UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Geodésica, Cartográfica y Topográfica

Detección de la línea de costa a partir de imágenes satelitales y algoritmos de aprendizaje profundo

Trabajo Fin de Grado

Grado en Ingeniería Geomática y Topografía

AUTOR/A: Correas Naranjo, Luis

Tutor/a: Palomar Vázquez, Jesús Manuel

Director/a Experimental: PASTOR NARANJO, FRANCISCO

CURSO ACADÉMICO: 2022/2023

Agradecimientos

Me gustaría empezar agradeciendo a la vida por dejarme disfrutar de lo más importante que tengo, mis padres y hermanos, que nunca han dejado de mostrarme su amor y apoyo incondicional. Habéis sido y sois un regalo invaluable y estoy agradecido por cada momento que hemos vivido juntos. Gracias por ser mi familia y por estar siempre a mi lado. Os quiero mucho.

A mis amigos y seres queridos, por vuestra paciencia y comprensión cuando dejé planes de lado para dedicarme a este proyecto.

A mis compañeros de carrera, especialmente a Alejandro, Raúl y Andreea, por compartir este camino conmigo y ser una fuente constante de motivación. Juntos superamos desafíos y celebramos logros, creando recuerdos inolvidables.

A mis profesores, por su sabia orientación y conocimientos compartidos. Su guía fue fundamental para mi crecimiento académico y personal.

A mis tutores, Jesús y Francisco. Vuestra dedicación y compromiso han sido cruciales en mi crecimiento y consecución de mis metas, aunque quiero hacer una mención especial en Fran, mi gran apoyo y el que me ha sufrido día a día. Muchas gracias por aguantarme y convertir un trabajo como este en una aventura continua. Si de algo estoy seguro es de que llegarás dónde te propongas, no tienes límites.

Y por supuesto, no puedo olvidar mencionar a Valery, mi tía, la mejor profesora e investigadora que tiene la universidad. Has sido mi consejera y motivación constante. Tu apoyo inquebrantable y tus palabras de aliento han sido el combustible que me ha impulsado a superar obstáculos y a creer en mis propias capacidades. Tu amor y dedicación no tienen comparación, y te estaré eternamente agradecido por todo lo que has hecho por mí.

Muchas gracias a todos.

Compromiso

El presente documento ha sido realizado completamente por el firmante; no ha sido entregado como otro trabajo académico previo y todo el material tomado de otras fuentes ha sido convenientemente entrecomillado y citado su origen en el texto, así como referenciado en la bibliografía.

Valencia, 6 de julio de 2023

Resumen

El seguimiento y monitorización de la línea de costa a lo largo del tiempo ha sido un factor determinante para el desarrollo social y económico de la zona costera. El cambio climático, entre otros factores, ha provocado que este trabajo sea ya no importante, sino necesario teniendo en cuenta que, por ejemplo, el turismo en España supone aproximadamente un 10 % del PIB, por lo que el seguimiento de la evolución de los sectores costeros es un factor fundamental en el desarrollo de esta actividad económica. El avance tecnológico y puesta en marcha de programas como Copernicus de observación de la tierra por parte de la Unión Europea en los últimos años, ha permitido el acceso de manera gratuita a datos obtenidos mediante observación por satélite, fomentando las investigaciones científicas, el avance tecnológico y la aplicación económica de medios más eficientes y productivos. Una de estas novedosas tecnologías es la inteligencia artificial.

En este Trabajo Fin de Grado pretendo demostrar el enorme potencial que la IA, más concretamente el aprendizaje profundo, tiene en el procesamiento y análisis de imágenes obtenidas desde plataformas espaciales, como satélites, así como las grandes ventajas que tiene respecto de las técnicas tradicionales, mediante la creación de un modelo de red neuronal que sea capaz de detectar, de manera automática e instantánea, la línea de costa a partir de una imagen, sin necesidad de realizar ninguna otra operación, de forma que ayude a la mejora de los sistemas automáticos tradicionales (normalmente basados en la obtención de índices de agua). Algunas de las ventajas más importantes que proporciona el uso de esta tecnología son:

- Escalabilidad: Las redes neuronales pueden procesar grandes cantidades de datos en paralelo, lo que las hace más eficientes para el procesamiento de grandes áreas y conjuntos de datos.

- Flexibilidad: Las redes neuronales pueden ser entrenadas con diferentes tipos de datos, incluyendo imágenes satelitales de diferentes resoluciones y bandas espectrales. Esto permite una mayor flexibilidad en la detección de la línea de costa en diferentes condiciones.

- Automatización: Una vez que se ha entrenado una red neuronal, puede ser utilizada para automatizar la detección de la línea de costa en nuevos conjuntos de datos, lo que ahorra tiempo y recursos.

- Generalización: Las redes neuronales pueden generalizar la detección de la línea de costa a diferentes condiciones climáticas y geográficas, lo que las hace más robustas que los algoritmos de detección basados en reglas predefinidas.

Resum

El seguiment i monitoratge de la línia de costa al llarg del temps ha sigut un factor determinant per al desenvolupament social i econòmic de la zona costanera. El canvi climàtic, entre altres factors, ha provocat que aquest treball siga ja no important, sinó necessari tenint en compte que, per exemple, el turisme a Espanya suposa aproximadament un 10 % del PIB, per la qual cosa el seguiment de l'evolució dels sectors costaners és un factor fonamental en el desenvolupament d'aquesta activitat econòmica. L'avanç tecnològic i posada en marxa de programes com Copernicus d'observació de la terra per part de la Unió Europea en els últims anys, ha permés l'accés de manera gratuïta a dades obtingudes mitjançant observació per satèl·lit, fomentant les investigacions científiques, l'avanç tecnològic i l'aplicació econòmica de mitjans més eficients i productius. Una d'aquestes noves tecnologies és la intel·ligència artificial.

En aquest Treball Fi de Grau pretenc demostrar l'enorme potencial que la IA, més concretament l'aprenentatge profund, té en el processament i anàlisi d'imatges obtingudes des de plataformes espacials, com a satèl·lits, així com els grans avantatges que té respecte de les tècniques tradicionals, mitjançant la creació d'un model de xarxa neuronal que siga capaç de detectar, de manera automàtica i instantània, la línia de costa a partir d'una imatge, sense necessitat de realitzar cap altra operació, de manera que ajude a la millora dels sistemes automàtics tradicionals (normalment basats en l'obtenció d'índexs d'aigua). Algunes dels avantatges més importants que proporciona l'ús d'aquesta tecnologia són:

- Escalabilitat: Les xarxes neuronals poden processar grans quantitats de dades en paral·lel, la qual cosa les fa més eficients per al processament de grans àrees i conjunts de dades.

- Flexibilitat: Les xarxes neuronals poden ser entrenades amb diferents tipus de dades, incloent-hi imatges satel·litàries de diferents resolucions i bandes espectrals. Això permet una major flexibilitat en la detecció de la línia de costa en diferents condicions.

- Automatització: Una vegada que s'ha entrenat una xarxa neuronal, pot ser utilitzada per a automatitzar la detecció de la línia de costa en nous conjunts de dades, la qual cosa estalvia temps i recursos.

- Generalització: Les xarxes neuronals poden generalitzar la detecció de la línia de costa a diferents condicions climàtiques i geogràfiques, la qual cosa les fa més robustes que els algorismes de detecció basats en regles predefinides.

Abstract

The tracking and monitoring of the coastline over time has been a determining factor in the social and economic development of the coastal area. Climate change, among other factors, has meant that this work is no longer important, but necessary, bearing in mind that, for example, tourism in Spain accounts for approximately 10 % of GDP, so that monitoring the evolution of coastal sectors is a fundamental factor in the development of this economic activity. Technological progress and the implementation of programmes such as Copernicus for earth observation by the European Union in recent years has allowed free access to data obtained by satellite observation, promoting scientific research, technological progress and the economic application of more efficient and productive means. One of these novel technologies is artificial intelligence.

In this Final Degree Project I intend to demonstrate the enormous potential that AI, more specifically deep learning, has in the processing and analysis of images obtained from space platforms, such as satellites, as well as the great advantages it has over traditional techniques, by creating a neural network model that is able to detect, automatically and instantaneously, the coastline from an image, without the need to perform any other operation, in a way that helps to improve traditional automatic systems (usually based on obtaining water indices). Some of the most important advantages provided by the use of this technology are:

- Scalability: Neural networks can process large amounts of data in parallel, making them more efficient for processing large areas and datasets.
- Flexibility: Neural networks can be trained on different types of data, including satellite images of different resolutions and spectral bands. This allows for greater flexibility in shoreline detection under different conditions.
- Automation: Once a neural network has been trained, it can be used to automate shoreline detection on new datasets, saving time and resources.
- Generalisation: Neural networks can generalise shoreline detection to different climatic and geographical conditions, making them more robust than predefined rule-based detection algorithms.

Índice general

Índice de figuras

Índice de cuadros

Listado de siglas

Adam	Adaptive Moment Estimation
AWEI	Automated Water Extraction Index
BCE	Binary Cross Entropy
BOA	Bottom of Atmosphere
CGAT	Cartografia GeoAmbiental y Teledeteccion
CNN	Convolutional neural network
CVBLab	Computer Vision and Behaviour Analysis Lab
DEM	Digital Elevation Model
GEE	Google Earth Engine
GNSS	Global navigation satellite system
HSV	Hue / Saturation / Value
IA	Inteligencia Artificial
IEEE	Institute of Electrical and Electronics Engineers
IPCC	Intergovernmental Panel on Climate Change
LUT	Look Up Table
LiDAR	Light Detection and Ranging
MNDWI	Modified Normalized Difference Water Index

MSI	MultiSpectral Instrument
MATLAB	Matrix laboratory
NDWI	Normalized Difference Water Index
NPDI	Normalized Difference Pond Index
PIB	Producto Interior Bruto
ReLu	Rectified Linear Unit
RGB	Red / Green / Blue
ROI	Region of interest
RMSP	Root Mean Square Propagation
SCL	Scene Classification
SR	Surface Reflectance
SSH	Secure Shell
SVM	Support Vector Machine
SWED	Sentinel-2 Water Edges Dataset
TFG	Trabajo Fin de Grado
UE	Unión Europea
UAV	Unmanned Aerial Vehicle
UTM	Universal Transverse Mercator
VHR	Very High Resolution
WGS	World Geodetic System

Parte I

Memoria

Capítulo 1

Introducción

1.1. Contexto y justificación del proyecto

Históricamente, los seres humanos se han sentido atraídos a establecerse en zonas costeras debido a, entre otros factores, los suelos fértiles, abundancia de alimentos y oportunidades de transporte y comercio que ofrecen estas regiones. Hoy en día, el 41 % de la población mundial vive a menos de 100 km de la costa [1], enfrentándose a los riesgos naturales y antropogénicos que amenazan tanto a las poblaciones humanas como a los ecosistemas que dependen de ellas [2]. Además, los últimos informes del Intergovernmental Panel on Climate Change (IPCC) indican que las regiones costeras son particularmente sensibles a los impactos del cambio climático [3], aumentando considerablemente su vulnerabilidad. Por esta razón, la disponibilidad de datos detallados y actualizados sobre la morfología de la línea de costa, definida como la forma o contorno de la frontera entre la tierra y el agua, es fundamental para entender y desarrollar medidas efectivas de gestión a largo plazo, siendo uno de los indicadores *proxy* más utilizados para la detección de los procesos de erosión o acreción costera [4]. Existen dos fuentes de datos principales para estudiar su morfología:

- **La medición directa en el terreno [5]**. La medición *in situ* proporciona los datos más precisos y detallados, pero es viable solo en áreas pequeñas debido a las limitaciones logísticas y de tiempo. Algunos de los métodos más usados son UAV, LiDAR, GNSS o fotografías aéreas.

