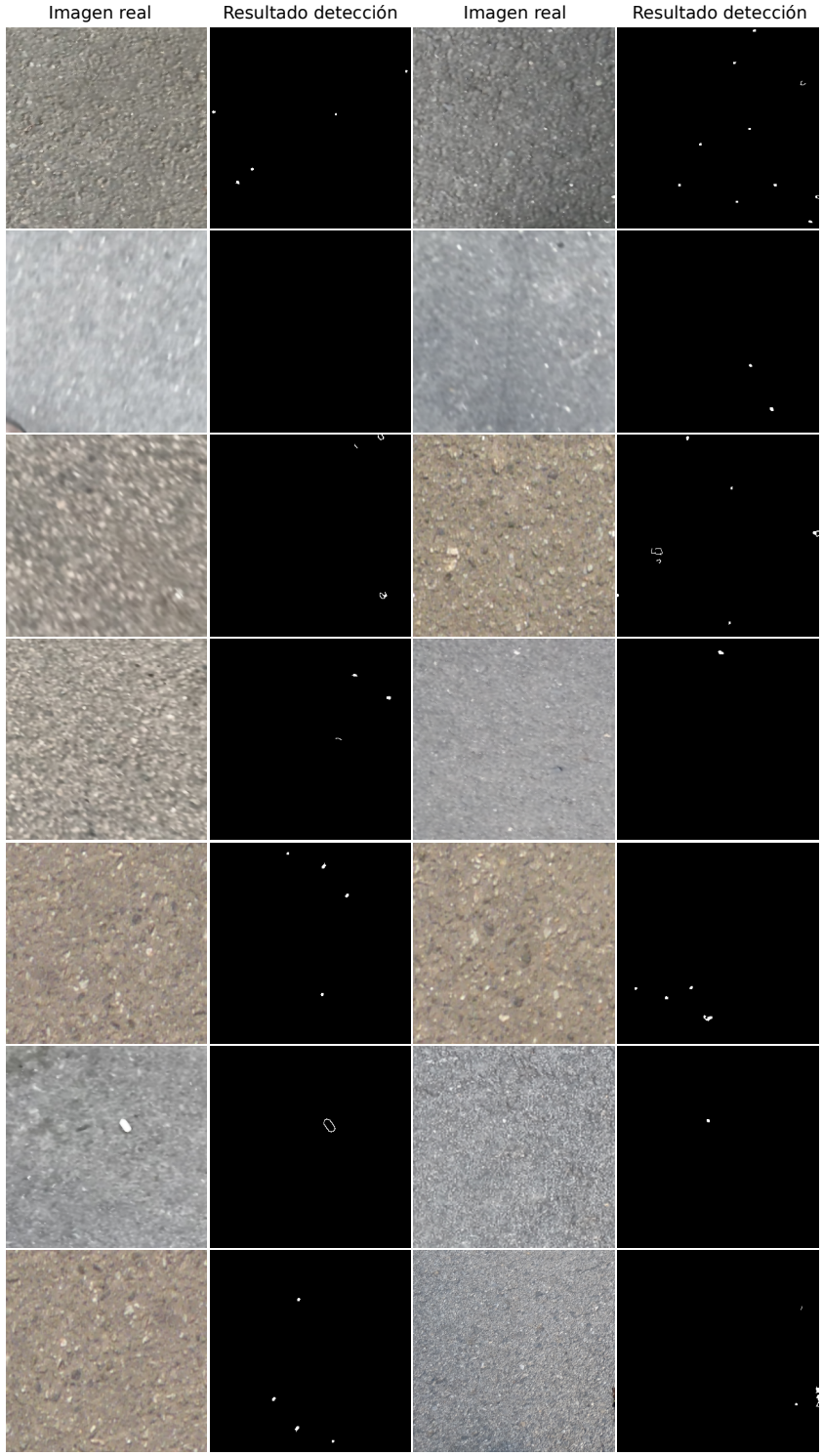


Figura 6.2
Resultados extracción de imagen binaria de varios pavimentos sin grietas.

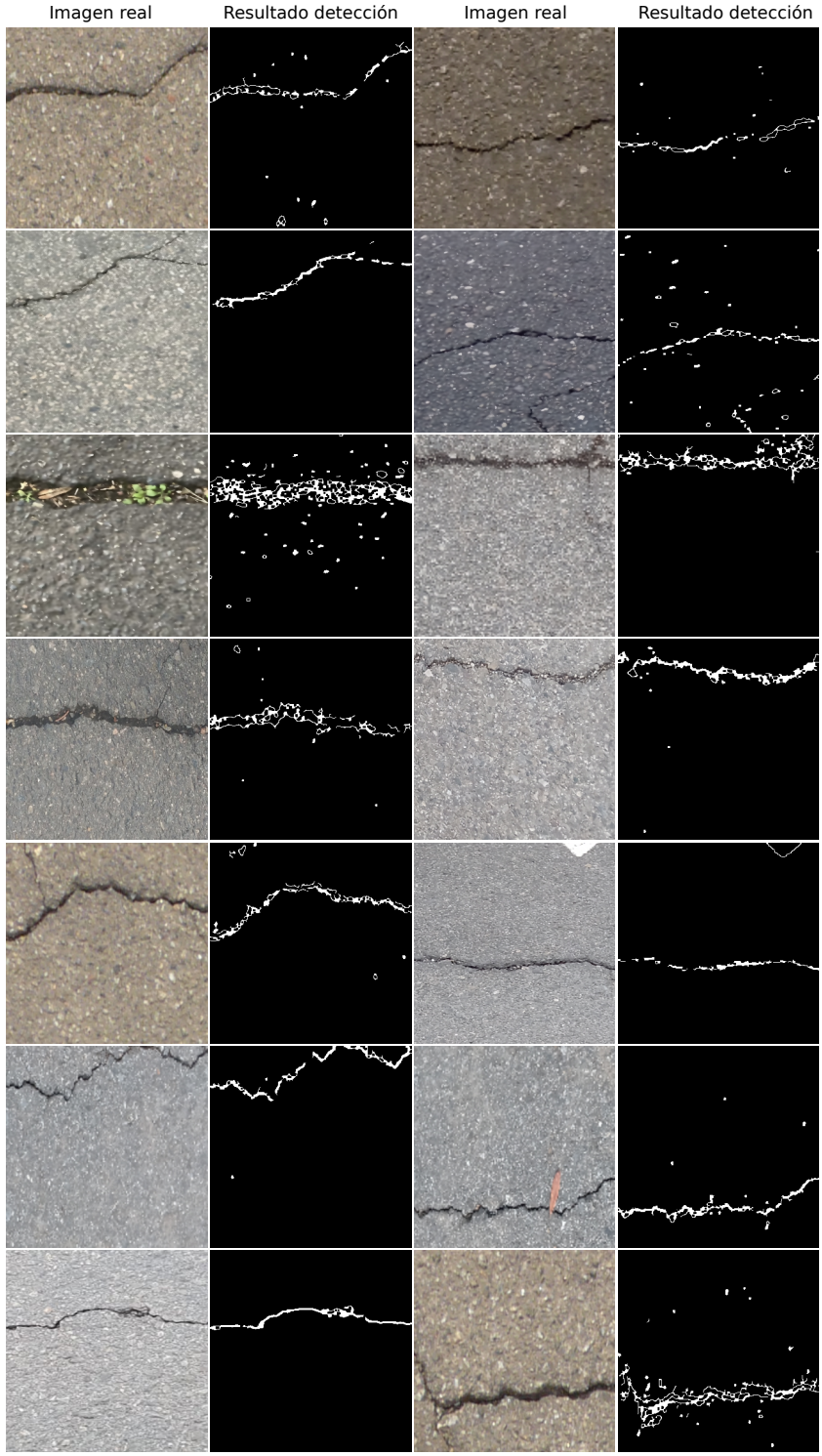


Figura 6.3
Resultados extracción de imagen binaria de varias grietas transversales.

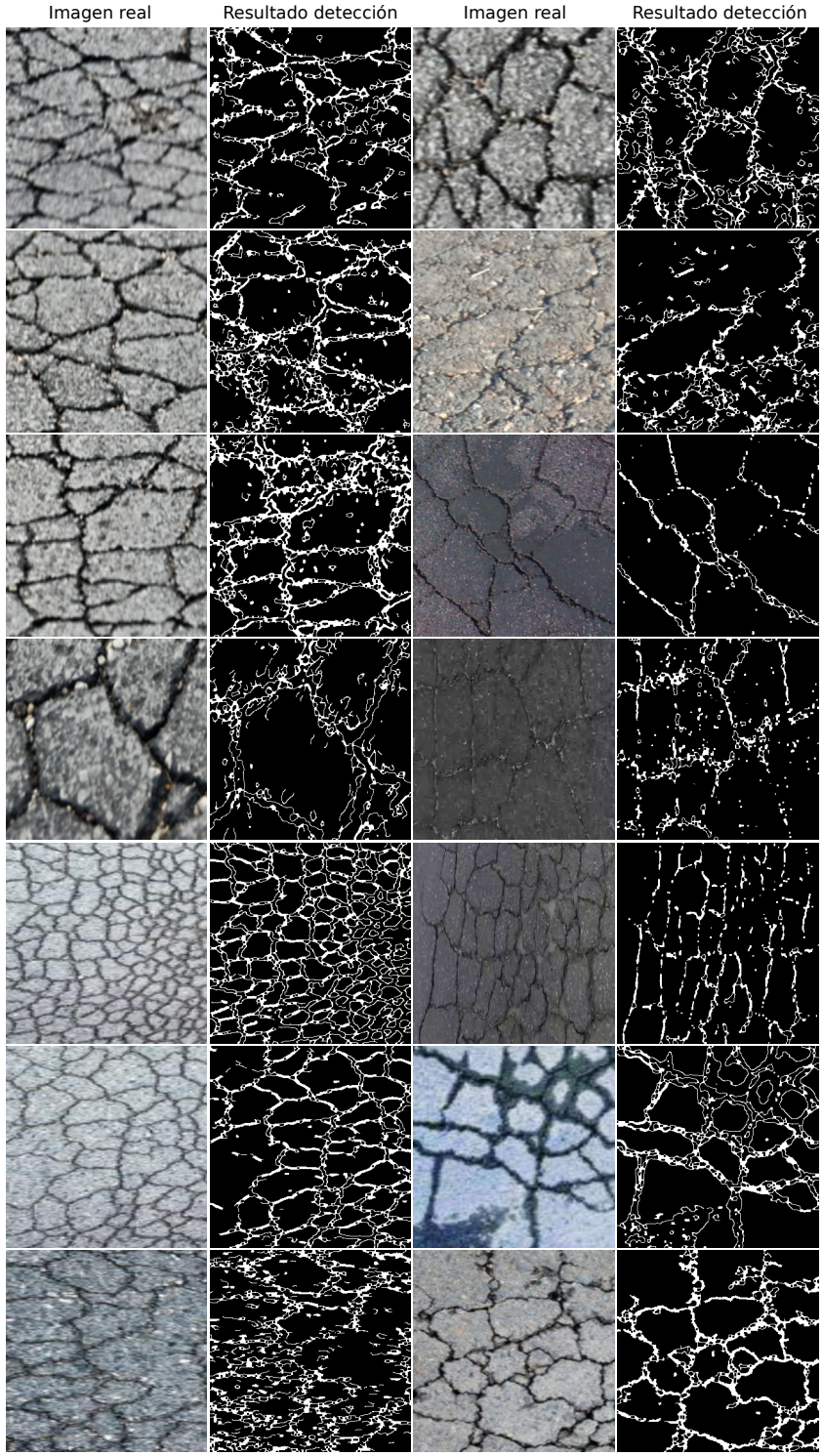


Figura 6.4
Resultados extracción de imagen binaria de varias grietas de tipo malla.

6.4 CLASIFICACIÓN DE CARACTERÍSTICAS

Una vez extraídas las integrales proyectivas horizontales y verticales (*La optimización de las proyecciones se ha dejado fuera de esta sección y se observará su efecto en la Sección 6.5*) se va a analizar el rendimiento de los algoritmos explicados en la Sección 3.3. Nótese que tanto los algoritmos de redes neuronales (NN) y máquinas de vectores soporte (SVM) han quedado fuera de esta sección y se verán en la Sección 6.6 debido a que son métodos empleados por otros autores, y se realizarán las comparaciones en dicha sección.