- **Imágenes de satélite y su explotación mediante técnicas de teledetección [6]**. La disposición de imágenes satélite como las series Landsat y Sentinel-2, permiten el seguimiento continuado de grades áreas de la franja costera. Si bien su resolución espacial (60-10 m/px) supone una cierta desventaja respecto de los métodos de medición directa, esta desventaja se ve ampliamente compensada por otros aspectos, como su gratuidad, cobertura global y alta resolución temporal (2-5 días si se combinan ambas series). Esta es la fuente de datos con la que se va a trabajar en este proyecto.

En las últimas décadas, la tecnología de percepción remota mediante satélites ha experimentado grandes avances en términos de resolución espacial, temporal y espectral, permitiendo obtener imágenes satelitales con mayor detalle, frecuencia y calidad, lo que ha mejorado significativamente la información que podemos obtener de la superficie terrestre. La detección de la línea de costa es una tarea para la cual estas imágenes son esenciales.

Estos avances se han logrado gracias a misiones como Copernicus, iniciativa liderada por la Unión Europea (UE) para desarrollar un sistema de observación de la Tierra mediante satélites cuyas características los hacen muy valiosos para la investigación y la toma de decisiones. Cada satélite se enfoca en diferentes áreas y utiliza diferentes tecnologías para recopilar datos. A continuación, se detallan características de los dos satélites con los que se va a trabajar.

- **Sentinel-2A** [7]: Lanzado en junio de 2015, Sentinel-2A es uno de los satélites que forma parte de la misión Copernicus. Proporciona imágenes ópticas de alta resolución de la superficie terrestre en 13 bandas espectrales, que van desde el espectro visible hasta el infrarrojo cercano. Tiene una anchura de barrido de 290 km y una resolución de 10 m para las bandas espectrales visibles e infrarroja cercana, y de 20 m para la banda infrarroja de onda corta (Tabla 1.1). El satélite tiene una órbita polar baja, que le permite cubrir todo el planeta en cinco días, y heliosíncrona, es decir, sincronizada con la posición del Sol, por lo que la captura de imágenes será siempre bajo las mismas condiciones de iluminación.

- **Sentinel-2B** [8]: Lanzado en marzo de 2017, Sentinel-2B es el segundo satélite de la serie Sentinel-2. Al igual que Sentinel-2A, proporciona imágenes ópticas de alta resolución de la superficie terrestre en 13 bandas espectrales, que van desde el visible hasta el infrarrojo cercano. Tiene una anchura de barrido de 290 km y una resolución de 10 m para las bandas espectrales visibles e infrarroja cercana, y de 20 m para la banda infrarroja de onda corta (Tabla 1.1). Al igual que su predecesor, Sentinel-2B tiene una órbita polar baja y heliosíncrona.

Banda	Descripción	Longitud de onda (nm)	Resolución espacial (m)
1	Coastal	443	60
2	Blue	490	10
3	Green	560	10
4	Red	665	10
5	Red Edge 1	705	20
6	Red Edge 2	740	20
7	Red Edge 3	783	20
8	NIR	842	10
8A	Narrow NIR	865	20
9	Water vapor	940	60
10	SWIR - Cirrus	1375	60
11	SWIR 1	1610	20
12	SWIR 2	2190	20

Tabla 1.1: Información de las bandas de Sentinel-2

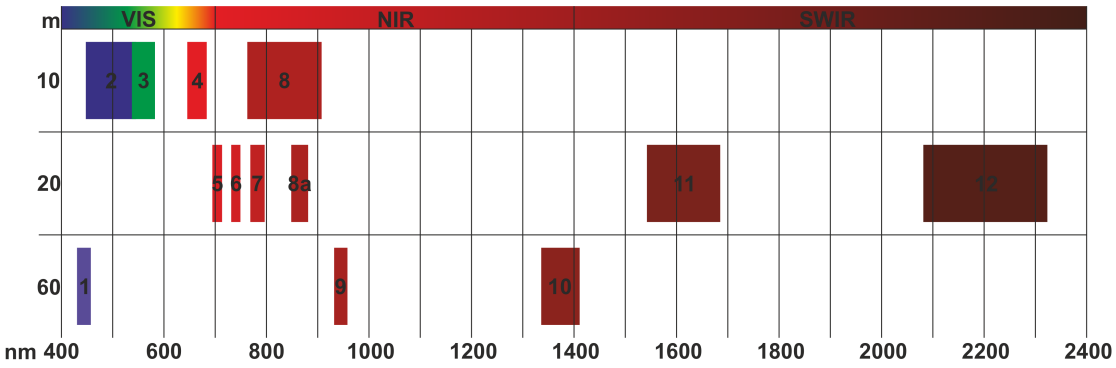


Figura 1.1: Bandas de Sentinel-2 y su relación con el espectro electromagnético. Fuente: [9]

Estos satélites están perfectamente sincronizados entre sí. Cada uno tarda aproximadamente 100 minutos para completar una órbita completa alrededor de la Tierra. Sin embargo, debido a la simetría de sus órbitas, la posición relativa de los dos satélites siempre es la misma. Esto significa que cada vez que Sentinel-2A completa una órbita, Sentinel-2B lo hace también, encontrándose en la misma posición relativa que antes.

Esta sincronización es extremadamente útil, ya que permite que los dos satélites proporcionen datos complementarios que se pueden usar juntos para obtener una cobertura más completa y precisa de la superficie terrestre. Al trabajar juntos, los satélites Sentinel-2A y Sentinel-2B pueden tomar imágenes de la misma área de la Tierra con una diferencia de tiempo muy corta.

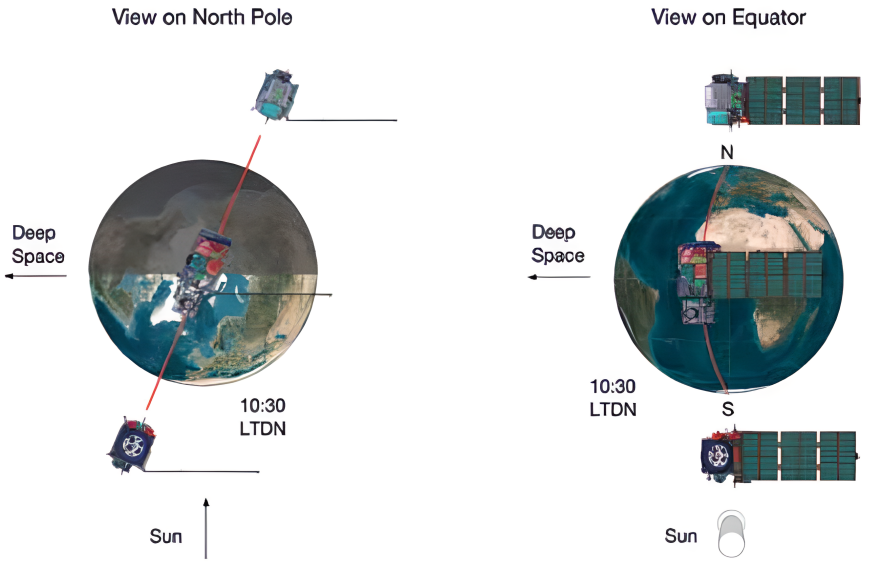


Figura 1.2: Representación gráfica ilustrativa de la órbita simétrica de los satélites Sentinel-2A y Sentinel-2B. Fuente: [10]

1.2. Estado del arte

A lo largo del tiempo, se han utilizado técnicas de procesamiento de imágenes para identificar la línea de costa, tales como detección de bordes, segmentación de objetos, filtrado de ruido, etc. Estas técnicas se basan en la extracción de características de la imagen para distinguir la interfaz agua-no agua. Algunos de los métodos más comunes son:

- **Umbralización [11, 12]**: Este método es uno de los más sencillos. Consiste en seleccionar un umbral de intensidad de imagen que permita distinguir entre los píxeles de agua y los de no agua. Normalmente este tipo de métodos se aplican sobre bandas donde el agua suele tener una alta absorción (baja reflectividad) y la tierra una alta reflectividad (baja absorción), por lo que las superficies de agua presentan valores bajos (oscuros), mientras que la tierra presenta valores altos (claros). Los algoritmos tradicionales se basan en la selección manual iterativa del umbral o en métodos más sofisticados de selección automática del umbral óptimo, como el método Otsu, entre otros.

- **Métodos de bordes [13]**: Basado en la detección de bordes, utiliza técnicas de procesamiento, tales como la detección de bordes de *Canny*, para identificar los cambios abruptos en la intensidad de la imagen. La detección de bordes se utiliza para extraer la información sobre la ubicación y la forma de la costa para posteriormente aplicar técnicas de segmentación.

- **Modelos matemáticos [14, 15]**: Existen varios modelos matemáticos que se pueden utilizar para modelar la forma de la costa. Uno de los más comunes es el modelo de la línea de costa fractal (*fractal coastline model*). En el contexto de la costa, la estructura fractal se refiere a la propiedad de la línea de la costa de tener una forma irregular y muy ramificada, que presenta detalles a diferentes escalas. Por ejemplo, la costa puede tener una forma ondulada y sinuosa, con bahías, penínsulas, y otros detalles de menor tamaño, que se repiten en diferentes escalas. Este modelo se basa en la idea de que la longitud de la línea de la costa depende de la escala utilizada para medirla.

- **Métodos basados en índices espectrales [16, 17]**: Este método se basa en la extracción de características relevantes de la imagen para la detección de la línea de costa y segmentación de la interfaz agua-no agua, como la textura, forma y color. Para ello, es muy común el uso de índices como el NDWI, AWEISH, AWEINSH, MNDWI, entre otros, que se basan en la extracción de características como la reflectancia en diferentes longitudes de onda, permitiendo distinguir entre la superficie del agua y la superficie terrestre en la imagen. Una vez extraídas, se pueden utilizar algoritmos de clasificación, como el *k-means* o el *SVM*, para segmentar la imagen.

En los últimos años han surgido nuevas técnicas de segmentación que utilizan métodos de aprendizaje profundo, como redes neuronales convolucionales (CNNs) [18], capaces de procesar grandes cantidades de datos y aprender a identificar patrones y características que sean relevantes para la tarea en cuestión. De hecho, los métodos tradicionales suelen ser menos precisos que los enfoques CNNs, especialmente cuando se trata de segmentar imágenes complejas o con ruido.

En 2020, en el *GRSS Data Fusion Contest* organizado anualmente por el IEEE, se propuso una prometedora metodología [19] para la detección de costa utilizando redes neuronales y fusión de datos. Este trabajo fue reconocido como una de las mejores contribuciones en un concurso enfocado en la resolución de problemas de fusión de datos de múltiples fuentes, como imágenes de satélite, datos LiDAR y datos de sensores terrestres.

A lo largo del proyecto se ha hecho uso de métodos más clásicos, como los índices comentados anteriormente, para comparar los resultados obtenidos mediante algoritmos de redes neuronales. En concreto, las expresiones matemáticas de los índices usados son:

$$\text{NDWI} = \frac{\text{Green} - \text{NIR}}{\text{Green} + \text{NIR}} \tag{1.1}$$

$$\text{MNDWI} = \frac{\text{Green} - \text{SWIR1}}{\text{Green} + \text{SWIR1}} \tag{1.2}$$

$$\text{AWEISH} = \text{Blue} + (2{,}5 \times \text{Green}) - (1{,}5 \times (\text{NIR} + \text{SWIR1})) - (0{,}25 \times \text{SWIR2}) \tag{1.3}$$

$$\text{AWEINSH} = 4 \times (\text{Green} - \text{SWIR1}) - (0{,}25 \times \text{NIR} + 2{,}75 \times \text{SWIR2}) \tag{1.4}$$

1.3. Objetivos

En este Trabajo Fin de Grado se pretende diseñar un modelo basado en red neuronal capaz de detectar la línea de costa a nivel de píxel a partir de imágenes satelitales aplicando técnicas de aprendizaje profundo para realizar una segmentación precisa y automatizada.