Al igual que ocurría en la sección anterior con los algoritmos de procesamiento de imágenes, en los algoritmo de aprendizaje automático supervisado (*véase la Sección 3.3*) también hay que definir de antemano los parámetros necesarios por cada método, los cuales se detallan a continuación:

- *Árbol de decisión (C4.5)*: Se selecciona un número mínimo de dos patrones por cada nodo hoja, así como un factor de confianza igual al 25 % y una poda a posteriori para evitar el aumento de profundidad innecesario del árbol.
- *Árboles de modelos logísticos (LMT)*: Se ha seleccionado 15 como el número mínimo de patrones para realizar las regresiones en las particiones del árbol.
- *Poda incremental repetida para la reducción de errores (RIPPER)*: Se lleva a cabo la poda de reglas para evitar su crecimiento.
- *Método de los* K *vecinos más cercanos (KNN)*: Se establece el número mínimo de vecinos a un valor igual a 1.
- *Árboles de decisión primero el mejor (BFTree)*: Se emplea el índice de *Gini* para realizar las particiones y se emplea el mismo número de patrones en los nodos hojas que C4.5.

Además, para el diseño de los experimentos se ha seguido una metodología de validación cruzada con 10 particiones estratificadas de acuerdo a lo detallado en la Sección 3.6.3 y la Sección 3.6.5 respectivamente. Esta combinación, permite comprobar el efecto del entrenamiento de los modelos con diferentes patrones de las bases de datos de la Tabla 6.1.

Por otro lado, en lo referente a las métricas usadas para el análisis del rendimiento de los algoritmos se emplean los valores de *Precision* y *Recall* mostrados en la Ecuación 3.19 y la Ecuación 3.20 respectivamente (*nótese que se utilizan sus nombres en inglés para que no haya confusión con el nombre de la primera métrica y la explicada en la Sección 3.5.1*), tal y como se observa en la Tabla 6.3, la Tabla 6.4 y la Tabla 6.5 para cada una de las bases de datos detalladas en la Sección 6.2.

De los resultados obtenidos del cálculo de la *Precision* y el *Recall* para la base de datos A, se puede observar en la Tabla 6.3, que para las grietas

Tabla 6.3
Resultados para los diferentes clasificadores con la base de datos A.

Datos	Método	Grieta Longitudinal		Grieta Transversal	
		Precision	Recall	Precision	Recall
Malla (100), Longitudinal(200), Transversal (200), Sin grietas (100)	C4.5	95.1 %	94.5 %	96.6 %	96.5 %
	LMT	93.6 %	90 %	92.1 %	95 %
	RIPPER	96 %	94.5 %	95.1 %	96 %
	KNN	100 %	99.5 %	100 %	99.5 %
	Naïve Bayes	96.7 %	97.5 %	83.3 %	100 %
	BFTree	93.3 %	95.5 %	96.5 %	93 %
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	C4.5	93.8 %	92 %	92.7 %	94 %
	LMT	94.9 %	97 %	99 %	96 %
	RIPPER	91.8 %	91 %	92.3 %	92 %
	KNN	100 %	94 %	92.9 %	100 %
	Naïve Bayes	81.1 %	100 %	89.3 %	35 %
	BFTree	97.1 %	91 %	91.9 %	99 %

de tipo longitudinal los clasificadores más sencillos como *KNN* y *Naïve Bayes* obtienen unos resultados bastante elevados. Esto se mantiene en el caso de *KNN* para el resto de clases, pero en el caso de *Naïve Bayes* dichos valores no se mantienen en ninguna de las dos métricas y en las imágenes en las que no existen ningún defecto el *Recall* que representa la exhaustividad no supera el 35 %. El resto de algoritmos mantienen unos valores siempre por encima del 91 %, mostrando en general los métodos basados en árboles de clasificación unos resultados ligeramente superiores al algoritmo basado en inducción de reglas *RIPPER*.

Para el caso de la base de datos B, como se muestra en la Tabla 6.4, se aprecia un comportamiento similar en el caso del algoritmo *Naïve Bayes*, aunque con estos datos el *Recall* en las grietas de tipo longitudinal es donde se encuentra su valor más bajo. Además, de forma generalizada se puede observar que los nuevos datos incorporados representan grietas más complejas que se traduce en una bajada en los porcentajes de la *Precision* y el *Recall* para todos los algoritmos quedándose por debajo del 90 % en las grietas de tipo longitudinal, de tipo malla y en patrones sin defectos. Este comportamiento bien puede deberse a la confusión de los modelos generados al etiquetar estos tipos concretos de grietas y se observa que este efecto perjudica al algoritmo *KNN* donde los valores para las grietas longitudinales y de tipo malla son menores que los de sus competidores basados en árboles de decisión e inducción de reglas.

Tabla 6.4
Resultados para los diferentes clasificadores con la base de datos B.

Datos	Método	Grieta Longitudinal		Grieta Transversal	
		Precision	Recall	Precision	Recall
Malla (380), Longitudinal(590), Transversal (558), Sin grietas (928)	C4.5	84.2 %	82.7 %	97.4 %	98.5 %
	LMT	83.6 %	85.2 %	97.4 %	97.5 %
	RIPPER	85.6 %	82.4 %	97.3 %	98.4 %
	KNN	82.3 %	85.1 %	91.9 %	97.2 %
	Naïve Bayes	75.9 %	47 %	87.1 %	95.9 %
	BFTree	84.5 %	81.2 %	96.3 %	98 %
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	C4.5	86.7 %	86.8 %	89.8 %	88.5 %
	LMT	89.1 %	91 %	91.1 %	86.7 %
	RIPPER	80.7 %	89.5 %	89.4 %	89.1 %
	KNN	97.3 %	81.8 %	93.9 %	91.2 %
	Naïve Bayes	77.7 %	82.6 %	62.5 %	74.3 %
	BFTree	88.9 %	87.6 %	88.2 %	89.4 %

Por último, los resultados de la Tabla 6.5 para la base de datos C, muestran como con la presencia de más datos los algoritmos basados en árboles de clasificación como *C4.5* y *BFTree* superan en términos de *Precision* y *Recall* a *KNN* en las grietas de tipo longitudinal y transversal respectivamente. Además, se demuestra que al igual que sucedía con las bases de datos anteriores, el método de *Naïve Bayes* no se comporta de la misma forma que los demás, obteniendo en la mayoría de los casos valores por debajo del 80 % para las dos métricas analizadas.

Una vez analizado el rendimiento de los diferentes algoritmos de clasificación, tal y como se propuso en la Sección 5.4 la idea no es emplear dichos modelos de forma individual, sino escoger los tres mejores y usarlos como los componentes de un ensemble de modelos regido por el voto de la mayoría. Para ello, en la Tabla 6.6 se expresan de forma compactada los valores de *Precision* y *Recall* para todas las clases en las diferentes bases de datos, utilizando la Ecuación 3.25 y la métrica del *F-Score* (*véase la Sección 3.5.5*). Además, se han marcado con colores el primer modelo que tiene el valor de *F-Score* más alto (*color rojo*), el segundo que tiene la métrica más alta (*color naranja*), y el tercero (*color marrón*). Hay que destacar que en esta tabla aunque el mejor modelo sería *KNN* para la base de datos A y C, este se ha descartado para su uso en el ensemble de modelos. Esto se debe a que *KNN* no genera un modelo a diferencia de los otros métodos analizados, sino que lleva a cabo la clasificación realizando comparaciones con todos los datos de entrenamiento, y a medida que estos crecen, la predicción de un nuevo dato tarda más tiempo (*o.2213 s*), siendo un comportamiento poco adecuado para trabajar en un sistema de recursos limitados, y en especial

Tabla 6.5
Resultados para los diferentes clasificadores con la base de datos C.

Datos	Método	Grieta Longitudinal		Grieta Transversal	
		Precision	Recall	Precision	Recall
Malla (3040), Longitudinal(4720), Transversal (4464), Sin grietas (7424)	C4.5	86.7 %	83.6 %	92.7 %	92.2 %
	LMT	84.5 %	81.7 %	91.7 %	89.8 %
	RIPPER	87.2 %	81.6 %	92.4 %	92.8 %
	KNN	86.4 %	80.3 %	90.8 %	96.8 %
	Naïve Bayes	70.5 %	45.4 %	86.1 %	95.4 %
	BFTree	87.4 %	80.9 %	93.4 %	97.8 %
		Grieta de tipo Malla		Pavimento sin grietas	
		Precision	Recall	Precision	Recall
	C4.5	93.2 %	92.3 %	93.8 %	96.6 %
	LMT	92.9 %	93 %	93.5 %	96.7 %
	RIPPER	92.2 %	92.3 %	93.3 %	96.8 %
	KNN	98.2 %	88.9 %	93.7 %	96.4 %
	Naïve Bayes	78.4 %	85.9 %	64.1 %	72.4 %
	BFTree	92.7 %	92.5 %	91.1 %	91.4 %

bajo las restricciones de tiempo real impuestas anteriormente.

Tras analizar la Tabla 6.6, se detecta que los modelos que siempre están dentro de los tres primeros puestos para las bases de datos analizadas, son el método *C4.5* y *BFTree*. Sin embargo, el tercer modelo recae entre el método *LMT* y *RIPPER*, motivo por el cual se deben de analizar dos ensembles diferentes:

- *Ensemble A*: Compuesto por los algoritmos *C4.5*, *BFTree* y *LMT*.
- *Ensemble B*: Compuesto por los algoritmos *C4.5*, *BFTRee* y *RIPPER*.

Tabla 6.6
Resultados de F-Score y tiempo para clasificar un nuevo dato.

Método	F-Score Base de datos A	F-Score Base de datos B	F-Score Base de datos C	Tiempo (s) Modelos Base de datos C
C4.5	94.2 %	89.2 %	91.3 %	0.00819
LMT	94.5 %	90.1 %	90.3 %	0.01334
RIPPER	93.3 %	89.7 %	91.1 %	$3.7532 \cdot 10^{-4}$
KNN	98.1 %	89.8 %	91.3 %	0.221362
Naïve Bayes	80.9 %	74.3 %	73.9 %	0.047550
BFTree	94.5 %	89.2 %	90.9 %	$2.38703 \cdot 10^{-4}$

Tal y como se puede observar en la Tabla 6.7 en la que se han marcado en rojo los valores máximos para cada ensemble, se puede concluir con certeza que el *Ensemble B* proporciona una mejor tolerancia a fallos

en la toma de decisión respecto al *Ensemble A*. Además, se demuestra de forma empírica que el uso de varios modelos heterogéneos dentro de un ensemble (*véase la Figura 5.9*) proporciona resultados superiores comparados a los obtenidos por los modelos individuales de la Tabla 6.3, Tabla 6.4 y la Tabla 6.5.

Tabla 6.7
Resultados de los diferentes ensembles.

Datos	**Método**	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (100) Longitudinal(200) Transversal (200) Sin grietas (100)	Ensemble A	97 %	94.5 %	97.1 %	**98 %**
	Ensemble B	**98.1 %**	**97 %**	**98 %**	**98 %**
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Ensemble A	93.9 %	**94 %**	**95.5 %**	97 %
	Ensemble B	**97.2 %**	**94 %**	94.6 %	**99 %**
Datos	**Método**	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (380) Longitudinal(590) Transversal (558) Sin grietas (928)	Ensemble A	85.9 %	**85.1 %**	**97.6 %**	98.4 %
	Ensemble B	**87 %**	**85.1 %**	97.5 %	**98.6 %**
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Ensemble A	**89.3 %**	**89.7 %**	**91.4 %**	89.6 %
	Ensemble B	89.2 %	**89.7 %**	91.2 %	**90.3 %**
Datos	**Método**	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (3040) Longitudinal(4720) Transversal (4464) Sin grietas (7424)	Ensemble A	89 %	85.6 %	94.5 %	**97.3 %**
	Ensemble B	**90.8 %**	**86.3 %**	**94.6 %**	97.2 %
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Ensemble A	95.2 %	**94.9 %**	94.1 %	93.3 %
	Ensemble B	**95.3 %**	**94.9 %**	**94.7 %**	**94.1 %**

6.5 CLASIFICACIÓN MEDIANTE CARACTERÍSTICAS OPTIMIZADAS

Determinados los modelos de clasificación que forman el ensemble, y observando las ventajas que este enfoque proporciona sobre los modelos individuales, es el momento de reducir la dimensionalidad de los atributos y dotar de interpretabilidad al modelo (*véase la Sección 5.3*). Dado que el ensemble está compuesto de dos algoritmos basados en árboles de clasificación, y reglas de decisión, es relativamente sencillo

que una persona que tenga el modelo completo y sea capaz de interpretarlo, pues inconscientemente solemos realizar reglas de asociación y esquemas de árboles de decisión desde la infancia. Sin embargo, dependiendo del número de atributos de los datos y de lo que estos representan, la tarea ya no es tan sencilla. Existen por tanto dos desventajas con el ensemble propuesto, y es que no permite realizar una predicción con imágenes de un tamaño diferente, y aunque las integrales proyectivas reducen el tamaño de la imagen, no proporcionan atributos fácilmente interpretables si el número de filas y columnas crece.

Debido a esto, se proponen tres enfoques para reducir los datos tal y como se detalla a continuación:

- *Ensemble PCA*: Ensemble empleando el conjunto de características reducida que proporciona el análisis de componentes principales manteniendo un 95 % de variabilidad de los datos. El número de atributos resultante para este porcentaje de variabilidad en las diferentes bases de datos es el siguiente:
  - *Base de datos A*: 7 atributos + 1 atributo de clase.
  - *Base de datos B*: 49 atributos + 1 atributo de clase.
  - *Base de datos C*: 46 atributos + 1 atributo de clase.
- *Ensemble PCA (4)*: Ensemble empleando el análisis de componentes principales seleccionando únicamente las cuatro primeras componentes principales.
- *Ensemble IPMDM*: Ensemble empleando la reducción de datos propuesta en la Sección 5.3.

La Tabla 6.8 muestra el resultado de los anteriores enfoques en el que se ha marcado en rojo el valor máximo de los valores de *Precision* y *Recall* para cada clase. Como se puede observar generalmente el ensemble junto con la reducción de datos propuesta en la Sección 5.3 se comporta casi como un clasificador perfecto sin fallos en la Base de datos A, llegando a serlo para la base de datos C. Y el uso del método de reducción de atributos basado en componentes principales es capaz de clasificar sin errores, un tipo en concreto de grietas como las transversales de la base de datos C, pero generalmente se queda como mínimo a una diferencia de un 1.5 %. No obstante, es en la base de datos B donde se aprecian las diferencias más significativas entre los tres enfoques, llegando a haber una diferencia en las grietas de tipo longitudinales en cuanto a valores de *Precision* (*y de forma similar en cuanto al Recall*) de un 18.5 % y un 21.3 % respectivamente para el análisis de componentes principales al 95 % de confianza y con cuatro componentes principales frente a la reducción propuesta en esta Tesis Doctoral.

Por tanto, la reducción propuesta es superior a los resultados que puede proporcionar el análisis de componentes principales, en sus dos versiones analizadas. Por otro lado, debido a que los atributos empleados en la reducción se pueden interpretar fácilmente por una persona, los modelos basados en reglas y en árboles, los cuales separan los

datos mediante los valores de los atributos, también adquieren esta características. Además, hay que destacar que al reducir las características a únicamente cuatro, el ensemble tarda la insignificante cifra de $1.8667 \cdot 10^{-4}$s en clasificar un nuevo dato, que unido al tiempo de procesamiento de las imágenes sigue proporcionando la misma tasa de imágenes por segundo (*aprox. 40 fps*).

Tabla 6.8
Resultados del ensemble con diferentes enfoques de reducción de datos.

Datos	Método	Grieta Longitudinal		Grieta Transversal	
		Precision	**Recall**	**Precision**	**Recall**
Malla (100) Longitudinal(200) Transversal (200) Sin grietas (100)	Ensemble PCA	95.8 %	96.5 %	96.5 %	96 %
	Ensemble PCA (4)	94.1 %	98 %	98.1 %	94 %
	Ensemble IPMDM	**99.5 %**	**100 %**	**100 %**	**99.5 %**
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Ensemble PCA	98.2 %	98 %	99.1 %	98 %
	Ensemble PCA (4)	98.2 %	99 %	**100 %**	98 %
	Ensemble IPMDM	**100 %**	**100 %**	**100 %**	**100 %**
Datos	**Método**	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (380) Longitudinal(590) Transversal (558) Sin grietas (928)	Ensemble PCA	78.4 %	79.5 %	96.5 %	95.1 %
	Ensemble PCA (4)	75.6 %	79.3 %	97.4 %	93.5 %
	Ensemble IPMDM	**96.9 %**	**94.2 %**	**99.8 %**	**99.8 %**
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Ensemble PCA	91.8 %	88.2 %	86.7 %	89 %
	Ensemble PCA (4)	90.6 %	87.9 %	86.7 %	89.4 %
	Ensemble IPMDM	**92.2 %**	**95.3 %**	**97.5 %**	**97.8 %**
Datos	**Método**	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (3040) Longitudinal(4720) Transversal (4464) Sin grietas (7424)	Ensemble PCA	97.9 %	98.4 %	**100 %**	**100 %**
	Ensemble PCA (4)	96.9 %	96.8 %	**100 %**	**100 %**
	Ensemble IPMDM	**100 %**	**100 %**	**100 %**	**100 %**
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Ensemble PCA	98.3 %	97.4 %	98.5 %	98.5 %
	Ensemble PCA (4)	98.2 %	98.3 %	97.2 %	97.2 %
	Ensemble IPMDM	**100 %**	**100 %**	**100 %**	**100 %**

6.6 COMPARACIÓN CON PROPUESTAS EXISTENTES

Una vez se ha seleccionado el mejor ensemble (*compuesto por C4.5, RIPPER y BFTree*) junto con la reducción de los atributos, queda comparar sus resultados con las propuestas existentes de otros autores de la literatura científica (*Véase el Capítulo 4*). Sin embargo, no todos los autores se centran en las mismas tareas ni trabajan sobre los mismos tipos de defectos, y por ello, se ha decidido llevar a cabo la comparativa con las tres siguientes propuestas de sus respectivos autores:

- *Cubero-Fernandez et al.* [42]: Estos autores se centran en la clasificación del mismo tipo de grietas así como proponer una clasificación multi-modelo.
- *Hoang y Nguyen* [75]: En el caso de estos autores la clasificación se realiza únicamente con un único modelo, y los tipos de defectos analizados son los mismos que persigue la presente Tesis Doctoral.
- *Li et al.* [101]: En este caso, los autores emplean una metodología mediante dos modelos en dos etapas o pasos para clasificar los defectos.

Los resultados con los trabajos de *Cubero-Fernandez et al.* y *Hoang y Nguyen* se pueden encontrar en la Tabla 6.9 para las tres bases de datos de la Tabla 6.1. Se puede observar que para la base de datos A, las diferencias con el ensemble en términos de *Precision* y *Recall* para las grietas longitudinales, transversales y en la clasificación de pavimentos sin defectos, son por lo general mayores al 20 % respecto a la propuesta de *Cubero-Fernandez et al.*, y que los modelos de estos autores únicamente se aproximan en términos de *Precision* en las grietas de tipo malla pero no en valores de *Recall*. En cuanto al trabajo de *Hoang y Nguyen*, se observa que esta diferencia se disminuye, pero que siguen por encima del 14 % para el caso de la detección de los patrones más comunes, la superficie sin defectos.

En el caso de la base de datos B, en la que las clases están desbalanceadas, se observa que el ensemble de modelos obtiene el máximo de los valores en todos los tipos de grietas en las dos métricas analizadas. La única excepción, se observa en la clase de los pavimentos sin defectos, donde en valores de *Recall*, el ensemble se encuentra un 0.4 % por debajo de la propuesta de *Cubero-Fernandez et al.*, pero que en términos de *Precision* para el mismo tipo de imágenes el ensemble obtiene un 16.03 % más. En lo referente a la propuesta de *Hoang y Nguyen* se observa que aunque se comporta en general un poco mejor que la de *Cubero-Fernandez et al.*, sigue sin poder competir en resultados con el ensemble.

En cuanto a los resultados de la base de datos C, tal y como se observa, el ensemble proporciona una salida en la predicción sin errores y que comparado con las propuestas de *Cubero-Fernandez et al.* y *Hoang y Nguyen*, la mejora en algunos tipos de grietas como las longitudinales se encuentra en torno al 15 % en términos de *Precision* y al 20 % en los

valores de *Recall*, y en el resto de grietas no son capaces de superar los obtenidos por el ensemble propuesto.

Tabla 6.9
Resultados de la comparación con otros autores para las tres bases de datos.

Datos	Método	Grieta Longitudinal		Grieta Transversal	
		Precision	Recall	Precision	Recall
Malla (100), Longitudinal(200), Transversal (200), Sin grietas (100)	*Cubero-Fernandez et al.* [42]	67 %	81.7 %	72 %	81 %
	Hoang y Nguyen [75]	97.5 %	96 %	97 %	96.5 %
	Ensemble	**99.5 %**	**100 %**	**100 %**	**99.5 %**
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Cubero-Fernandez et al. [42]	97 %	78.8 %	85 %	80.1 %
	Hoang y Nguyen [75]	91.6 %	98 %	88.7 %	86 %
	Ensemble	**100 %**	**100 %**	**100 %**	**100 %**
Datos	**Método**	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (380), Longitudinal(590), Transversal (558), Sin grietas (928)	*Cubero-Fernandez et al.* [42]	80.89 %	57.43 %	91.77 %	68.10 %
	Hoang y Nguyen [75]	76.3 %	78.6 %	89.7 %	88.9 %
	Ensemble	**96.9 %**	**94.2 %**	**99.8 %**	**99.8 %**
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Cubero-Fernandez et al. [42]	61.28 %	80.11 %	81.47 %	**98.2 %**
	Hoang y Nguyen [75]	90 %	87.4 %	93.7 %	93.5 %
	Ensemble	**92.2 %**	**95.3 %**	**97.5 %**	97.8 %
Datos	**Método**	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (3040), Longitudinal(4720), Transversal (4464), Sin grietas (7424)	*Cubero-Fernandez et al.* [42]	84.8 %	70.6 %	97.1 %	85.5 %
	Hoang y Nguyen [75]	83 %	79.3 %	93 %	91.9 %
	Ensemble	**100 %**	**100 %**	**100 %**	**100 %**
		Grieta de tipo Malla		**Pavimento sin grietas**	
		Precision	**Recall**	**Precision**	**Recall**
	Cubero-Fernandez et al. [42]	72.4 %	84.3 %	88.1 %	97.7 %
	Hoang y Nguyen [75]	95.4 %	92.8 %	91.3 %	95.6 %
	Ensemble	**100 %**	**100 %**	**100 %**	**100 %**

Las comparaciones de los resultados con el trabajo de *Li et al.* [101], se muestran en la Tabla 6.10. En este caso, los autores no consideran las imágenes en las que no existen defectos, y por lo tanto se han eli-

minado dichos patrones de la base de datos para poder realizar la comparación. Además, al igual que hacen los autores, se realiza una clasificación en dos etapas, primero en grietas de tipo malla o lineal, y luego estas últimas en longitudinales y transversales. Debido a esto, ahora existen dos ensembles (*cada uno de ellos compuesto por los mismos algoritmos y la reducción de atributos seleccionados en las secciones anteriores*) para la realización de la clasificación en dos pasos. Con esta comparación, se pretende analizar la metodología presentada en esta Tesis Doctoral en un escenario de clasificación por partes, pero empleando la propuesta del ensemble.

Tabla 6.10
Resultados del ensemble considerando sólo tres tipos de clases para las tres bases de datos.