Para lograr este objetivo, se utilizará como base de partida una red neuronal convolucional (CNN) y se entrenará, tal y como se comenta en el apartado 2.1, con un conjunto de datos etiquetados que incluye imágenes satelitales y sus correspondientes máscaras de segmentación usadas como verdad terreno. La CNN será capaz de aprender las características importantes de las imágenes para predecir la segmentación de la interfaz agua-no agua en nuevas imágenes.

Se explorarán diferentes arquitecturas y parámetros de entrenamiento, como el tamaño del lote, la tasa de aprendizaje, el optimizador y el número de épocas, para obtener un modelo preciso y generalizable. Además, se evaluará la calidad de las segmentaciones utilizando las métricas de evaluación *Accuracy*, *Precision*, *Recall*, coeficiente Kappa, puntuación F1, puntuación *Dice* y el coeficiente de correlación *Matthew*, comparando el resultado con métodos de segmentación tradicionales basados en índices de agua como los vistos anteriormente.

Por último, como objetivo especifico se busca construir una base de datos de elaboración propia mediante la plataforma Google Earth Engine compuestas únicamente por imágenes costeras para entrenar y validar los modelos.

1.4. Organización del escrito

A continuación, se presenta la estructura de esta memoria que documenta el trabajo realizado en este Trabajo Fin de Grado.

- **Capítulo 1**.

  En este capítulo se aborda el contexto y la justificación del proyecto, donde se proporciona una descripción del entorno en el que se desarrolla el estudio y se destacan los problemas que se pretenden abordar; el estado del arte, en el cual se realiza una revisión de los trabajos y artículos de investigaciones existentes que están relacionados directamente con el proyecto; y los objetivos, donde se establecen las metas específicas que se pretenden lograr con el proyecto.

- **Capítulo 2**.

  El capítulo 2 tiene como finalidad presentar una descripción de las bases de datos utilizadas para el entrenamiento del modelo. Se detallan las características de las bases de datos, incluyendo el tamaño, la diversidad de muestras, la calidad de los datos y otros atributos relevantes. También se mencionan las fuentes de las bases de datos, indicando si son bases de datos públicas, privadas o generadas específicamente para el proyecto.

- **Capítulo 3**.

  Este capítulo describe los pasos seguidos para la realización del modelo de red neuronal, así como las técnicas de preprocesado llevadas a cabo.

- **Capítulo 4**.

  En el capítulo 4 se analizan los resultados obtenidos y se comparan con otros conseguidos a partir de otros modelos de segmentación y otras metodologías más clásicas tales como índices de detección de masas de agua.

- **Capítulo 5**.

  Para finalizar, se realiza una breve conclusión donde se analizan los resultados obtenidos y, además, se proponen mejoras de futuro.

Capítulo 2

Datos

2.1. Entorno de trabajo

A continuación, se detalla el lenguaje de programación y *softwares* que han hecho posible el desarrollo de este Trabajo Fin de Grado.

2.1.1. Python y librerías empleadas

Python es un lenguaje de programación usado en multitud de campos, de alto nivel y de código abierto, creado por Guido van Rossum en la década de 1980. Se destaca por su legibilidad y estructura visual del código, lo que facilita su comprensión y escritura. De hecho, se ha convertido en el lenguaje de programación más utilizado en los últimos años gracias, en gran medida, a su extensa biblioteca que proporciona una amplia gama de módulos y herramientas para diversas tareas, tal y como se podrá ver a continuación.

Pytorch

PyTorch es una biblioteca tensorial optimizada de aprendizaje profundo basada en *Python* y *Torch* y se utiliza principalmente para aplicaciones que utilizan *GPU* y *CPU*. Aunque existen otras librerías orientadas a este campo como *TensorFlow* y *Keras*, *Pytorch* es la más usada ya que utiliza gráficos de computación dinámica y es completamente *Pythonic*, es decir, permite ejecutar y probar partes del código en tiempo real.

Matplotlib

Matplotlib es una librería de visualización de datos en *Python*. Proporciona una amplia variedad de gráficos y herramientas para crear visualizaciones estáticas, interactivas y animadas. Es ampliamente utilizada en el campo de la ciencia de datos y la visualización de datos.

Numpy

Numpy es una librería fundamental en *Python* para el cálculo numérico y la manipulación de arreglos multidimensionales. Proporciona funciones eficientes para operaciones matemáticas y estadísticas, y es ampliamente utilizada en el procesamiento de datos científicos y análisis numérico.

Tifffile

Tifffile es una librería de *Python* que permite leer y escribir imágenes en formato TIFF. Además, proporciona funciones para acceder y manipular metadatos.

Pandas

Pandas es una librería de *Python* para el análisis y manipulación de datos. Proporciona estructuras de datos flexibles y eficientes, como los *DataFrames*, que permiten realizar operaciones de limpieza, transformación y análisis de datos de manera sencilla.

Albumentations

Albumentations es una librería de aumento de datos para el procesamiento de imágenes. Proporciona una amplia gama de transformaciones y aumentos de datos para imágenes, como recorte, rotación, cambio de brillo y contraste, entre otros. Es comúnmente utilizada para aumentar la diversidad y cantidad de datos de entrenamiento.

Scikit-learn

Scikit-learn es una librería de aprendizaje automático en *Python*. Proporciona una variedad de algoritmos de aprendizaje automático supervisado y no supervisado, así como herramientas para la evaluación y validación de modelos.

2.1.2. Softwares

MobaXterm

MobaXterm es un software que proporciona una solución completa de terminal y herramientas de red para usuarios de Windows. Combina múltiples funcionalidades, como terminal de SSH, servidor X11, cliente VNC, transferencia de archivos y más, en una única aplicación. Aunque existe una versión de pago, el programa es de uso libre. Este ha sido el entorno de desarrollo usado para programar y ejecutar el código necesario para la creación del modelo.

Sentinel Applications Platform (SNAP)

SNAP, *software* público y de libre acceso, es una plataforma de procesamiento y análisis de datos de observación de la Tierra desarrollado por la Agencia Espacial Europea (ESA) a través del programa Copernicus y se utiliza específicamente para el procesamiento de datos de los satélites Sentinel. Con SNAP se han visualizado y comprobado la calidad de las imágenes de todas las bases de datos usadas en el proyecto mediante una inspección visual.

QGIS

QGIS (Quantum GIS) es un sistema de información geográfica de código abierto que permite la visualización, edición y análisis de datos geoespaciales. Proporciona una amplia gama de herramientas y funciones para trabajar con datos geográficos, realizar análisis espaciales... Con esta herramienta se han producido los mapas disponibles en el texto. Además, con el *plugin* Serval, se han corregido manualmente las máscaras obtenidas con GEE.

2.2. Base de datos

Para el desarrollo del proyecto se han utilizado tres bases de datos: SWED, *dataset* público, para la implementación y entrenamiento del modelo; una de elaboración propia, creada a partir de imágenes descargadas de GEE con sus correspondientes máscaras; y, por último, una base de datos proporcionada por el grupo de Cartografia GeoAmbiental y Teledeteccion para su validación en diferentes entornos de la costa europea.

2.2.1. Sentinel-2 Water Edges Dataset (SWED)

La base de datos empleada para el entrenamiento del modelo de segmentación de la interfaz agua-no agua ha sido la *Sentinel-2 Water Edges Dataset (SWED)* [20]. Esta base de datos, disponible solo para proyectos de investigación, contiene una colección de imágenes capturadas por el satélite Sentinel-2 de la Agencia Espacial Europea (ESA) durante el periodo de 2017 a 2021.

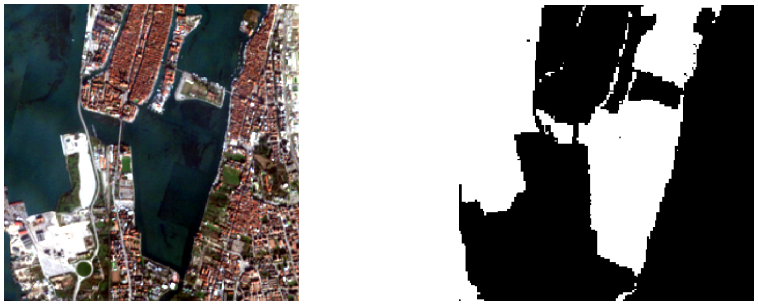
Las imágenes se obtuvieron a través del *Copernicus Open Access Hub*, filtrando por productos de nivel 2A y sin nubes. La resolución espacial más alta del Instrumento Multiespectral (MSI) seleccionado es de 10 m, por lo que cualquier banda MSI con una resolución de 20 o 60 m se remuestreó a una resolución de 10 m utilizando la técnica de interpolación del vecino más cercano (Nearest-neighbor interpolation).

Para el etiquetado del conjunto de entrenamiento, se utiliza un enfoque de *clustering* semi-supervisado mediante el *software* de uso libre *QGIS*. Se genera una imagen de falso color a partir de bandas espectrales que muestren un alto contraste entre píxeles de agua y no agua, y se aplica el método de agrupamiento *k-means clustering*. Los *clusters* se combinan hasta que solo quedan dos; uno que contiene píxeles de agua y otro que contiene píxeles de no agua. Finalmente, se realiza una comparación visual con imágenes aéreas de alta resolución y se corrigen manualmente los píxeles etiquetados incorrectamente para producir máscaras de segmentación *pixel-level*.

Otra de las ventajas de SWED es su gran cobertura geográfica, ya que incluye imágenes de diversas regiones costeras de todo el mundo, garantizando una mayor diversidad y generalización del modelo entrenado con esta base de datos.

Nombre	Sentinel-2 Water Edges Dataset
Número de muestras	28.192
Tamaño de las imágenes	256 x 256
Resolución espectral	12
Resolución espacial	10m
Año	2017 - 2021

Tabla 2.1: Base de datos para la segmentación de la interfaz agua-no agua



(a) Imagen S2 RGB (b) Máscara verdad terreno

Figura 2.1: Muestra par de imágenes (S2 y su verdad terreno) SWED. Fuente: Elaboración propia

2.2.2. Base de datos de elaboración propia - GEE

Aunque el número de imágenes del *dataset* SWED es, a priori, más que suficiente para un entrenamiento óptimo, no todas las muestras son útiles, tal y como se verá en las siguientes secciones. Además, la diversidad de condiciones atmosféricas que contiene es nula, ya que todas las imágenes se obtuvieron a partir de un filtrado de nube del 0 %. Por esto, se tomó la decisión de obtener un conjunto de imágenes adicional, aprovechando las funcionalidades de Google Earth Engine, con nuevas tomas de Sentinel-2 de zonas exclusivamente costeras de distintas geografías, añadiendo además algunas muestras que contuvieran un pequeño porcentaje de nubes (del 5 % como máximo) con el propósito de entrenar al modelo para interpretar adecuadamente las zonas de sombra que estas pudieran causar en la superficie.

Al igual que la otra base de datos, este conjunto se compone de la imagen Sentinel-2 con su resolución espectral completa y la máscara de la interfaz agua-no agua correspondiente obtenida a partir del índice NDWI. Sin embargo, como se puede ver en la figura 2.2, cada una de las máscaras obtenidas se procesaron manualmente para corregir los píxeles mal clasificados y ajustar al máximo posible la línea de costa para una correcta detección mediante la herramienta *Serval* de *QGIS*, convirtiéndolas en las verdades terreno que se usaron en el entrenamiento del modelo.