Datos	**Método Paso 1**	**Grieta de tipo Malla**		**Grieta de tipo Lineal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (100) Lineal(400) Transversal (200) Longitudinal (200)	*Li et al.* [101]	**100 %**	**100 %**	**100 %**	**100 %**
	Ensemble	**100 %**	**100 %**	**100 %**	**100 %**
	Método Paso 2	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
	Li et al. [101]	86.6 %	81 %	82.2 %	87.5 %
	Ensemble	**99 %**	**100 %**	**100 %**	**99 %**
Datos	**Método Paso 1**	**Grieta de tipo Malla**		**Grieta de tipo Lineal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (380) Lineal(1148) Transversal (558) Longitudinal (590)	*Li et al.* [101]	92 %	69.5 %	90.7 %	98 %
	Ensemble	**95 %**	**94.2 %**	**98.1 %**	**98.3 %**
	Método Paso 2	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
	Li et al. [101]	98 %	94.2 %	94.8 %	98 %
	Ensemble	**99 %**	**98.8 %**	**98.7 %**	**98.9 %**
Datos	**Método Paso 1**	**Grieta de tipo Malla**		**Grieta de tipo Lineal**	
		Precision	**Recall**	**Precision**	**Recall**
Malla (3040) Lineal(9184) Transversal (4464) Longitudinal (4720)	*Li et al.* [101]	91.1 %	96.4 %	90.6 %	97.7 %
	Ensemble	**100 %**	**100 %**	**100 %**	**100 %**
	Método Paso 2	**Grieta Longitudinal**		**Grieta Transversal**	
		Precision	**Recall**	**Precision**	**Recall**
	Li et al. [101]	97.8 %	94.9 %	94.8 %	97.7 %
	Ensemble	**100 %**	**100 %**	**100 %**	**100 %**

Tal y como aparece marcado en color rojo en Tabla 6.10, para la base de datos A, no existirían diferencias entre el ensemble propuesto y la metodología de *Li et al.* para la distinción entre grietas de tipo malla

y lineales, puesto que realizan una clasificación perfecta. Sin embargo, para el resto de clases y bases de datos, el ensemble siempre obtiene resultados más altos y en el caso de la base de datos C, se llega a la clasificación perfecta, cosa que no ocurre con la propuesta de *Li et al.*. Esto demuestra que aunque se lleve a cabo una clasificación en dos pasos con diferentes modelos individuales, la combinación de varios modelos en un ensemble es más tolerante a fallos de clasificación.

7 CONCLUSIONES Y TRABAJOS FUTUROS

La conclusión final es que sabemos muy poco, y no obstante es asombroso que sepamos tanto, y más sorprendente aún que tan poco conocimiento nos dé tanto poder.
Bertrand Russell [1]

ÍNDICE

1 Bertrand Arthur William Russell (1872 - 1970) de origen británico fue filósofo, matemático, lógico y escritor galardonado con el Premio Nobel de Literatura en 1950.

7.1 CONCLUSIONES

El principal objetivo que ha guiado la realización de la presente Tesis Doctoral ha sido el desarrollo de un *sistema automático para la detección y la clasificación de grietas en pavimentos*. Para tal fin se distinguieron una serie de objetivos concretos que guiaban las partes que necesitaba alcanzar el sistema. Por ello, el sistema propuesto se divide en dos partes de acuerdo a los objetivos marcados, la detección de grietas, y la clasificación de las grietas.

En cuanto a la detección de las grietas, se puede observar en la metodología expuesta y corroborar con los experimentos que el sistema es capaz de procesar las imágenes a una tasa aproximada de 40 imágenes por segundo con una resolución de 320x320. Esta característica lo hace susceptible de poder trabajar con cámaras a una tasa mayor de 25 fps, o bien emplear resoluciones mayores si fuera necesario. Además, tal y como se demuestra con el hardware utilizado para llevar a cabo los experimentos, no existen impedimentos en emplearse la metodología propuesta en sistemas heterogéneos de bajo coste y recursos computacionales limitados, tal y como perseguía uno de los objetivos de la presente Tesis Doctoral.

En lo referente a la segunda parte del sistema, la cual está centrada en la clasificación de las imágenes en grietas de tipo malla, grietas longitudinales y grietas transversales, así como clasificar cuando una imagen corresponde a un asfalto libre de grietas, se ha propuesto el uso de un ensemble de modelos. Dicho ensemble está regido por el voto de la mayoría y compuesto por el número mínimo de métodos de clasificación para poder realizar este esquema de votación. Por otro lado, se ha proporcionado una reducción de los atributos que alimentan estos modelos del ensemble, consiguiendo reducir el espacio de características de entrada a sólo cuatro, siendo éstas invariantes a la resolución de la imagen y la localidad del defecto en la imagen. Estas nuevas características "dada su naturaleza" son interpretables, a diferencia de otros enfoques basados únicamente en integrales proyectivas. Estas características permiten generar un ensemble basado en árboles de clasificación y reglas de inducción que se comporta casi como un clasificador sin errores para las tres bases de datos analizadas en este documento. De este modo, se ha demostrado que un ensemble de modelos junto con la reducción de datos propuesta, se comporta mucho mejor que los modelos individuales. Considerando los resultados de todas las bases de datos analizadas, la diferencia mínima y máximas son de un 2.4 % y 12.6 % para la métrica de *Precision*, y un 1 % y 16.4 % para la métrica de *Recall*. Además, este enfoque proporciona unos resultados más precisos que las propuestas de otros autores (*basadas en otros modelos de clasificación individuales o en varios pasos de clasificación*), existiendo diferencias de hasta un 20 % en términos de métricas mencionadas anteriormente.

Una de las ventajas principales que se pueden deducir del sistema propuesto es su versatilidad, debido a que al estar separado en dos partes,

es posible utilizar cada una de ellas de forma independiente en sistemas distribuidos o incluso sustituir completamente alguna de ellas para emplear otras metodologías como la extracción de características basada en aprendizaje profundo, aunque en ese caso habría que analizar su rendimiento en sistemas de baja capacidad de cómputo.

Además de todo lo anterior, para llevar a cabo los experimentos, se ha creado una base de datos, la cual ha sido etiquetada patrón a patrón manualmente y que se facilita a la comunidad científica para que puedan hacer un uso sin restricciones de ella, para entrenar otros modelos diferentes o bien compararse con la metodología propuesta en este documento.

Por todo ello, se considera que todos los objetivos sin excepción han sido cumplidos satisfactoriamente con resultados aún mejor de los esperados originalmente, y que sin duda el trabajo realizado tiene su aportación en la literatura científica y plantea nuevas vías que descubrir y pasos que mejorar como se detalla en la siguiente sección.

7.2 TRABAJOS FUTUROS

Dada las características del sistema que se ha propuesto así como sus resultados, se pueden plantear diferentes mejoras así como nuevos campos de aplicación que permitan aprovechar su versatilidad.

Una de las primeras mejoras que se puede aplicar es la parte de extracción de características. Tal y como se observó en la Sección 6.3 el cambio del lugar del procesamiento de CPU a GPU mejoró drásticamente la tasa de imágenes por segundos a la que se puede trabajar, y es posible emplear el mismo enfoque con el resto de métodos siguiendo esta filosofía para analizar si es posible obtener una velocidad mayor de procesamiento. Además en esta parte del sistema, en concreto, en la obtención de la imagen binaria se detectaban pequeños artefactos, los cuales no tienen un gran impacto en la clasificación, pero que podrían eliminarse mediante una paso similar a la propuesta de *Amhaz et al.* [9], o bien mediante algún algoritmo de agrupamiento basado en densidades como los que se introdujeron en la Sección 3.2.1. No obstante, habría que analizar el impacto que tendría en el rendimiento de la clasificación y el tiempo empleado para eliminarlos.

En lo referente a la clasificación de defectos, sería interesante realizar un estudio para determinar el grado de deterioro de una grieta, para decidir en qué casos es necesario realizar un tratamiento preventivo o una reparación completa, basándose por ejemplo en el ancho de las grietas en la imagen o detectando su profundidad (*mediante visión estereoscópica o sensores de profundidad*). Por otro lado, sería interesante ampliar en investigaciones futuras el número de defectos que puedan aparecer en la capa del asfalto más allá de las grietas como baches, degradación de la granulación del asfalto o adición de polución a la superficie, entre otros [147]. Además, aprovechando la modularidad que presenta el uso de los ensembles de modelos, podría ser interesan-

te realizar un análisis de la incorporación de algoritmos basados en lógica difusa [164, 165] que permitan agregar más información de los expertos, cuando esta sea imprecisa y basada en intuiciones.

Por último, dada la versatilidad del sistema al poder incorporarse en diferentes sistemas de computo de recursos limitados similares al utilizado para llevar a cabo los experimentos (*NVIDIA Jetson Nano* [130]), se podría emplear para analizar grietas en superficies cementadas como las calles, o en lugares críticos como túneles o presas, en las que sea necesario el uso de un sistema aéreo no tripulado de unas dimensiones reducidas.

8

CONCLUSIONS AND FUTURE WORKS

The final conclusion is that we know very little, and yet it is astonishing that we know so much, and still more astonishing that so little knowledge can give us so much power.
Bertrand Russell [1]

ÍNDICE

1 Bertrand Arthur William Russell (1872 - 1970) Albert of British origin was a philosopher, mathematician, logician and writer who was awarded with the Nobel Prize for Literature in 1950.

8.1 CONCLUSIONS

This Doctoral Thesis has been guided by the main objective to develop an *Automatic system for pavement crack detection and classification*. For this reason, different specific objectives were recognized to identify the parts that the system needed to achieve. Therefore, the proposed system is divided into two parts according to the objectives, crack detection, and crack classification.

Regarding crack detection stage, it can be observed in the methodology and the experiments exposed, that the system can process images at approximately 40 fps with a resolution of 320x320. The proposed system is able to work with cameras at a frame rate greater than 25 fps, or use higher resolutions, if necessary. Also, as it is shown by the hardware used to carry out the experiments, that there are no impediments to apply the proposed methodology in low-cost heterogeneous systems with constrained computational resources, as one of the objectives sought in this Doctoral Thesis.

The second task of the system is focused on classifying images into alligator cracks, longitudinal cracks, and transverse cracks. Besides, it must be able to determine when an image corresponds to an asphalt free of cracks. For this task, the use of an ensemble model has been proposed. This ensemble is guided by majority voting and it is composed of the minimum number of classification methods to carry out this voting scheme. Also, a reduction of the attributes that feed the ensemble model has been provided, mapping the space of input features to only four. These new features are invariant to the resolution of the image and the location of the crack in the image. Besides, these attributes are interpretable unlike other approaches based solely on projective integrals or principal component analysis. This allows the generation of an ensemble based on classification trees and induction rules that behaves almost like an error-free classifier for the three databases analyzed. Thus, it has been demonstrated that the proposed ensemble model and data reduction obtain more accurate results than any of the individual models. Analyzing the results of all the databases, the minimum and maximum differences are 2.4 % and 12.6 % for the *Precision metric*, and 1 % and 16.4 % for the *Recall* metric. Also, this approach provides better results than the proposals of other authors (*based on single or multi-step classification models*), with differences of up to 20 % in terms of the metrics above-mentioned.

One of the main advantages that can be deduced from the proposed system is its versatility because it can be split into two parts. In consequence, it is possible to use each part independently in distributed systems or even to replace completely some of them to use other methodologies such as feature extraction based on deep learning. However, in this last case, we would have to analyze its performance in low-resource systems.

In addition, to carry out the experiments, a database has been created, which has been labeled manually pattern by pattern, and which is provided to the scientific community with unrestricted use. It allows other experts to use it, to train other different models, or even to compare themselves with the methodology proposed in this document.

Hence, it is considered that all the objectives without exception have been satisfactorily completed with even better results than initially expected. Also, the work carried out has its contribution to the scientific literature and raises new avenues to be discovered and steps to be improved as detailed in the following section.

8.2 FUTURE WORKS

The features of the system that has been proposed as well as its results could be improved, and considering its versatility could be applied to new fields of application.

The first improvement that can be applied is in the feature extraction part. As detailed in Section 6.3, the change in processing context from CPU to GPU improved the frame rate. Hence, the same change could be applied to other methods to analyze if it is possible to obtain a higher frame rate. Also, in this part of the system, small artifacts were detected in the binary image, which does not have a large impact on classification, but they could be removed in an additional step similar to the proposal of *Amhaz et al.* [9], or by some density-based clustering algorithm as introduced in Section 3.2.1. However, the impact on classification performance and the time taken to remove them should be analyzed.

Another point to analyze is in the classification step. It would be interesting to carry out a study to determine the degree of deterioration of a crack, to decide in which cases it is necessary a preventive treatment or a complete reparation. This could be based for example on the width of the cracks in the image or by detecting their depth (*by means of stereoscopic vision or depth sensors*). On the other hand, it would be interesting to extend in future investigations the number of defects that may appear in the asphalt layer beyond the cracks, such as potholes, ravelling, or addition of pollution to the surface, among others [147]. Also, taking advantage of the modularity presented by the use of an ensemble model, it could be interesting to carry out an analysis of the incorporation of algorithms based on fuzzy logic [164, 165] allowing the addition of more information from experts, when this is imprecise and based on their intuition.

Finally, given the versatility of the system to be used in different computer systems with constrained resources, like the one used in the experiment section (*NVIDIA Jetson Nano* [130]), it would be interesting to use the system in other problems scenarios. For example, the system could be used to analyze the cracks in concrete structures such as

streets, or even in critical places such as tunnels or dams, where the use of small unmanned aerial systems are necessary.

BIBLIOGRAFÍA

Las personas no son recordadas por el número de veces que fracasan, sino por el número de veces que tuvieron éxito.
Thomas Alva Edison [1]

[1] Noor A. and Mokhtar M.: *Experiencing various color models on colored images*. International Journal of Computer Applications, 169(2):29–33, jul 2017. (Citado en la página 15.)

[2] Hervé Abdi and Lynne J. Williams: *Principal component analysis*. WIREs Comput. Stat., 2(4):433–459, July 2010, ISSN 1939-5108. (Citado en la página 53.)

[3] Behnaz Abdollahi, Naofumi Tomita, and Saeed Hassanpour: *Data augmentation in training deep learning models for medical image analysis*. In *Intelligent Systems Reference Library*, pages 167–180. Springer International Publishing, 2020. (Citado en la página 73.)

[4] Hamed Habibi Aghdam and Elnaz Jahani Heravi: *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*. Springer Publishing Company, Incorporated, 1st edition, 2017, ISBN 331957549X. (Citado en la página 73.)

[5] Abbas Ahmadi, Sadjad Khalesi, and MohammadReza Bagheri: *Automatic road crack detection and classification using image processing techniques, machine learning and integrated models in urban areas: A novel image binarization technique*. Journal of Industrial and Systems Engineering, 11(Special issue: 14th International Industrial Engineering Conference):85–97, 2018, ISSN 1735-8272. (Citado en la página 89.)

[6] Dihao Ai, Guiyuan Jiang, Lam Siew Kei, and Chengwu Li: *Automatic pixel-level pavement crack detection using information of multi-scale neighborhoods*. IEEE Access, 6:24452–24463, 2018. (Citado en la página 88.)

[7] Mostafa Al-Emran: *Hierarchical reinforcement learning - a survey*. International Journal of Computing and Digital Systems, 4(2):137–143, apr 2015. (Citado en la página 49.)

[8] Abbas Alharan, Radhwan Alsagheer, and Ali Al-Haboobi: *Popular decision tree algorithms of data mining techniques: A review*. International Journal of Computer Science and Mobile Computing, 6:133–142, June 2017. (Citado en la página 59.)

1 Thomas Alva Edison (1847 - 1931) ha pasado a la historia como uno de los inventores más prolíficos de los siglos XIX y XX con más de 1000 patentes registradas.

[9] Rabih Amhaz, Sylvie Chambon, Jerome Idier, and Vincent Baltazart: *Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection*. IEEE Transactions on Intelligent Transportation Systems, 17(10):2718–2729, oct 2016. (Citado en las páginas 87, 99, 131 y 135.)

[10] Mihael Ankerst, Markus Breunig, Hans Peter Kriegel, and Joerg Sander: *Optics: Ordering points to identify the clustering structure*. In *Proceedings ACM SIGMOD International Conference on Management of Data*, volume 28, pages 49–60, June 1999. (Citado en la página 53.)

[11] ARM Corporation: *Cortex-a57*, 2019. [En línea]. Disponible en:. `https://developer.arm.com/ip-products/processors/cortex-a/cortex-a57`, [Accedido el 10 de Junio de 2020]. (Citado en la página 110.)

[12] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath: *Deep reinforcement learning: A brief survey*. IEEE Signal Processing Magazine, 34(6):26–38, nov 2017. (Citado en la página 49.)

[13] Nii O. Attoh-Okine: *Analysis of learning rate and momentum term in backpropagation neural network algorithm trained to predict pavement performance*. Advances in Engineering Software, 30(4):291 – 302, 1999, ISSN 0965-9978. (Citado en la página 67.)

[14] Peyman Babashamsi, Nur Izzi Md Yusoff, Halil Ceylan, Nor Ghani Md Nor, and Hashem Salarzadeh Jenatabadi: *Evaluation of pavement life cycle cost analysis: Review and analysis*. International Journal of Pavement Research and Technology, 9(4):241–254, 2016. (Citado en la página 2.)

[15] Jamshid Bagherzadeh and Hasan Asil: *A review of various semi-supervised learning models with a deep learning and memory approach*. Iran Journal of Computer Science, 2(2):65–80, dec 2018. (Citado en la página 49.)

[16] Pierre Baldi: *Autoencoders, unsupervised learning and deep architectures*. In *Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop - Volume 27*, UTLW'11, page 37–50. JMLR.org, 2011. (Citado en la página 74.)

[17] Seongdeok Bang, Somin Park, Hongjo Kim, Yeo san Yoon, and Hyoungkwan Kim: *A deep residual network with transfer learning for pixel-level road crack detection*. In *Proceedings of the 35th International Symposium on Automation and Robotics in Construction (ISARC)*. International Association for Automation and Robotics in Construction (IAARC), 2018. (Citado en la página 88.)

[18] David Barber: *Nearest neighbour classification*. In *Bayesian Reasoning and Machine Learning*, pages 322–328. Cambridge University Press, 2012. (Citado en la página 56.)

[19] Rahim Barzegar, Masoud Sattarpour, Ravinesh Deo, Elham Fijani, and Jan Adamowski: *An ensemble tree-based machine learning model for predicting the uniaxial compressive strength of travertine rocks*. Neural Computing and Applications, aug 2019. (Citado en la página 104.)

[20] Francesco Bella, Alessandro Calvi, and Fabrizio D'Amico: *Impact of pavement defects on motorcycles' road safety*. Procedia - Social and Behavioral Sciences, 53:942–951, 2012. (Citado en la página 2.)

[21] Faycal Bensaali and Abbes Amira: *Design and efficient fpga implementation of an rgb to ycrcb color space converter using distributed arithmetic*. In *Lecture Notes in Computer Science*, pages 991–995, August 2004. (Citado en la página 17.)

[22] Saket Bhardwaj and Ajay Mittal: *A survey on various edge detector techniques*. Procedia Technology, 4:220–226, 2012. (Citado en la página 32.)

[23] A. P. Bradley, R. P. W. Duin, P. Paclik, and T. C. W. Landgrebe: *Precision-recall operating characteristic (p-roc) curves in imprecise environments*. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 4, pages 123–127, 2006. (Citado en la página 78.)

[24] Leo Breiman: *Random forests*. Mach. Learn., 45(1):5–32, October 2001, ISSN 0885-6125. (Citado en la página 89.)

[25] Leo Breiman, Jerome Friedman, Charles J. Stone, and R.A. Olshen: *Classification and Regression Trees*. Chapman & Hall, New York, 1984, ISBN 0-412-04841-8. (Citado en las páginas 58 y 63.)

[26] A Broggi and E.D Dickmanns: *Applications of computer vision to intelligent vehicles*. Image and Vision Computing, 18(5):365–366, apr 2000. (Citado en la página 10.)

[27] Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao: *Diversity creation methods: a survey and categorisation*. Information Fusion, 6(1):5 – 20, 2005, ISSN 1566-2535. Diversity in Multiple Classifier Systems. (Citado en la página 71.)

[28] M. Budagavi: *Video compression using blur compensation*. In *IEEE International Conference on Image Processing 2005*, volume 2, pages II–882, Sep. 2005. (Citado en la página 27.)

[29] Jie Cai, Jiawei Luo, Shulin Wang, and Sheng Yang: *Feature selection in machine learning: A new perspective*. Neurocomputing, 300:70 – 79, 2018, ISSN 0925-2312. (Citado en la página 50.)

[30] John Canny: *A computational approach to edge detection*. Readings in Computer Vision, pages 184–203, 1987. (Citado en la página 38.)

[31] Ben Carterette: *Precision and recall*. In *Encyclopedia of Database Systems*, pages 2779–2779, New York, NY, 2018. Springer New York, ISBN 978-1-4614-8265-9. (Citado en la página 77.)

[32] Young Jin Cha, Wooram Choi, and Oral Büyüköztürk: *Deep learning-based crack damage detection using convolutional neural networks*. Computer-Aided Civil and Infrastructure Engineering, 32(5):361–378, 2017. (Citado en la página 88.)

[33] Gary Chai, Rudi van Staden, Hong Guan, Greg Kelly, and Sanaul Chowdhury: *The impacts of climate change on pavement maintenance in queensland, australia*. In *Materials and Infrastructures 2*, pages 207–221. John Wiley & Sons, Inc., 2016. (Citado en la página 4.)

[34] Chih Chung Chang and Chih Jen Lin: *LIBSVM: A library for support vector machines*. ACM Transactions on Intelligent Systems and Technology, 2(3):1–27, apr 2011. (Citado en la página 68.)

[35] Qiuxiang Chang and Zhenkai Xiong: *Vision-aware target recognition toward autonomous robot by kinect sensors*. Signal Processing: Image Communication, 84:115810, may 2020. (Citado en la página 10.)

[36] Chen Chen, Ray Surette, and Mubarak Shah: *Automated monitoring for security camera networks: promise from computer vision labs*. Security Journal, feb 2020. (Citado en la página 10.)

[37] Chunyu Chen and Keyu Xie: *Face recognition based on two-dimensional principal component analysis and kernel principal component analysis*. Information Technology Journal, 11(12):1781–1785, dec 2012. (Citado en la página 53.)

[38] H.D. Cheng, X.H. Jiang, and Y. Sun: *A survey on color image segmentation*. Proceedings of the Joint Conference on Information Sciences, 4:346–349, January 1998. (Citado en la página 19.)

[39] Jierong Cheng, Wei Xiong, Wenyu Chen, Ying Gu, and Yusha Li: *Pixel-level crack detection using u-net*. In *TENCON 2018 - 2018 IEEE Region 10 Conference*. IEEE, 2018. (Citado en la página 88.)

[40] Chien-Cheng Yu and Bin-Da Liu: *A backpropagation algorithm with adaptive learning rate and momentum coefficient*. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, volume 2, pages 1218–1223 vol.2, 2002. (Citado en la página 66.)

[41] William W. Cohen: *Fast effective rule induction*. In Armand Prieditis and Stuart Russell (editors): *Machine Learning Proceedings 1995*, pages 115 – 123. Morgan Kaufmann, San Francisco (CA), 1995, ISBN 978-1-55860-377-6. (Citado en las páginas 63 y 64.)

[42] A. Cubero-Fernandez, Fco. J. Rodriguez-Lozano, Rafael Villatoro, Joaquin Olivares, and Jose M. Palomares: *Efficient pavement crack detection and classification*. EURASIP Journal on Image and Video Processing, 2017(1):39, 2017, ISSN 1687-5281. (Citado en las páginas 91, 111, 125 y 126.)

[43] M. L. Dantas Dias and A. R. R. Neto: *Evolutionary support vector machines: A dual approach*. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 2185–2192, 2016. (Citado en las páginas 69 y 70.)

[44] Dale Steven Decker: *Best practices for crack treatments in asphalt pavements*. In *Proceedings of 6th Eurasphalt & Eurobitume Congress*. Czech Technical University in Prague, jun 2016. (Citado en la página 4.)

[45] G. Deng, L. W. Cahill, and G. R. Tobin: *The study of logarithmic image processing model and its application to image enhancement*. IEEE Transactions on Image Processing, 4(4):506–512, April 1995, ISSN 1941-0042. (Citado en la página 31.)

[46] Li Deng: *A tutorial survey of architectures, algorithms, and applications for deep learning*. APSIPA Transactions on Signal and Information Processing, 3, 2014. (Citado en la página 49.)

[47] Chunwang Dong, Gaozhen Liang, Bin Hu, Haibo Yuan, Yongwen Jiang, Hongkai Zhu, and Jiangtao Qi: *Prediction of congou black tea fermentation quality indices from color features using nonlinear regression methods*. Scientific Reports, 8(1), jul 2018. (Citado en las páginas 19 y 20.)