Nombre	Base de datos propia - GEE
Número de muestras	150
Tamaño de las imágenes	256 x 256
Resolución espectral	12
Resolución espacial	10m
Año	2020 - 2022

Tabla 2.2: Base de datos creada mediante la plataforma Google Earth Engine

(a) Imagen S2 (b) Máscara NDWI (c) Verdad terreno

Figura 2.2: Ejemplo de procesado de máscara en base de datos propia - GEE. La subfigura (a) corresponde a la imagen S2 original, (b) a la máscara NDWI obtenida directamente desde GEE y (c) a la máscara NDWI corregida manualmente y usada como verdad terreno. Fuente: Elaboración propia

2.2.3. Base de datos Cartografia GeoAmbiental y Teledeteccion (CGAT)

El grupo de Cartografia GeoAmbiental y Teledeteccion ha preparado una base de datos con imágenes S2 de diferentes zonas costeras con el objetivo de validar y comprobar la verdadera capacidad de generalización del modelo de aprendizaje automático generado. Esta base de datos se ha creado adecuando las características radiométricas y espaciales a la base de datos que se utilizó para entrenar el modelo, explicada en el apartado 2.1.1, lo que garantiza que los datos de validación sean consistentes y comparables.

Con respecto al etiquetado, cada una de las máscaras se ha obtenido por fotointerpretación a partir de imágenes de muy alta resolución (VHR), obtenidas desde el sistema PANDA de Copernicus, coincidente en fecha y hora con la toma del satélite S2, por lo que su precisión es mucho mayor que las vistas en las anteriores bases de datos, dado que se basa completamente en el conocimiento humano y en la capacidad de discernir detalles sutiles que los algoritmos automáticos pueden pasar por alto. Sin embargo, también es un proceso más lento y laborioso, ya que requiere mucho más tiempo para etiquetar cada píxel individualmente.

La evaluación es un paso importante para garantizar que el modelo sea robusto y generalizable a situaciones nuevas y desconocidas. Esta mayor precisión en las máscaras puede ser especialmente importante en este proceso, puesto que se conocerán los límites reales de precisión que es capaz de alcanzar.

Nombre	CGAT
Número de muestras	60
Tamaño de las imágenes	256 x 256
Resolución espectral	12
Resolución espacial	10m
Año	2023

Tabla 2.3: Base de datos proporcionada por el CGAT para la detección de la línea de costa

(a) Imagen S2 RGB (b) Máscara verdad terreno

Figura 2.3: Muestra par de imágenes (S2 y su verdad terreno) de la base de datos CGAT. Fuente: Elaboración propia

Capítulo 3

Metodología

3.1. Técnicas de preprocesado

Las imágenes usadas en las distintas bases de datos, tal y como se ha explicado, han sido sometidas a técnicas de preprocesado para mejorar su calidad y eliminar efectos perjudiciales para su interpretación. La base de datos proporcionada por el CGAT fue creada a partir de productos *Level-1C*, mientras que las imágenes de SWED y la creada con GEE se obtuvieron a partir de productos *Level-2A*. A continuación, se explica la principal diferencia entre ambos productos.

3.1.1. Corrección atmosférica

La fuente principal para la generación de productos *Level-2A* son los productos *Level-1C*. Sin embargo, la distinción clave entre ambos es la conocida como *Top-Of-Atmosphere (TOA) correction* que se le aplica a los *Level-1C*, de tal forma que el resultado que se genera deja de ser un producto de reflectancia *Bottom of Atmosphere (BOA)* para ser de Reflectancia Superficial (SR). Esta corrección consiste en determinar y subsanar las distorsiones generadas por la atmósfera en los valores de radiancia que capta el sensor provenientes de la superficie terrestre.

(a) Producto *Level-1C* (b) Producto *Level-2A*

Figura 3.1: Ejemplo entre productos *Level-1C* y *Level-2A*. Fuente: Elaboración propia

Para ello, de acuerdo con lo especificado en la guía de usuario de Copernicus [21], se usan algoritmos de procesamiento de última generación como *Sen2Cor* [22] (disponible como *plugin* en SNAP), donde a partir de un módulo SCL (Scene Classification) y con ayuda de modelos digitales de elevación (DEM), usados para determinar la incidencia angular de la luz solar y las sombras proyectadas en la superficie terrestre, se recopilan propiedades de reflectancia para definir la posible existencia o ausencia de nubes en la imagen, obteniendo como resultado un mapa de clasificación con diferentes clases (nubes, sombras, agua, nieve, vegetación...). Aunque esta funcionalidad no está directamente relacionada con la corrección atmosférica, la información recopilada por el clasificador es de gran ayuda para llevar a cabo los siguientes pasos.

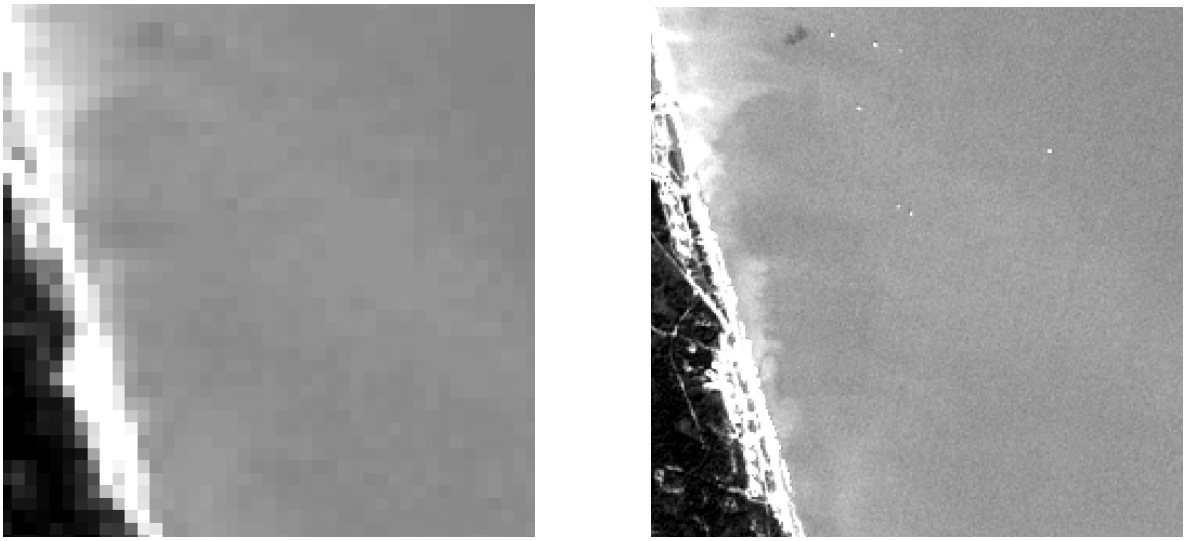
A continuación, el algoritmo procesa los efectos de dispersión y absorción atmosférica y ajusta los valores de reflectancia para obtener una estimación más precisa de la reflectancia en la superficie terrestre, con el fin de generar unas tablas de consulta que se emplearán como *Look Up Table* (LUT), mediante el *software* de uso libre *libRadtran* [23], con la información sobre los perfiles atmosféricos y los parámetros de radiación solar.

Sen2Cor, a partir de la geometría de la escena y todos los parámetros atmosféricos recopilados a lo largo del proceso (como los tipos de aerosoles, vapor de agua, etc), usa las tablas generadas junto a los valores de la reflectancia medida en los productos *Level-1C* para aplicar la corrección atmosférica.

Es importante destacar que aunque existen otros algoritmos de procesamiento, la ESA escogió este para obtener sus productos *Level-2A*, debido al buen rendimiento que se obtiene en distintas condiciones tanto geográficas como atmosféricas, además de otros factores [24].

3.1.2. Remuestreo

Sentinel-2, tal y como se puede ver en la Tabla 1.1, ofrece una amplia gama de posibilidades en lo que a resolución espectral se refiere. Sin embargo, no todas las bandas tienen la misma resolución espacial [25].



(a) Imagen Sentinel-2 de 60m / píxel (Banda 1)

(b) Imagen Sentinel-2 de 10m / píxel (Banda 2)

Figura 3.2: Imágenes Sentinel-2 de distinta resolución espacial. Fuente: Elaboración propia

Para poder trabajar con ellas de manera consistente, es necesario realizar un proceso llamado remuestreo [26], que implica ajustar la resolución espacial con el objetivo de que todas tengan el mismo tamaño de píxel.

Este proceso puede llevarse a cabo utilizando técnicas de interpolación o diezmado, dependiendo de si se desea aumentar o disminuir la resolución espacial. En Sentinel-2, se realiza comúnmente para igualar la resolución espacial a la banda con la resolución más alta, es decir, a 10 metros [26]. De esta manera, se puede garantizar una comparabilidad adecuada y aprovechar al máximo la información espectral disponible. Tal y como se ha comentado en el apartado 2.1.1, las imágenes utilizadas en el entrenamiento del modelo han sido previamente sometidas a una interpolación del vecino más cercano [20]. Esta técnica, en lugar de calcular valores intermedios mediante cálculos matemáticos complejos, asigna a cada píxel de la imagen resultante el valor del píxel más cercano en la imagen original [27].

En el caso de aumentar el tamaño de la imagen, se multiplica por el factor de interpolación el número de píxeles en cada dimensión y se asigna a cada nuevo píxel el valor del píxel más cercano en la imagen original, produciendo una imagen ampliada con un aspecto más pixelado, ya que realmente la información que contiene la imagen es la misma.

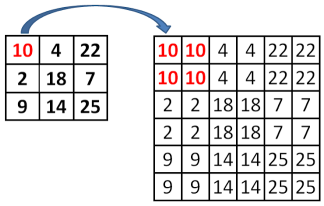


Figura 3.3: Aplicación de la técnica *Nearest-neighbor interpolation* en aumento por un factor 2. Fuente: [28]

Por otro lado, si se reduce el tamaño de la imagen, se calcula la relación de escala entre la imagen original y la imagen resultante. Luego, para cada nuevo píxel en la imagen reducida, se encuentra su píxel correspondiente en la imagen original aplicando la relación de escala y se asigna el valor del píxel más cercano. Esto produce una imagen con menos detalles y suavizada, ya que se pierde información al asignar el valor de un solo píxel cercano a múltiples píxeles en la imagen resultante. Sin embargo, si se produjera el efecto conocido como *aliasing*, como consecuencia de la desalineación entre los píxeles de la imagen original y los nuevos píxeles calculados durante la interpolación, habría que aplicar técnicas de filtrado que atenúe las altas frecuencias en la imagen.

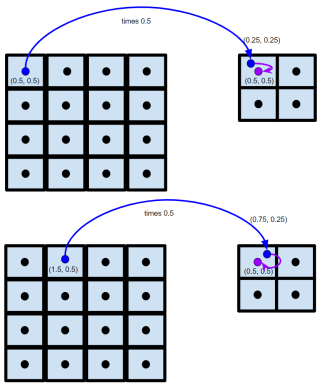


Figura 3.4: Aplicación de la técnica *Nearest-neighbor interpolation* en reducción. Fuente: [29]

Es importante tener en cuenta que la técnica de interpolación del vecino más cercano no proporciona resultados de alta calidad o suavidad en comparación con otros métodos más avanzados, como la interpolación bilineal o la interpolación bicúbica. Sin embargo, es computacionalmente más eficiente [30], lo cual es extremadamente útil con los recursos disponibles en este caso.