[48] Z. Dong, Z. Zhou, Z. Li, C. Liu, P. Huang, L. Liu, X. Liu, and J. Kang: *Convolutional neural networks based on rram devices for image recognition and online learning tasks*. IEEE Transactions on Electron Devices, 66(1):793–801, 2019. (Citado en la página 66.)

[49] ElizaYingzi Du, Robert Ives, Alan van Nevel, and Jin Hua She: *Advanced image processing for defense and security applications*. EURASIP Journal on Advances in Signal Processing, 2010(1), mar 2011. (Citado en la página 10.)

[50] Soumya Dutta and Bidyut B. Chaudhuri: *A color edge detection algorithm in RGB color space*. In *2009 International Conference on Advances in Recent Technologies in Communication and Computing*. IEEE, 2009. (Citado en la página 33.)

[51] Adel El-Shahat: *Introductory chapter: Artificial neural networks*. In *Advanced Applications for Artificial Neural Networks*. InTech, feb 2018. (Citado en la página 65.)

[52] Escuela Politécnica Superior de Córdoba: *Información Corporativa*, 2017. [En línea]. Disponible en:. `http://www.uco.es/organiza/centros/eps/es/epsc/informacion-corporativa`, [Accedido el 5 de Abril de 2020]. (Citado en la página 39.)

[53] F. Esposito, D. Malerba, G. Semeraro, and J. Kay: *A comparative analysis of methods for pruning decision trees*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(5):476–493, may 1997. (Citado en la página 59.)

[54] Martin Ester, Hans Peter Kriegel, Jörg Sander, and Xiaowei Xu: *A density-based algorithm for discovering clusters in large spatial databases with noise*. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996. (Citado en la página 53.)

[55] Habib Fathi, Fei Dai, and Manolis Lourakis: *Automated as-built 3d reconstruction of civil infrastructure using computer vision: Achievements, opportunities, and challenges*. Advanced Engineering Informatics, 29(2):149–161, apr 2015. (Citado en la página 10.)

[56] Ike Fitriyaningsih and Yuniarta Basani: *Flood prediction with ensemble machine learning using BP-NN and SVM*. Jurnal Teknologi dan Sistem Komputer, 7(3):93–97, jul 2019. (Citado en la página 104.)

[57] Kyle Foss: *Welcome to your autonomous life: Self driving cars are a new reality*. Science Trends, 2017. (Citado en la página 2.)

[58] Jerome Friedman, Trevor Hastie, and Robert Tibshirani: *Additive logistic regression : A statistical view of boosting*. Annals of statistics, 28(2):337–407, 2000, ISSN 0090-5364. (Citado en la página 58.)

[59] Price DJ Galbraith RM: *Scottish road network climate change study*, 2005. (Citado en la página 4.)

[60] Xianwei Gao, Chun Shan, Changzhen Hu, Zequn Niu, and Zhen Liu: *An adaptive ensemble machine learning model for intrusion detection*. IEEE Access, 7:82512–82521, 2019. (Citado en la página 104.)

[61] N.J. Garber and L.A. Hoel: *Traffic & highway engineering*, 2008, ISBN 9781111800833. (Citado en la página 3.)

[62] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, Pablo Martinez-Gonzalez, and Jose Garcia-Rodriguez: *A survey on deep learning techniques for image and video semantic segmentation*. Applied Soft Computing, 70:41 – 65, 2018, ISSN 1568-4946. (Citado en la página 73.)

[63] Ginés García Mateos: *Procesamiento de caras humanas mediante integrales proyectivas*. Tesis de Doctorado, Universidad de Murcia, 2006. (Citado en la página 43.)

[64] Gobierno de España - Ministerio de Transportes, Movilidad y Agenda Urbana: *Catálogo y evolución de la red de carreteras*, 2018. [En línea]. Disponible en:. https://www.mitma.gob.es/carreteras/catalogo-y-evolucion-de-la-red-de-carreteras, [Accedido el 6 de Mayo de 2020]. (Citado en la página 2.)

[65] Gobierno de España - Ministerio de Transportes, Movilidad y Agenda Urbana: *Datos mensuales de tráfico*, 2018. [En línea]. Disponible en:. https://www.mitma.es/carreteras/trafico-velocidades-y-accidentes-mapa-estimacion-y-evolucion/datos-mensuales-de-trafico/datos-mensuales-de-trafico-en-la-rce, [Accedido el 6 de Mayo de 2020]. (Citado en la página 2.)

[66] Heitor Murilo Gomes, Jean Paul Barddal, Fabrício Enembreck, and Albert Bifet: *A survey on ensemble learning for data stream classification*. ACM Comput. Surv., 50(2):23:1–23:36, 2017, ISSN 0360-0300. (Citado en la página 71.)

[67] Rafael C. Gonzalez and Richard E. Woods: *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., USA, 4nd edition, 2017, ISBN 1292223049. (Citado en la página 11.)

[68] Asela Gunawardana and Guy Shani: *A survey of accuracy evaluation metrics of recommendation tasks*. J. Mach. Learn. Res., 10:2935–2962, December 2009, ISSN 1532-4435. (Citado en la página 76.)

[69] David J. Hand and Keming Yu: *Idiot's bayes–Not so stupid after all?* International Statistical Review, 69(3):385–398, 2001. (Citado en la página 57.)

[70] Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin: *The elements of statistical learning: Data mining, inference, and prediction*. Math. Intell., 27:83–85, November 2004. (Citado en la página 82.)

[71] G. D. Hastings and A. Rubin: *Colour spaces - a review of historic and modern colour models**. African Vision and Eye Health, 71(3):133–143, dec 2012. (Citado en la página 12.)

[72] Marti A. Hearst: *Support vector machines*. IEEE Intelligent Systems, 13(4):18–28, July 1998, ISSN 1541-1672. (Citado en la página 68.)

[73] Dr. A. Hema and R. Saravanakumar and: *A survey on feature extraction technique in image processing*. International Journal of Trend in Scientific Research and Development, Volume-2(Issue-4):448–451, jun 2018. (Citado en la página 10.)

[74] Anusara Hirunyawanakul, Kedkarn Chaiyakhan, Ratiporn Chanklan, Kittisak Kerdprasop, and Nittaya Kerdprasop: *A new efficient method to improve handwritten signature recognition*. In *The Proceedings of the 2nd International Conference on Industrial Application Engineering 2015*. The Institute of Industrial Applications Engineers, 2015. (Citado en la página 34.)

[75] Nhat Duc Hoang and Quoc Lam Nguyen: *A novel method for asphalt pavement crack classification based on image processing and machine learning*. Engineering with Computers, 2018. (Citado en las páginas 89, 125 y 126.)

[76] Karel Horak, Jan Klecka, Ondrej Bostik, and Daniel Davidek: *Classification of SURF image features by selected machine learning algorithms*. In *2017 40th International Conference on Telecommunications and Signal Processing (TSP)*. IEEE, jul 2017. (Citado en la página 90.)

[77] Khalid Hussain, Shanto Rahman, Md. Mostafijur Rahman, Shah Mostafa Khaled, M. Abdullah Al Wadud, Muhammad Asif Hossain Khan, and Mohammad Shoyaib: *A histogram specification technique for dark image enhancement using a local transformation method*. IPSJ Transactions on Computer Vision and Applications, 10(1), feb 2018. (Citado en la página 42.)

[78] Noor A. Ibraheem, Mokhtar M. Hasan, Rafiqul Z. Khan, and Pramod K. Mishra: *Understanding color models: A review*. ARPN Journal of Science and Technology, 2(3):265–275, Apr 2012. (Citado en las páginas 17 y 19.)

[79] A. Ibrahim, M.K. Osman, N.A.M. Yusof, K.A. Ahmad, N.H. Harun, and R.A.A. Raof: *Characterization of cracking in pavement distress using image processing techniques and k-nearest neighbour*. Indonesian Journal of Electrical Engineering and Computer Science, 14(2):810, 2019. (Citado en la página 89.)

[80] Keith Jack: *Video Demystified: a handbook for the digital engineer*. Newnes, Burlington, 5th edition, 2007, ISBN 978-0-7506-8395-1. (Citado en la página 17.)

[81] Xiaojuan Jiang, Yinghua Zhang, Wensheng Zhang, and Xian Xiao: *A novel sparse auto-encoder for deep unsupervised learning*. In *2013 Sixth International Conference on Advanced Computational Intelligence (ICACI)*. IEEE, oct 2013. (Citado en la página 75.)

[82] Rita Justo-Silva and Adelino Ferreira: *Pavement maintenance considering traffic accident costs*. International Journal of Pavement Research and Technology, 12(6):562–573, nov 2019. (Citado en la página 2.)

[83] Ammar Ismael Kadhim: *Survey on supervised machine learning techniques for automatic text classification*. Artificial Intelligence Review, 52(1):273–292, jan 2019. (Citado en la página 49.)

[84] Samruddhi Y. Kahu, Rajesh B. Raut, and Kishor M. Bhurchandi: *Review and evaluation of color spaces for image/video compression*. Color Research & Application, 44(1):8–33, nov 2018. (Citado en la página 12.)

[85] Gábor Kallós: *A generalization of pascal's triangle using powers of base numbers*. Annales mathématiques Blaise Pascal, 13(1):1–15, 2006. (Citado en la página 28.)

[86] Memoona Khanum, Tahira Mahboob, Warda Imtiaz, Humaraia Abdul Ghafoor, and Rabeea Sehar: *A survey on unsupervised machine learning algorithms for automation, classification and maintenance*. International Journal of Computer Applications, 119(13):34–39, jun 2015. (Citado en la página 48.)

[87] Rohit Khokher, Ram Chandra Singh, and Rahul Kumar: *Footprint recognition with principal component analysis and independent component analysis*. Macromolecular Symposia, 347(1):16–26, jan 2015. (Citado en la página 53.)

[88] Hyunjun Kim, Eunjong Ahn, Myoungsu Shin, and Sung Han Sim: *Crack and noncrack classification from concrete surface images using machine learning*. Structural Health Monitoring, 18(3):725–738, 2018. (Citado en las páginas 3 y 90.)

[89] David B. Kirk and Wen mei W. Hwu: *Programming Massively Parallel Processors, Third Edition: A Hands-on Approach*, chapter 2. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition, 2016, ISBN 0128119861, 9780128119860. (Citado en la página 14.)

[90] Frank Klawonn and Frank Rehm: *Clustering techniques for outlier detection*. In *Encyclopedia of Data Warehousing and Mining*, pages 180–183. IGI Global, 2005. (Citado en la página 50.)

[91] Ron Kohavi: *A study of cross-validation and bootstrap for accuracy estimation and model selection*. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, page 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc., ISBN 1558603638. (Citado en la página 82.)

[92] A. Koschan and M. Abidi: *Detection and classification of edges in color images*. IEEE Signal Processing Magazine, 22(1):64–73, jan 2005. (Citado en la página 33.)

[93] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton: *Imagenet classification with deep convolutional neural networks*. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (editors): *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. (Citado en la página 73.)

[94] Pavel Krömer, Jan Platoš, and Václav Snášel: *Fast dimension reduction based on nmf*. In Zhihua Cai, Chengyu Hu, Zhuo Kang, and Yong Liu (editors): *Advances in Computation and Intelligence*, pages 424–433, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg, ISBN 978-3-642-16493-4. (Citado en la página 53.)

[95] Suhaili Beeran Kutty, Shuria Saaidin, Puteri Nor Ashikin Megat Yunus, and Suhana Abu Hassan: *Evaluation of canny and sobel operator for logo edge detection*. In *2014 International Symposium on Technology Management and Emerging Technologies*. IEEE, may 2014. (Citado en la página 35.)

[96] R. Lancini and S. Tubaro: *Progressive image transmission based on image projections*. In *Proceedings of 1st International Conference on Image Processing*, volume 3, pages 861–865 vol.3, 1994. (Citado en la página 43.)

[97] Niels Landwehr, Mark Hall, and Eibe Frank: *Logistic model trees*. Machine Learning, 59(1-2):161–205, 2005. (Citado en la página 62.)

[98] Pat Langley and Herbert A. Simon: *Applications of machine learning and rule induction*. Commun. ACM, 38(11):54–64, 1995, ISSN 0001-0782. (Citado en la página 63.)