3.1.3. Desbalanceo entre clases

Uno de los principales problemas que afecta directamente a la estabilidad de un modelo de redes neuronales es la falta de homogeneidad en el número de muestras con presencia de unas clases y otras. En este caso, tal y como se puede apreciar en la figura 3.5, el 71 % de los píxeles con valor 0 de las máscaras verdaderas (*ground truth*) se corresponde a zonas clasificadas como 'no agua', mientras que el 29 % restante corresponde a 'agua'.

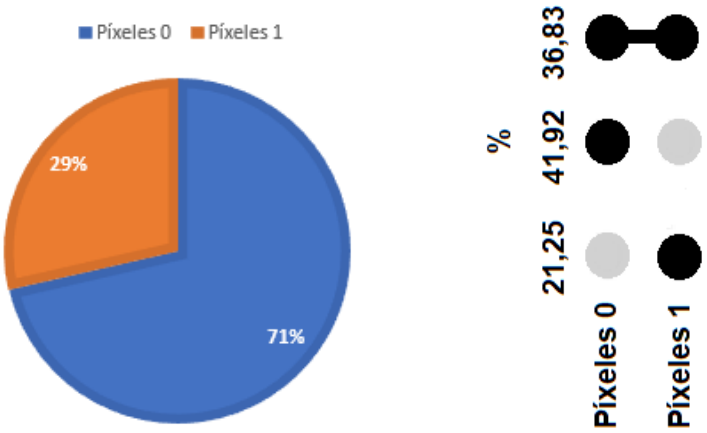


Figura 3.5: Estadística de los píxeles de la base de datos. Fuente: Elaboración propia

Este desbalanceo introducía en la propagación de pesos del entrenamiento del modelo un ruido que provocaba el empeoramiento de los resultados, por lo que se aplicó un ligero filtrado para descartar aquellas imágenes donde no existiera la presencia de alguna de las dos clases, propiciando así una mayor equidad entre el número de muestras donde ambas clases estuvieran presentes. Sin embargo, esto provocó la pérdida de, aproximadamente, el 98 % de todo el conjunto de datos, es decir, de las 28.192 imágenes disponibles, 464 fueron las destinadas al entrenamiento del modelo. Al ser un número bajo de muestras, se aplicó una técnica conocida como *data augmentation* con el fin de poder aumentar el número y diversidad de estas.

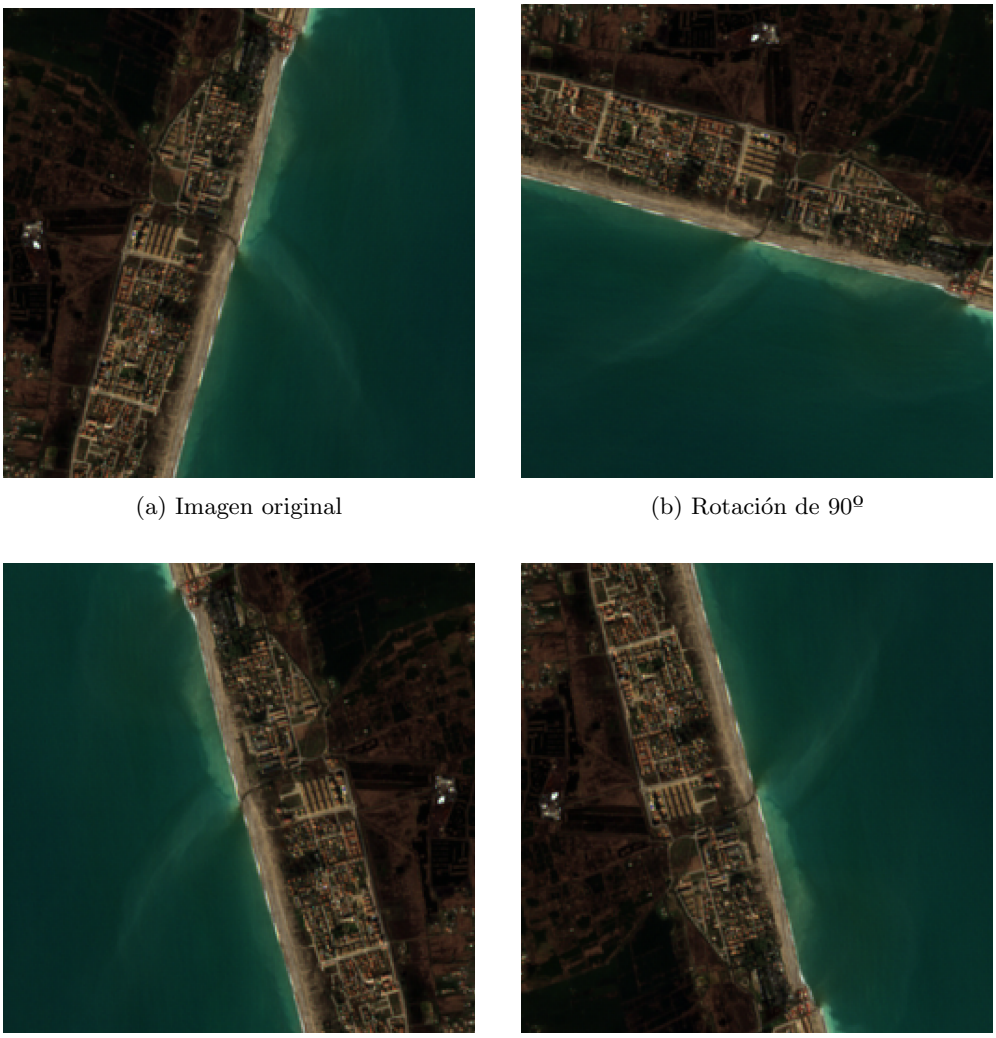
3.2. Técnicas de *Data augmentation*

Aumentar la cantidad y diversidad de imágenes que componen la base de datos encargada del entrenamiento del modelo es una práctica muy común en el ámbito del aprendizaje automático y visión por computador. Esto, conocido como *data augmentation*, ayuda a mejorar el rendimiento y la capacidad de generalización del modelo en caso de que la base de datos original no sea lo suficientemente amplia.

Tal y como se ha comentado en la sección 3.1, solucionar el desbalanceo entre clases supuso eliminar un número de imágenes bastante grande en proporción al total de la base datos, por lo que a cada una de las imágenes que superaron el filtrado se les aplicó 5 transformaciones, a través de la librería *albumentations*, como las que se van a comentar en los siguientes apartados. De esta manera, de 464 imágenes disponibles, se pasó a 2.320.

3.2.1. Volteo vertical - horizontal y rotación

Las imágenes aptas para el entrenamiento se someten a una transformación donde se voltean, tanto vertical como horizontalmente, con una probabilidad del 50 % . Además, también pueden verse sometidas a una rotación aleatoria, tal y como se muestra en la siguiente figura.



(a) Imagen original

(b) Rotación de 90º

(c) Volteo horizontal

(d) Volteo vertical

Figura 3.6: Transformaciones aplicadas a una imagen (1). Fuente: Elaboración propia

3.2.2. Cambios en la tonalidad, saturación y brillo (HSV)

A partir de unos atributos de tonalidad, brillo y saturación modificados arbitrariamente y definidos previamente a la transformación, la imagen se ve sometida a cambios en su modelo de color, aumentando la diversidad en los datos de entrenamiento. Es importante destacar que esta modificación al espacio de color de la imagen NO se aplica a la máscara, al contrario que las rotaciones, puesto que alteraría los resultados.

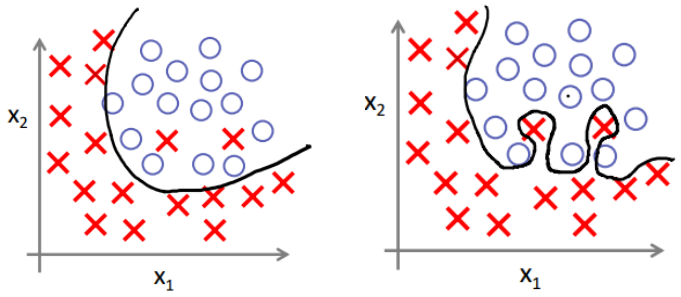
(a) Hue (Tonalidad) (b) Saturation (Saturación) (c) Value (Brillo)

Figura 3.7: Transformaciones aplicadas a una imagen (2). Fuente: Elaboración propia

El número de transformaciones aplicadas se determinó a partir de experimentos de ablación, donde se llegó a la conclusión de que un número demasiado elevado de transformaciones con los datos que se tenían causaba en el modelo un problema conocido como sobreajuste (*overfitting*).

El *overfitting* (figura 3.8) es una anormalidad que se produce durante el entrenamiento del modelo de red neuronal cuando este tiende a ajustarse demasiado a las imágenes de entrenamiento y no es capaz de generalizar correctamente a nuevas muestras, o lo que es lo mismo, el modelo simplemente recuerda las imágenes con las que ha entrenado y se olvida de aprender los patrones y características importantes del conjunto de datos.

En este caso, un número mayor de transformaciones implicaría generar muchas imágenes prácticamente iguales, por lo que, aunque el número de muestras sería mayor, la información que podría aprender el modelo apenas variaría en relación con su complejidad. El caso contrario también puede provocar el mismo problema, es decir, si la base de datos es demasiado pequeña y el modelo demasiado complejo desencadenaría en el entrenamiento del modelo un sobreajuste que afectaría negativamente al rendimiento del mismo.



(a) Caso óptimo (b) Caso con overfitting

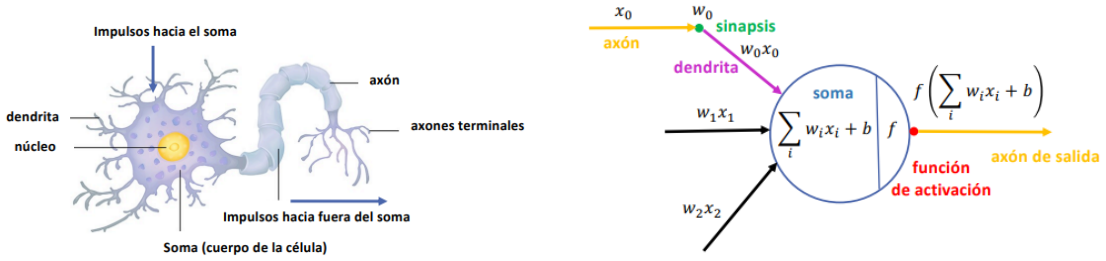
Figura 3.8: Problema de sobreajuste. Fuente: [31]

3.3. Método propuesto

3.3.1. Principios básicos de una red neuronal

Concepto de neurona

La unidad básica del cerebro humano es la neurona y de la misma manera lo es para una red neuronal. En la figura 3.9 se muestran las similitudes entre la estructura de una neurona biológica y una neurona artificial. Cada una recibe señales de entrada de sus dendritas y produce señales de salida a lo largo de su axón, el cual acaba ramificándose y conectándose mediante sinapsis a las dendritas de otras neuronas. En el modelo computacional de una neurona, las señales que viajan por los axones, x_0, interactúan multiplicativamente, $w_0 x_0$, con las dendritas de la otra neurona en función de la fuerza sináptica en esa sinapsis en concreto, w_0.

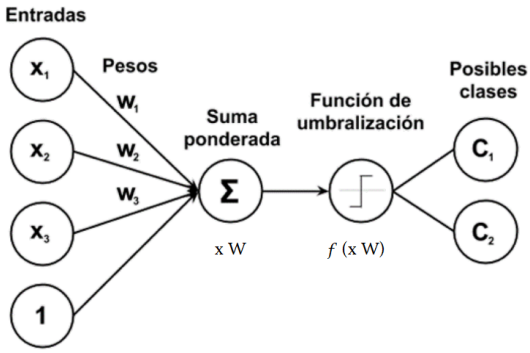


(a) Neurona biológica (b) Neurona en una red neuronal

Figura 3.9: Similitud entre una neurona humana y una neurona de una red neuronal. Fuente: [31]

Perceptrón simple

Un perceptrón es la red neuronal más simple, la que se compone de una sola neurona. El algoritmo del perceptrón (figura 3.10) fue inventado en 1958 por Frank Rosenblatt.



Siendo $x = [x_1, x_2, x_3, 1]$ y $W = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ b \end{bmatrix}$

Figura 3.10: Estructura de un perceptrón simple. Fuente: Elaboración propia

La operación principal realizada por un perceptrón simple es una suma ponderada de las entradas multiplicadas por sus respectivos pesos. Seguidamente, se aplica una función de activación, que introduce no linealidad y determina si el perceptrón se activa o no, al resultado de la suma para producir la salida que podrá utilizarse como entrada para la siguiente capa de neuronas o como resultado final de la red neuronal.

Perceptrón multicapa

Un perceptrón multicapa (MLP, por sus siglas en inglés) es una extensión del perceptrón simple que consta de múltiples capas de neuronas interconectadas. En contraste con el perceptrón simple, que solo tiene una capa de neuronas, el perceptrón multicapa puede tener una capa de entrada, una o más capas ocultas y una capa de salida.

Cada neurona está conectada con las neuronas de la capa anterior y la capa siguiente a través de conexiones ponderadas. Estas conexiones están representadas por matrices de pesos, que determinan la influencia de cada neurona en las capas anteriores en el resultado de las capas posteriores. Mediante una función de activación, como la ReLU (explicada en las siguientes líneas), se calcula la salida en función de la suma ponderada de las entradas provenientes de las neuronas de la capa anterior.

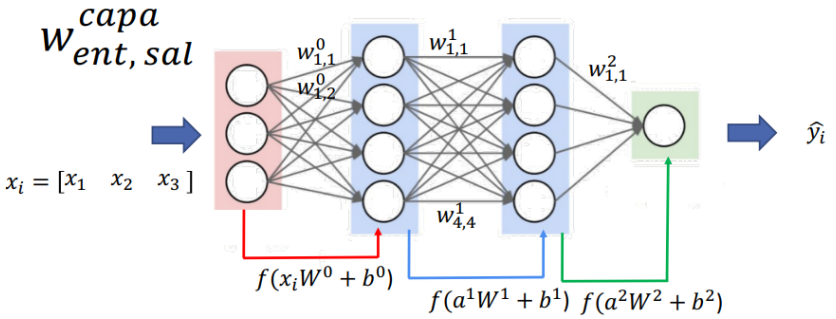


Figura 3.11: Estructura de una red neuronal. Fuente: [31]

$$\text{Siendo } W^0 = \begin{bmatrix} w^0_{1,1} & w^0_{1,2} & w^0_{1,3} & w^0_{1,4} \\ w^0_{2,1} & w^0_{2,2} & w^0_{2,3} & w^0_{2,4} \\ w^0_{3,1} & w^0_{3,2} & w^0_{3,3} & w^0_{3,4} \end{bmatrix}_{3\times4}, W^1 = \begin{bmatrix} w^1_{1,1} & w^1_{1,2} & w^1_{1,3} & w^1_{1,4} \\ w^1_{2,1} & w^1_{2,2} & w^1_{2,3} & w^1_{2,4} \\ w^1_{3,1} & w^1_{3,2} & w^1_{3,3} & w^1_{3,4} \\ w^1_{4,1} & w^1_{4,2} & w^1_{4,3} & w^1_{4,4} \end{bmatrix}_{4\times4} \text{ y } W^2 = \begin{bmatrix} w^2_{1,1} \\ w^2_{2,1} \\ w^2_{3,1} \\ w^2_{4,1} \end{bmatrix}_{4\times1}.$$

El proceso de entrenamiento de un perceptrón multicapa se basa en el algoritmo de propagación hacia delante (*forward propagation*) y la retropropagación (*backpropagation*), que ajusta los pesos de las conexiones para minimizar una función de pérdida o error.

Forward and backward propagation

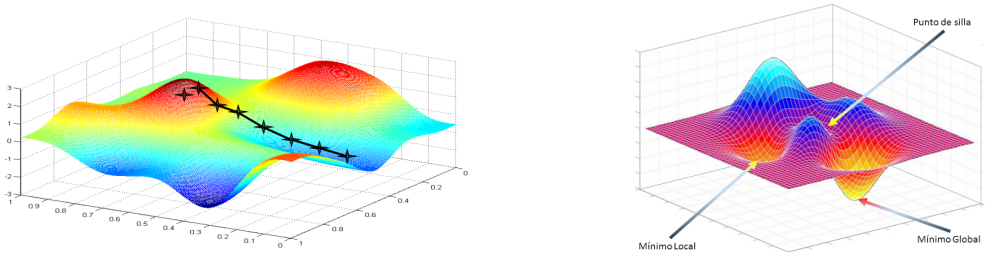
La señal se desplaza a lo largo de la red neuronal mediante un proceso llamado *forward propagation*. Al empezar el entrenamiento, los pesos de las conexiones entre las neuronas son inicializados aleatoriamente, de manera que la capa de entrada es alimentada por primera vez con los datos de la imagen original. Cada una de las capas del modelo está compuesta por estas neuronas, encargadas de procesar la información mediante operaciones matemáticas.

La salida generada por las neuronas servirá como entrada para las siguientes y así sucesivamente hasta llegar a la capa de salida, encargada de generar una predicción mediante una función de activación más específica. Cuando se obtiene la predicción, la función de pérdidas es la responsable de compararla con la *ground truth* y medir el nivel de discrepancia entre ambas, siendo por tanto el objetivo reducir al máximo posible este valor. Una vez obtenida la pérdida, el error calculado se propaga hacia atrás (*backpropagation*) a través de las mismas conexiones que generaron la predicción, de forma que se calcula la contribución de cada peso en el error total de la salida y se recalculan los coeficientes con el fin de disminuir dicha pérdida.

Esta retropropagación utiliza el gradiente descendente para actualizar los pesos de manera iterativa, junto con una tasa de aprendizaje, η, que controla el tamaño de los ajustes que se aplican a cada peso, buscando el mínimo global de la función de pérdidas. Sin embargo, si los parámetros no se ajustan adecuadamente, el proceso puede quedarse estancado en un mínimo local o punto de silla (figura 3.12b).

La expresión matemática que lo representa queda de la siguiente manera:

$$w_0(t+1) = w_0(t) - \eta \frac{\partial J}{\partial w_0} \qquad w_1(t+1) = w_1(t) - \eta \frac{\partial J}{\partial w_1} \tag{3.1}$$



(a) Direcciones de búsqueda del gradiente (b) Función de pérdidas

Figura 3.12: Gradiente descendente. Fuente: [32]

Por lo general, valores altos de η conllevan un aprendizaje rápido, causando un riesgo de no converger. Así mismo, emplear valores muy bajos puede llevar a un aprendizaje excesivamente lento.

A cada una de las iteraciones de este proceso se le llama época. Esto permite al modelo ajustar gradualmente sus pesos para mejorar su rendimiento en los datos de entrenamiento. Tal y como se ha visto en la figura 3.8, el caso óptimo de entrenamiento es aquel donde las curvas de pérdidas convergen en 0 a lo largo de las épocas mientras que las curvas de precisión aumentan.

3.3.2. Estructura de una CNN

Una *Convolutional Neural Network* (CNN) es un tipo de modelo de aprendizaje profundo especialmente diseñado para el procesamiento de imágenes, siendo altamente eficiente en el reconocimiento de patrones visuales y utilizado en tareas de clasificación, detección y segmentación de imágenes.

La figura 3.13 muestra la estructura de una CNN que tiene como entrada una imagen compuesta por los canales RGB y como salida un vector de tres elementos, donde cada uno representa la probabilidad de que una entrada determinada pertenezca a cada una de las clases definidas previamente (en el ejemplo tres). Esta estructura suele consistir en capas convolucionales, encargadas de extraer las características y patrones relevantes, seguidas de capas *pooling* que reducen el tamaño espacial de las características extraídas por las capas convolucionales, preservando las más relevantes y disminuyendo la cantidad de parámetros, y capas *fully connected* (top model) que, tras aplicarles una función de activación, producen las probabilidades de salida.

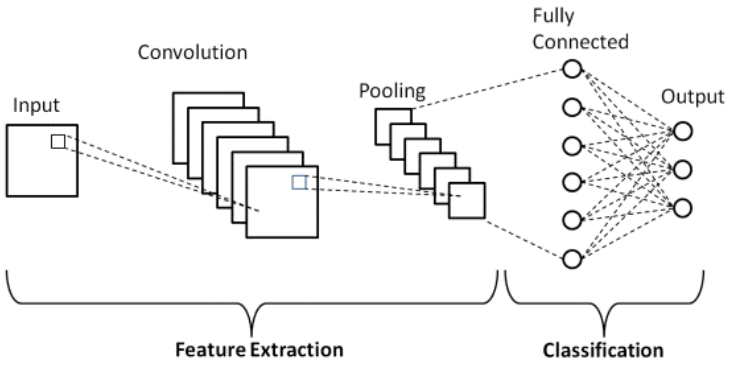


Figura 3.13: Estructura de una CNN básica con una salida de 3 clases. Fuente: [33]

Bloques convolucionales

Un bloque convolucional se compone por dos de los tres elementos comentados anteriormente, capas convolucionales y *pooling*. En función de la complejidad de la tarea que se esté llevando a cabo, el número de capas intermedias que se encuentran entre la capa de entrada y la capa de salida, también conocidas como capas ocultas, puede aumentarse o disminuirse. Su función es procesar la información de entrada mediante la aplicación de transformaciones no lineales, permitiendo a la red aprender representaciones más complejas y abstractas de los datos.

Las primeras capas aplican filtros convolucionales a la entrada de la red para extraer características importantes de la imagen original. La capa convolucional utiliza un pequeño filtro (*kernel*) que se desliza por la imagen en una operación de convolución. En cada posición de este filtro, se realiza una multiplicación elemento por elemento entre los valores de la imagen de entrada y los valores del kernel. Luego, se suman todos los valores multiplicados y se almacena en una nueva imagen (o mapa de activación). El proceso se repite en diferentes regiones de la imagen, produciendo cada vez un valor diferente en el mapa de características. Estos valores representan características específicas de la imagen, como bordes, texturas, patrones...

El tamaño del filtro y la cantidad de filtros utilizados en la capa convolucional se pueden ajustar para controlar la cantidad y la complejidad de las características extraídas de la imagen de entrada.

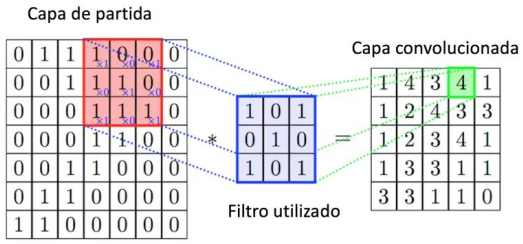


Figura 3.14: Proceso de convolución a los datos de entrada. Fuente: Elaboración propia

Las capas de *pooling* reducen el tamaño de los mapas de activación generados por las capas convolucionales, extrayendo información importante mientras se reduce la cantidad de parámetros y la complejidad computacional. La operación más común utilizada en las capas de *pooling* es el *max pooling*, que toma la salida de cada filtro convolucional y la divide en regiones no solapadas, definidas por un *stride* que determina la cantidad de píxeles que se desplaza el filtro. Luego, en cada región, se selecciona el valor máximo y se descarta el resto. De esta manera, se reduce el tamaño del mapa de características y se mantienen las características más importantes.