[99] Jaeyoung Lee, BooHyun Nam, and Mohamed Abdel-Aty: *Effects of pavement surface conditions on traffic crash severity*. Journal of Transportation Engineering, 141(10):04015020, 2015. (Citado en la página 2.)

[100] Baoxian Li, Kelvin C. P. Wang, Allen Zhang, Enhui Yang, and Guolong Wang: *Automatic classification of pavement crack using deep convolutional neural network*. International Journal of Pavement Engineering, pages 1–7, 2018. (Citado en la página 90.)

[101] Li Li, Lijun Sun, Guobao Ning, and Shengguang Tan: *Automatic pavement crack recognition based on bp neural network*. Promet Traffic&Transportation, 26(1):11–22, 2014. (Citado en las páginas 91, 99, 125, 126 y 127.)

[102] Zhiqiang Li, Chengqi Cheng, Mei Po Kwan, Xiaochong Tong, and Shaohong Tian: *Identifying asphalt pavement distress using UAV LiDAR point cloud data and random forest classification*. ISPRS International Journal of Geo-Information, 8(1):39, 2019. (Citado en la página 90.)

[103] Ziming Li and Maarten de Rijke: *The impact of linkage methods in hierarchical clustering for active learning to rank*. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '17*. ACM Press, 2017. (Citado en la página 51.)

[104] R. G. Lins and S. N. Givigi: *Automatic crack detection and measurement based on image analysis*. IEEE Transactions on Instrumentation and Measurement, 65(3):583–590, 2016. (Citado en la página 87.)

[105] Wenting Luo, Kelvin C. P. Wang, Lin Li, Qiang Joshua Li, and Mike Moravec: *Surface drainage evaluation for rigid pavements using an inertial measurement unit and 1-mm three-dimensional texture data*. Transportation Research Record: Journal of the Transportation Research Board, 2457(1):121–128, 2014. (Citado en las páginas 2, 4 y 90.)

[106] Scott M. Lynch: *Bayesian theory, history, applications, and contemporary directions*. In James D. Wright (editor): *International Encyclopedia of the Social & Behavioral Sciences (Second Edition)*, pages 378 – 382. Elsevier, Oxford, second edition edition, 2015, ISBN 978-0-08-097087-5. (Citado en la página 57.)

[107] Teresa López-Montero y Rodrigo Miró: *El envejecimiento en mezclas asfálticas.* Cuadernos tecnológicos de la PTC, 2015. (Citado en la página 3.)

[108] Hossin M and Sulaiman M.N: *A review on evaluation metrics for data classification evaluations*. International Journal of Data Mining & Knowledge Management Process, 5(2):01–11, mar 2015. (Citado en las páginas 77 y 79.)

[109] Alberto S. Aguado Mark S. Nixon: *Feature Extraction & Image Processing for Computer Vision*, chapter 13, pages 541–600. Elsevier, Oxford, UK, 3rd edition, 2012, ISBN 978-0-123-96549-3. (Citado en la página 14.)

[110] Stephen Marsland: *Unsupervised learning*. In *Machine Learning*, pages 281–304. Chapman and Hall/CRC, oct 2014. (Citado en la página 48.)

[111] Al Mehdi and M. Shahidur: *A comparative study between brightness preserving bi-histogram and tri-histogram equalization for image contrast enhancement*. International Journal of Computer Applications, 140(2):35–39, apr 2016. (Citado en la página 42.)

[112] Roland Memisevic: *Deep learning: Architectures, algorithms, applications*. In *2015 IEEE Hot Chips 27 Symposium (HCS)*. IEEE, aug 2015. (Citado en la página 49.)

[113] Lingheng Meng, Shifei Ding, Nan Zhang, and Jian Zhang: *Research of stacked denoising sparse autoencoder*. Neural Comput. Appl., 30(7):2083–2100, October 2018, ISSN 0941-0643. (Citado en la página 74.)

[114] Brian N. Mills, Susan L. Tighe, Jean Andrey, James T. Smith, and Ken Huen: *Climate change implications for flexible pavement design and performance in southern canada*. Journal of Transportation Engineering, 135(10):773–782, 2009. (Citado en la página 4.)

[115] Erxue Min, Xifeng Guo, Qiang Liu, Gen Zhang, Jianjing Cui, and Jun Long: *A survey of clustering with deep learning: From the perspective of network architecture*. IEEE Access, 6:39501–39514, 2018. (Citado en la página 49.)

[116] MIPI Alliance: *Mipi camera serial interface 2 (mipi csi-2)*, 2010. [En línea]. Disponible en:. https://www.mipi.org/specifications/csi-2, [Accedido el 24 de Mayo de 2020]. (Citado en la página 111.)

[117] Phillip A. Mlsna and Jeffrey J. Rodríguez: *Gradient and laplacian edge detection*. In *Handbook of Image and Video Processing*, pages 535–553. Elsevier, 2005. (Citado en la página 37.)

[118] Phillip A. Mlsna and Jeffrey J. Rodríguez: *The Essential Guide to Image Processing*, chapter 19, pages 495 – 524. Academic Press, Boston, 2009, ISBN 978-0-12-374457-9. (Citado en la página 32.)

[119] Arun Mohan and Sumathi Poobal: *Crack detection using image processing: A critical review and analysis*. Alexandria Engineering Journal, 57(2):787–798, jun 2018. (Citado en las páginas 3 y 87.)

[120] Iqbal Muhammad and Zhu Yan: *Supervised machine learning approaches: A survey*. ICTACT Journal on Soft Computing, 05(03):946–952, apr 2015. (Citado en la página 48.)

[121] M. N. Murty and Rashmi Raghava: *Kernel-based svm*. In *Support Vector Machines and Perceptrons: Learning, Optimization, Classification, and Application to Social Networks*, pages 57–67, Cham, 2016. Springer International Publishing, ISBN 978-3-319-41063-0. (Citado en la página 70.)

[122] Vinod Nair and Geoffrey E. Hinton: *Rectified linear units improve restricted boltzmann machines*. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA, 2010. Omnipress, ISBN 9781605589077. (Citado en la página 74.)

[123] U. Narayanan, A. Unnikrishnan, V. Paul, and S. Joseph: *A survey on various supervised classification algorithms*. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pages 2118–2124, 2017. (Citado en la página 48.)

[124] Remi Niel, Jasper Krebbers, Madalina M. Drugan, and Marco A. Wiering: *Hierarchical reinforcement learning for real-time strategy games*. In *Proceedings of the 10th International Conference on Agents and Artificial Intelligence*. SCITEPRESS - Science and Technology Publications, 2018. (Citado en la página 49.)

[125] Sushma NiketBorade and Ratnadeep R. Deshmukh: *Comparative study of principal component analysis and independent component analysis*. International Journal of Computer Applications, 92(15):45–49, apr 2014. (Citado en la página 53.)

[126] Nisha, Rajesh Mehra, and Lalita Sharma: *Comparative analysis of canny and prewitt edge detection techniques used in image processing*. International Journal of Engineering Trends and Technology, 28(1):48–53, oct 2015. (Citado en la página 36.)

[127] Northwest Pavement Management Systems Users Group and R. Keith Kay: *Pavement surface condition - rating manual*, Northwest Technologies Transfer Center Washington State, Department of Transportation Local Programs Division, 1992. (Citado en la página 4.)

[128] Nurhayati, I. Soekarno, I. K. Hadihardaja, and M. Cahyono: *A study of hold-out and k-fold cross validation for accuracy of groundwater modeling in tidal lowland reclamation using extreme learning machine*. In *2014 2nd International Conference on Technology, Informatics, Management, Engineering Environment*, pages 228–233, 2014. (Citado en la página 80.)

[129] NVIDIA Corporation: *Maxwell architecture*, 2014. [En línea]. Disponible en:. `https://developer.nvidia.com/maxwell-compute-architecture`, [Accedido el 10 de Junio de 2020]. (Citado en la página 110.)

[130] NVIDIA Corporation: *Jetson nano developer kit*, 2020. [En línea]. Disponible en:. `https://developer.nvidia.com/embedded/jetson-nano-developer-kit`, [Accedido el 24 de Mayo de 2020]. (Citado en las páginas 110, 132 y 135.)

[131] S.D Olabarriaga and A.W.M Smeulders: *Interaction in the segmentation of medical images: A survey*. Medical Image Analysis, 5(2):127–142, jun 2001. (Citado en la página 10.)

[132] J. Arturo Olvera-López, J. Ariel Carrasco-Ochoa, and J. Francisco Martínez-Trinidad: *A new fast prototype selection method based on clustering*. Pattern Analysis and Applications, 13(2):131–141, jan 2009. (Citado en la página 50.)

[133] Rafael Palomar, José M. Palomares, José M. Castillo, Joaquín Olivares, and Juan Gómez-Luna: *Parallelizing and optimizing LIP-canny using NVIDIA CUDA*. In *Trends in Applied Intelligent Systems*, pages 389–398. Springer Berlin Heidelberg, 2010. (Citado en la página 38.)

[134] J.M. Palomares, J. Gonzalez, E. Ros, and A. Prieto: *General logarithmic image processing convolution*. IEEE Transactions on Image Processing, 15(11):3602–3608, nov 2006. (Citado en la página 31.)

[135] Jung Me Park and Yi Lu Murphey: *Edge detection in grayscale, color, and range images*. In *Wiley Encyclopedia of Computer Science and Engineering*, pages 1–16. American Cancer Society, 2008, ISBN 9780470050118. (Citado en la página 32.)

[136] Dipti D. Patil, V.M. Wadhai, and J.A. Gokhale: *Evaluation of decision tree pruning algorithms for complexity and classification accuracy*. International Journal of Computer Applications, 11(2):23–30, dec 2010. (Citado en la página 59.)

[137] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei Ling Shyu, Shu Ching Chen, and S. S. Iyengar: *A survey on deep learning: Algorithms, techniques, and applications*. ACM Comput. Surv., 51(5), September 2018, ISSN 0360-0300. (Citado en la página 73.)

[138] Kevin L. Priddy and Paul E. Keller: *Artificial Neural Networks*. SPIE, 2005, ISBN 9-78081945-987-9. (Citado en la página 65.)

[139] Swarnalatha Purushotham and B. K. Tripathy: *Evaluation of classifier models using stratified tenfold cross validation techniques*. In *Communications in Computer and Information Science*, pages 680–690. Springer Berlin Heidelberg, 2012. (Citado en la página 83.)

[140] Issa Qabajeh, Fadi Thabtah, and Francisco Chiclana: *A dynamic rule-induction method for classification in data mining*. Journal of Management Analytics, 2(3):233–253, 2015. (Citado en la página 64.)

[141] Yu Qiao, Sikai Chen, Majed Alinizzi, and Samuel Labi: *Modeling the relationships between pavement distress and performance*. In *Advances in Materials and Pavement Performance Prediction*, pages 11–13. CRC Press, 2018. (Citado en la página 2.)

[142] J. R. Quinlan: *Improved use of continuous attributes in c4.5*. Journal of Artificial Intelligence Research, 4:77–90, 1996. (Citado en la página 60.)

[143] J. Ross Quinlan: *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993, ISBN 1-55860-238-0. (Citado en la página 60.)

[144] Stefania C. Radopoulou and Ioannis Brilakis: *Automated detection of multiple pavement defects*. Journal of Computing in Civil Engineering, 31(2):04016057, 2017. (Citado en la página 2.)

[145] Mohammad Hossein Rafiei and Hojjat Adeli: *A novel machine learning-based algorithm to detect damage in high-rise building structures*. The Structural Design of Tall and Special Buildings, 26(18):e1400, 2017. (Citado en la página 4.)

[146] Mohammad Hossein Rafiei and Hojjat Adeli: *Novel machine-learning model for estimating construction costs considering economic variables and indexes*. Journal of Construction Engineering and Management, 144(12):04018106, 2018. (Citado en la página 4.)

[147] Antonella Ragnoli, Maria De Blasiis, and Alessandro Di Benedetto: *Pavement distress detection methods: A review*. Infrastructures, 3(4):58, 2018. (Citado en las páginas 4, 131 y 135.)

[148] Pradeep Rai and Shubha Singh: *A survey of clustering techniques*. International Journal of Computer Applications, 7(12):1–5, oct 2010. (Citado en la página 50.)

[149] J. Rama, C. Nalini, and A. Kumaravel: *Image pre-processing: enhance the performance of medical image classification using various data augmentation technique*. ACCENTS Transactions on Image Processing and Computer Vision, pages 7–14, feb 2019. (Citado en la página 10.)

[150] Chandan K. Reddy and Bhanukiran Vinzamuri: *A survey of partitional and hierarchical clustering algorithms*. In *Data Clustering*, pages 87–110. Chapman and Hall/CRC, sep 2018. (Citado en la página 51.)

[151] O. Ronneberger, P.Fischer, and T. Brox: *U-net: Convolutional networks for biomedical image segmentation*. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. (Citado en la página 88.)

[152] Rui Xu and D. Wunsch: *Survey of clustering algorithms*. IEEE Transactions on Neural Networks, 16(3):645–678, 2005. (Citado en la página 50.)

[153] D.E. Rumelhart, G.E. Hinton, and R.J. Williams: *Learning internal representations by error propagation*. In *Readings in Cognitive Science*, pages 399–421. Elsevier, 1988. (Citado en la página 66.)

[154] Omer Sagi and Lior Rokach: *Ensemble learning: A survey*. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4):e1249, 2018. (Citado en la página 71.)

[155] Claude Sammut and Geoffrey I. Webb: *Encyclopedia of Machine Learning and Data Mining*, chapter K, pages 695–697. Springer US, Boston, MA, 2nd edition, 2017, ISBN 978-1-4899-7687-1. (Citado en las páginas 51 y 57.)

[156] Nadezhda Sazonova and Stephanie Schuckers: *Fast and efficient iris image enhancement using logarithmic image processing*. In B. V. K. Vijaya Kumar, Salil Prabhakar, and Arun A. Ross (editors): *Biometric Technology for Human Identification VII*. SPIE, apr 2010. (Citado en la página 31.)

[157] Erich Schubert and Peter J. Rousseeuw: *Faster k-medoids clustering: Improving the PAM, CLARA, and CLARANS algorithms*. In *Similarity Search and Applications*, pages 171–187. Springer International Publishing, 2019. (Citado en la página 51.)

[158] Amy Schweikert, Paul Chinowsky, Xavier Espinet, and Michael Tarbert: *Climate change and infrastructure impacts: Comparing the impact on roads in ten countries through 2100*. Procedia Engineering, 78:306–316, 2014. (Citado en la página 3.)

[159] Mehmet Sezgin and Bülent Sankur: *Survey over image thresholding techniques and quantitative performance evaluation*. Journal of Electronic Imaging, 13(1):146, jan 2004. (Citado en la página 22.)

[160] Haijian Shi: *Best-first decision tree learning*. Master's thesis, University of Waikato, Hamilton, NZ, 2007. COMP594. (Citado en la página 58.)

[161] Hoo Chang Shin, Matthew R. Orton, David J. Collins, Simon J. Doran, and Martin O. Leach: *Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data*. IEEE Trans. Pattern Anal. Mach. Intell., 35(8):1930–1943, August 2013, ISSN 0162-8828. https://doi.org/10.1109/TPAMI.2012.277. (Citado en la página 75.)

[162] Connor Shorten and Taghi M. Khoshgoftaar: *A survey on image data augmentation for deep learning*. Journal of Big Data, 6(1), jul 2019. (Citado en las páginas 73 y 112.)

[163] G.T. Shrivakshan and Chandramouli Chandrasekar: *A comparison of various edge detection techniques used in image processing*. International Journal of Computer Science Issues, 9:269–276, September 2012. (Citado en la página 34.)

[164] Nazmul Siddique and Hojjat Adeli: *Fuzzy systems and applications*. In *Computational Intelligence*, chapter 3, pages 65–101. John Wiley & Sons, Ltd, 2013, ISBN 9781118534823. (Citado en las páginas 132 y 135.)

[165] Harpreet Singh, Madan M. Gupta, Thomas Meitzler, Zeng Guang Hou, Kum Kum Garg, Ashu M. G. Solo, and Lotfi A. Zadeh: *Real-life applications of fuzzy logic*. Advances in Fuzzy Systems, 2013:1–3, 2013. (Citado en las páginas 132 y 135.)

[166] P. Singh and P. A. Meshram: *Survey of density based clustering algorithms and its variants*. In *2017 International Conference on Inventive Computing and Informatics (ICICI)*, pages 920–926, 2017. (Citado en la página 52.)

[167] Yogiraj Singh and Ashish Mohan: *A survey on unsupervised clustering algorithm based on k-means clustering*. International Journal of Computer Applications, 156(8):6–9, dec 2016. (Citado en la página 50.)

[168] Rashmi Sharan Sinha, Sang Moon Lee, Minjoong Rim, and Seung Hoon Hwang: *Data augmentation schemes for deep learning in an indoor positioning application*. Electronics, 8(5):554, may 2019. (Citado en la página 73.)

[169] P. S. Sisodia, V. Tiwari, and A. Kumar: *Analysis of supervised maximum likelihood classification for remote sensing image*. In *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*, pages 1–4, 2014. (Citado en la página 90.)

[170] Douglas R. Smith: *The design of divide and conquer algorithms*. Science of Computer Programming, 5:37–58, 1985. (Citado en la página 58.)

[171] Irwin Sobel and Gary Feldman: *A 3x3 isotropic gradient operator for image processing*. A talk at the Stanford Artificial Project in, pages 271–272, 1968. (Citado en la página 35.)

[172] Marina Sokolova, Nathalie Japkowicz, and Stan Szpakowicz: *Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation*. In *Proceedings of the 19th Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence*, AI'06, page 1015–1021, Berlin, Heidelberg, 2006. Springer-Verlag, ISBN 3540497870. (Citado en la página 78.)

[173] Jose Manuel Soto Hidalgo: *Desarrollo de modelos difusos para representar la semántica del color*. Tesis de Doctorado, Universidad de Granada, 2014. (Citado en la página 12.)

[174] Billie F. Spencer, Vedhus Hoskere, and Yasutaka Narazaki: *Advances in computer vision-based civil infrastructure inspection and monitoring*. Engineering, 5(2):199–222, apr 2019. (Citado en las páginas 4 y 10.)

[175] Marcin Staniek: *Detection of cracks in asphalt pavement during road inspection processes*. Scientific Journal of Silesian University of Technology. Series Transport, 96:175–184, sep 2017. (Citado en la página 3.)

[176] M. Stone: *Cross-validatory choice and assessment of statistical predictions (with discussion)*. Journal of the Royal Statistical Society: Series B (Methodological), 38(1):102–102, 1976. (Citado en la página 62.)

[177] Alaa Tharwat: *Classification assessment methods*. Applied Computing and Informatics, 2018, ISSN 2210-8327. (Citado en las páginas 76 y 77.)

[178] The Advanced Informatics Research Group: *Road crack dataset*, 2019. [En línea]. Disponible en:. https://www.uco.es/giia/road-crack-dataset/, [Accedido el 23 de Mayo de 2020]. (Citado en la página 112.)

[179] Gabriel Thomas: *Image segmentation using histogram specification*. In *2008 15th IEEE International Conference on Image Processing*. IEEE, 2008. (Citado en la página 42.)

[180] C. Tomasi and R. Manduchi: *Bilateral filtering for gray and color images*. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pages 839–846, 1998. (Citado en la página 29.)

[181] Thomas P. Trappenberg: *Regression and optimization*. In *Fundamentals of Machine Learning*, pages 93–120. Oxford University Press, nov 2019. (Citado en la página 55.)

[182] USFWS: *Aves de Tángara, libres de derecho de autor*, 2016. [En línea]. Disponible en:. https://pixnio.com/es/animales/aves/aves-tanager/colorido-occidental-tanager-rama#, [Accedido el 5 de Abril de 2020]. (Citado en la página 10.)

[183] M. A. Vega-Rodriguez: *Review: Feature extraction and image processing*. The Computer Journal, 47(2):271–272, feb 2004. (Citado en la página 10.)

[184] Priyanka Verma and Rajeev Kumar: *A literature survey on classification algorithms of machine learning*. International Journal of Computer Applications, 179(53):47–50, jun 2018. (Citado en la página 55.)

[185] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre Antoine Manzagol: *Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion*. J. Mach. Learn. Res., 11:3371–3408, December 2010, ISSN 1532-4435. (Citado en la página 75.)

[186] Baoxian Wang, Yiqiang Li, Weigang Zhao, Zhaoxi Zhang, Yufeng Zhang, and Zhe Wang: *Effective crack damage detection using multilayer sparse feature representation and incremental extreme learning machine*. Applied Sciences, 9(3):614, 2019. (Citado en la página 88.)

[187] Feng Wang, Gui Tang Wang, Rui Huang Wang, and Xiao Wu Huang: *FPGA implementation of laplacian of gaussian edge detection algorithm*. Advanced Materials Research, 282-283:157–160, jul 2011. (Citado en la página 37.)

[188] Teng Wang, Kasthurirangan Gopalakrishnan, Omar Smadi, and Arun K. Somani: *Automated shape-based pavement crack detection approach*. Transport, 33(3):598–608, 2018. (Citado en la página 87.)

[189] Xuedong Wang, Tiancheng Li, Shudong Sun, and Juan M. Corchado: *A survey of recent advances in particle filters and remaining challenges for multitarget tracking*. Sensors, 17(12), 2017, ISSN 1424-8220. (Citado en la página 87.)

[190] Jie Wei, Hesheng Zhang, and Limin Jia: *State monitoring approach based on improved principal component analysis*. Journal of Electronic Measurement and Instrument, 2009(7):51–55, dec 2009. (Citado en la página 53.)

[191] Tian Xishan: *A novel image edge detection algorithm based on prewitt operator and wavelet transform*. International Journal of Advancements in Computing Technology, 4(19):73–82, oct 2012. (Citado en la página 36.)

[192] Guan Xu, Fang Chen, Guangwei Wu, and Xiaotao Li: *Active solution of homography for pavement crack recovery with four laser lines*. Scientific Reports, 8(1), 2018. (Citado en las páginas 88 y 90.)

[193] Yun Xu and Royston Goodacre: *On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning*. Journal of Analysis and Testing, 2(3):249–262, jul 2018. (Citado en la página 80.)

[194] S. Yadav and S. Shukla: *Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification*. In *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pages 78–83, 2016. (Citado en la página 81.)

[195] Lijuan Yan and Yanshen Liu: *An ensemble prediction model for potential student recommendation using machine learning*. Symmetry, 12(5):728, may 2020. (Citado en la página 104.)

[196] Qun Yang and Shishi Zhou: *Identification of asphalt pavement transverse cracking based on vehicle vibration signal analysis*. Road Materials and Pavement Design, pages 1–19, jan 2020. (Citado en la página 3.)

[197] Takahiro Yano and Yoshimitsu Kuroki: *Fast implementation of gaussian filter by parallel processing of binominal filter*. In *2016 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*. IEEE, oct 2016. (Citado en la página 28.)

[198] Min Yao and Changming Zhu: *Study and comparison on histogram-based local image enhancement methods*. In *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*. IEEE, jun 2017. (Citado en la página 42.)

[199] Yongchang Wang and L. Zhu: *Research and implementation of svd in machine learning*. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, pages 471–475, 2017. (Citado en la página 53.)

[200] Chen Yu, Chen Dian-ren, Li Yang, and Chen Lei: *Otsu's thresholding method based on gray level-gradient two-dimensional histogram*. In *Proceedings of the 2Nd International Asia Conference on Informatics in Control, Automation and Robotics - Volume 3*, CAR'10, pages 282–285, Piscataway, NJ, USA, 2010. IEEE Press, ISBN 978-1-4244-5192-0. (Citado en la página 25.)

[201] Ruichuan Zhang and Nora El-Gohary: *Big data and machine learning*. In *Computing in Civil Engineering 2019*. American Society of Civil Engineers, jun 2019. (Citado en la página 4.)

[202] Zhi Hua Zhou: *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 1st edition, 2012, ISBN 1439830037, 9781439830031. (Citado en la página 71.)

[203] Li Zhuo, Xiaochen Hu, Liying Jiang, and Jing Zhang: *A color image edge detection algorithm based on color difference*. Sensing and Imaging, 17(1), aug 2016. (Citado en la página 33.)

[204] Şaban Öztürk and Bayram Akdemir: *Comparison of edge detection algorithms for texture analysis on glass production*. Procedia - Social and Behavioral Sciences, 195:2675–2682, jul 2015. (Citado en la página 32.)