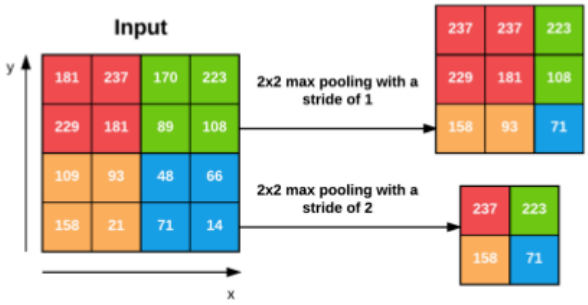


Figura 3.15: Proceso de *max pooling*. Fuente: [34]

Las capas totalmente conectadas (*fully connected*) son la última parte de la CNN, donde se combinan funciones afines ($y = Wx + b$) y no lineales (como la ReLu, Softmax, Sigmoide, etc).

Estas capas reciben datos de la capa *flatten*, que es una capa unidimensional consecuencia de convertir el último volumen (que ya será de un tamaño reducido) en un vector, y se someten, en primer lugar, a la operación afín y luego a una función no lineal para aprender y modelar relaciones complejas en los datos.

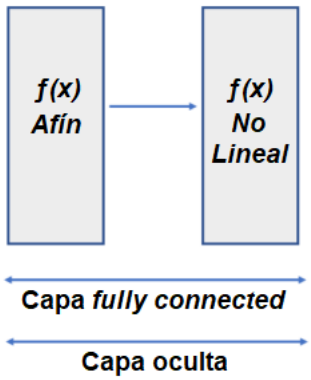


Figura 3.16: Capa *fully connected*. Fuente: Elaboración propia

La salida se envía a la función de activación para obtener la distribución de probabilidad sobre el conjunto final del número total de clases.

Capas de activación (*Activation function layers)*

Uno de los principiales objetivos en el desarrollo de una red neuronal es conseguir que sea capaz de aprender y representar relaciones no lineales entre las distintas características de la imagen, de forma que su capacidad para abordar problemas más complejos y diversos aumente. Las capas de activación son las encargadas de llevar esto a cabo.

Cada una de las neuronas del vector unidimensional generado se ve sometida a una función de activación que se encargará de determinar si esa neurona en cuestión ofrece una información lo realmente valiosa como para transmitirla a la siguiente capa (ya sea otra oculta o la de salida).

Aunque depende del objetivo, la función más común en las *hidden layers* es la Rectified Linear Unit (ReLU). Esta es una función lineal definida a trozos que mostrará la entrada directamente si es positiva; de lo contrario, mostrará cero [35], siendo su expresión matemática de la siguiente forma:

$$f(x) = \text{máx}(0, x) \tag{3.2}$$

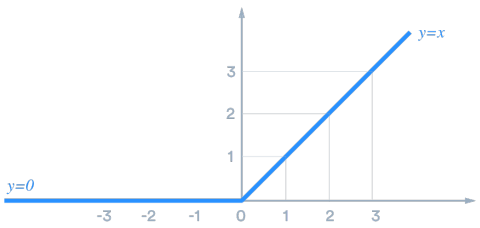


Figura 3.17: Función Rectified Linear Unit. Fuente: [36]

Se ha convertido en la función de activación por defecto para muchos tipos de redes neuronales por ser computacionalmente más ligera y, normalmente, obtener mejores resultados. Sin embargo, no es lo suficientemente precisa como para actuar en la capa de salida.

Para determinar si la salida de un modelo pertenece a una clase u otra, las funciones *Sigmoide* y *Softmax* son las más usadas debido a sus propiedades y capacidad de adpatación en diferentes tipos de problemas.

La función Softmax toma un vector de valores $x = (x_1, x_2, \ldots, x_n)$ y produce un vector de probabilidades $p = (p_1, p_2, \ldots, p_n)$, donde n es el número de clases posibles. Su ecuación se define como:

$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{3.3}$$

Para cada elemento x_i en el vector de entrada x, se aplica la función exponencial y se normaliza dividiendo por la suma exponencial de todos los elementos.

Por otro lado, la función Sigmoide toma un valor real x y produce una probabilidad p en el rango de 0 a 1. Su ecuación se define como:

$$\text{Sigmoide}(x) = \frac{1}{1 + e^{-x}} \tag{3.4}$$

Donde e es la base del logaritmo natural.

3.3.3. U-Net

Para este Trabajo Fin de Grado, el modelo propuesto como resolución a la tarea de segmentación de línea de costa ha sido la *U-Net*, una popular arquitectura de aprendizaje profundo diseñada para la segmentación semántica. Aunque fue desarrollada originalmente para imágenes médicas, su desempeño en distintos campos la ha convertido en una de las mejores redes neuronales para segmentación. Si bien su estructura (figura 3.18) no se asemeja del todo a la de una CNN como la vista en el apartado 3.3.1, su funcionalidad es muy parecida.

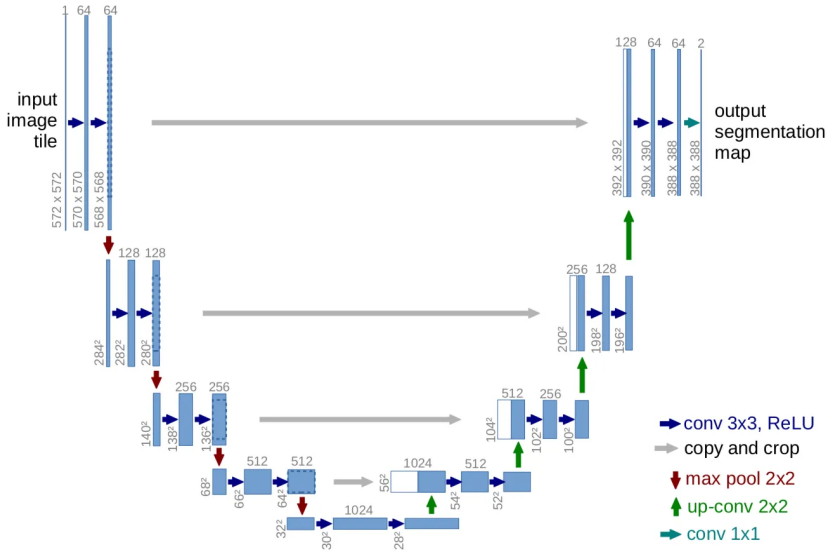


Figura 3.18: Estructura de una *U-Net*. Fuente: [37]

Esta red se compone de dos bloques principales; el *encoder*, encargado de contraer la imagen, y el *decoder*, encargado de expandirla. Además, posee una característica que la hace diferencial contra el resto de redes, las *skip connections*.

Codificador (Encoder)

El objetivo principal de la segmentación semántica es discernir los elementos que hay en una imagen y dónde se encuentran, tal y como se pretende con la detección de objetos o la clasificación de imágenes.

Como se ha explicado en secciones anteriores, hasta la CNN más simple contiene un codificador, y es que este es el encargado de crear una representación compacta de la imagen de entrada de menor dimensión que contenga solo la información más importante de la imagen. En otras palabras, el codificador se utiliza para extraer características.

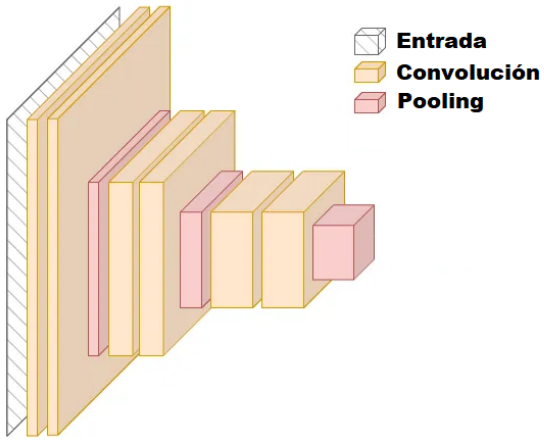


Figura 3.19: Representación gráfica de un codificador. Fuente: Elaboración propia

Al igual que en una CNN, esta parte de la red está compuesta por capas convolucionales y capas de *pooling*. A partir de la combinación de varias de estas capas se extrae información más precisa, pasando de detalles elementales, como bordes y colores, a características de alto nivel, como contexto espacial, patrones espectrales... La red aprenderá qué características son importantes para la clasificación y las extraerá para crear una representación compacta de la imagen. El problema es que esta representación no incluye la ubicación de las características en la imagen.

Este es un factor crucial ya que, por ejemplo, al conocer la ubicación de las características, la red es capaz de adquirir información sobre las relaciones espaciales entre diferentes objetos y estructuras en la imagen. Por ejemplo, las características extraídas de una orilla de un río pueden estar relacionadas con las características de la vegetación circundante, lo que permite identificar la presencia de agua y distinguirla de otras áreas. Además, al identificar con precisión los bordes, contornos o regiones de interés, el modelo puede generar máscaras más precisas, lo que resulta en una segmentación más precisa.

Decodificador (Decoder)

Otro problema del codificador es que su salida tiene una dimensión reducida. Si se utiliza para la clasificación, la etapa final o **top model** sería un perceptrón multicapa con una neurona por cada clase. Sin embargo, para la segmentación, la salida ha de ser una imagen con la misma altura y anchura que la entrada.

Para esto, es necesario un decodificador. Como se ve en la figura 3.20, desde *conv1* hasta *conv4* corresponde a la estructura del codificador vista anteriormente. A partir de *conv4* empieza el decodificador, encargado de reconstruir la imagen a partir de la representación compacta con la mayor precisión posible. Al igual que el codificador, se compone de bloques convolucionales pero con la novedad de que ahora se usan capas de deconvolución que aumentan la dimensionalidad de la imagen.

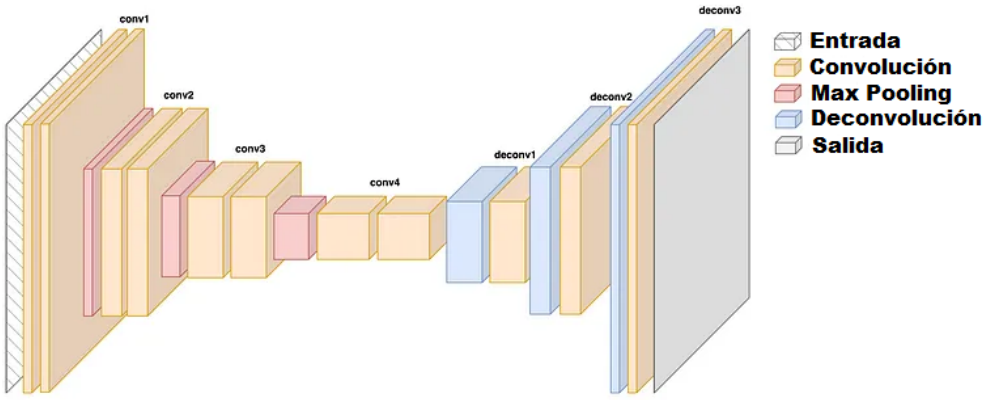


Figura 3.20: Representación gráfica de un *encoder-decoder*. Fuente: Elaboración propia

Como ya se ha mencionado, las capas de *pooling* utilizarán un método predefinido para reducir la dimensionalidad, como por ejemplo mediante un *max pooling*. En contraposición, las capas de deconvolución aumentan la dimensionalidad utilizando una función de remuestreo que se actualiza a medida que se entrena el modelo.

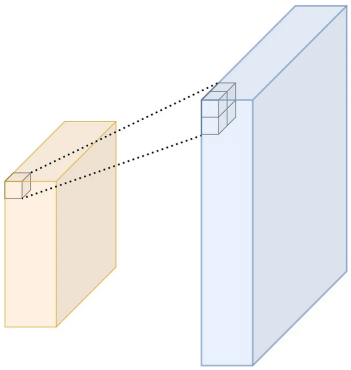


Figura 3.21: Representación gráfica de una capa de deconvolución. Fuente: Elaboración propia

Cuando la arquitectura *encoder-decoder* se utiliza para reconstruir la imagen original a partir de ruido, siendo la *ground truth* la imagen original, se llama *autoencoder*.

El decodificador es capaz de pasar las características importantes al codificador. Sin embargo, la pérdida de la localización de las características es prácticamente inevitable si la base de datos no contiene una cantidad de datos exageradamente grande, de modo que el decodificador no es capaz de aprender a reconstruir con precisión las imágenes. Con las *skip connections*, se puede solventar este problema.

Skip connections

En una U-Net, a diferencia de un *encoder-decoder* tradicional (figura 3.20), se encuentran las conocidas como *skip connections* que se utilizan para pasar información de las capas convolucionales a las capas de deconvolución directamente. En esencia, lo que se transmite es la ubicación de la característica extraída por las capas convolucionales, es decir, indican a la red de qué parte de la imagen proceden las características.

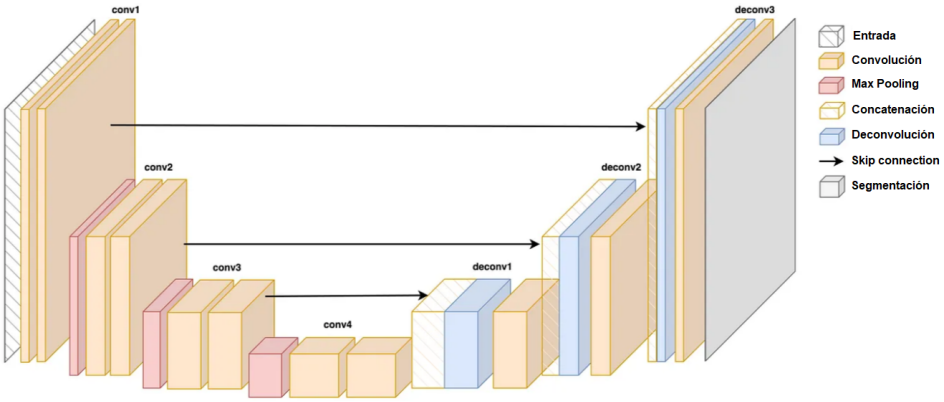


Figura 3.22: Muestra de *Skip connections* en una U-Net. Fuente: Elaboración propia

Este proceso se lleva a cabo concatenando la última capa del bloque convolucional y la primera del bloque deconvolucional opuesto. Como se puede ver tanto en la figura 3.18 y 3.22, la estructura de la U-Net es simétrica, por lo que las dimensiones de las capas opuestas serán las mismas, facilitando la combinación de las capas en un único tensor. La convolución se realiza entonces como en cualquier otra CNN, aplicando el *kernel* sobre el tensor concatenado.

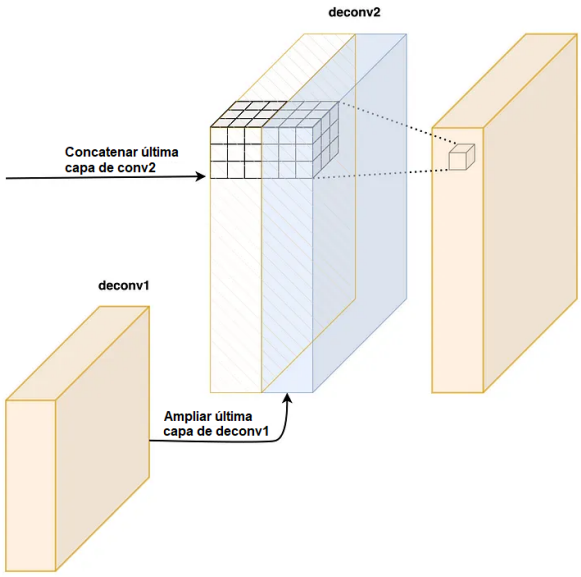


Figura 3.23: Ilustración del proceso de concatenación. Fuente: Elaboración propia

Con esto, se consigue que las características pasen de la capa anterior a la capa ampliada (bloque azul) y que la ubicación se transmita desde la capa de convolución opuesta (bloque anaranjado).

Al combinar esta información, se logra mejorar el rendimiento en la segmentación y se contribuye a reducir la cantidad de datos requeridos para entrenar la red. No obstante, como se detallará en la sección 3.3.4, hay varios hiperparámetros que influyen en la precisión de los resultados y en el rendimiento general del modelo.

3.3.4. Adaptación del modelo a la detección de línea de costa

EfficientNet-B7* e *ImageNet

Para abordar la tarea de segmentación de línea de costa, se ha mejorado y optimizado la arquitectura original de la U-Net mediante el uso del codificador *EfficientNet-B7* [38] y el uso de pesos preentrenados de *ImageNet* [39].

Un *encoder* convencional, como el visto en la sección anterior, está compuesto por capas convolucionales de tamaño fijo. Aunque estas capas pueden capturar características importantes de las imágenes, su capacidad de representación es limitada en comparación con un *encoder* basado en la arquitectura *EfficientNet-B7*, mostrada en la figura 3.24.

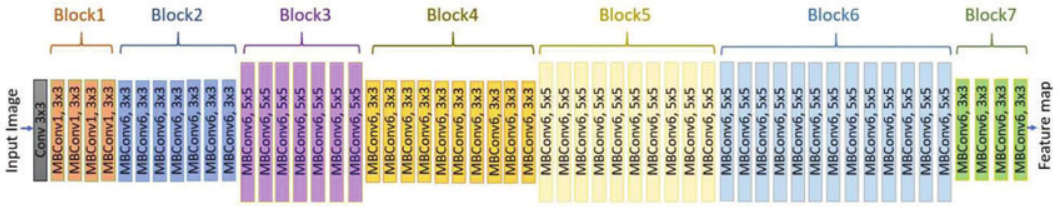


Figura 3.24: Arquitectura del modelo *EfficientNet-B7*. Fuente: [40]

Propuesta en el año 2019 por los investigadores Mingxing Tan y Quoc Le en su trabajo *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks* [38], la arquitectura *EfficientNet* se basa en tres componentes principales: escalado de ancho (width scaling), escalado de profundidad (depth scaling) y escalado de resolución (resolution scaling). El modelo *EfficientNet-B7* se obtiene al aplicar un *width scaling* de 4, lo que significa que el número de canales en cada capa se incrementa cuatro veces en comparación con el modelo base. En cuanto al *depth scaling*, utiliza un factor de aumento de profundidad de 8, lo que implica que tiene una mayor cantidad de capas en comparación con el modelo base, respetando la condición de simetría con el *decoder*. Por último, el *resolution scaling* implica aumentar la resolución de entrada del modelo.

Cada bloque *MBConv* contiene una secuencia de capas que incluyen convoluciones, normalización por lotes (para normalizar las salidas de las capas intermedias), funciones de activación y operaciones de reducción de dimensionalidad (max pooling).

Al utilizar un enfoque de escalado compuesto, que optimiza el rendimiento y la eficiencia del modelo en función de sus dimensiones, se consigue una arquitectura más profunda y compleja capaz de capturar características más sutiles y significativas. Además, ha sido previamente entrenada en un conjunto masivo de imágenes de carácter general de *ImageNet* (con más de 14 millones de imágenes), como animales, edificios, personas... por lo que se aprovechan los conocimientos aprendidos por el modelo en tareas previas de clasificación de imágenes.

Fine-tuning

Sin embargo, con el objetivo de preservar las características generales aprendidas con *ImageNet* y adaptar el modelo a nuestras imágenes, ha sido necesario aplicar una técnica conocida como *fine-tuning*.

Durante este proceso, el *encoder* se mantiene congelado, por lo que sus pesos no son afectados en el momento del entrenamiento, siendo los mismos que se obtuvieron en el entrenamiento con *ImageNet*. Mientras, el *decoder*, encargado de la segmentación, se descongela [41] y se ajusta en base a los datos específicos de las imágenes de SWED. Esto permite una adaptación especializada y una mayor capacidad de representación para capturar los detalles y las peculiaridades relevantes para la tarea de segmentación en nuestras imágenes.

Función de activación *Softmax* en la capa de salida

Como se ha visto y explicado en la sección 3.3.2, la función de activación *ReLU* ha sido utilizada para activar cada una de las neuronas en todo el modelo. Sin embargo, para la capa de salida, se ha empleado la *Softmax* (expresión 3.3) que toma un vector de valores reales y los normaliza, asignando una probabilidad a cada elemento. Esta normalización garantiza que todas las probabilidades sean no negativas y sumen 1, permitiendo interpretar las salidas del modelo como probabilidades de pertenencia a cada clase. En la *U-Net* entrenada existe una *softmax* para cada píxel, obteniendo la máscara a partir de las clases que tienen la probabilidad máxima.

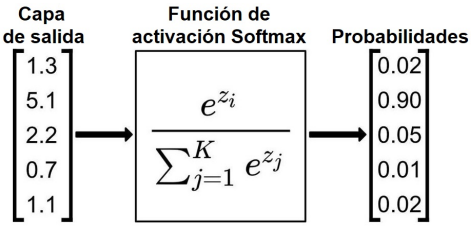


Figura 3.25: Comportamiento de la función *Softmax*. Fuente: Elaboración propia

Para poder hacer funcional esta modificación, se convirtieron, previamente al entrenamiento del modelo, las máscaras (verdad terreno) a formato *one-hot encoding*.

El formato de codificación *one-hot* es una técnica utilizada para representar clases de manera binaria. En este contexto, implica convertir cada valor de una clase en un vector binario cuyo tamaño es igual al número total de clases posibles, de modo que se asigna un vector binario único para cada clase. Todos los elementos del vector se establecen en 0, excepto el elemento correspondiente a la clase en cuestión, que se establece en 1. Por lo tanto, si hay C clases posibles, el vector *one-hot* tendrá una longitud de C, y solo un elemento será 1, mientras que los demás serán 0.

Función de pérdida *Cross-Entropy*

La función de pérdida elegida para este proyecto ha sido la *Cross-Entropy*, cuya expresión matemática es la siguiente.

$$L_n = -\sum_{i=1}^{n} y_i \log(\hat{y}_i) \tag{3.5}$$

$$L = \frac{1}{N} \sum_{n=1}^{N} L_n \tag{3.6}$$

Donde n es el número de clases, N el *batch size*, y_i la verdad terreno e $\hat{y}_i$ la predicción.

Esta función de pérdida calcula la diferencia entre dos distribuciones de probabilidad para un conjunto dado de ocurrencias o variables aleatorias, indicando la diferencia promedio entre los valores predichos y los valores reales. Para mejorar la precisión del modelo, se debe tratar de minimizar su resultado. La puntuación de la entropía cruzada se encuentra entre 0 y 1, siendo 0 el valor perfecto.

Mientras que la pérdida cuadrática y otras funciones castigan las predicciones incorrectas, la entropía cruzada penaliza duramente la sobreconfianza. A diferencia de la pérdida de probabilidad logarítmica negativa, que no considera la confianza en la predicción, la entropía cruzada penaliza tanto las predicciones incorrectas pero confiadas como las predicciones correctas pero menos confiadas.

Algoritmo de optimización *Adam*

La *Adaptive Moment Estimation* (ADAM) [42] es un algoritmo utilizado en la técnica de optimización por descenso de gradiente y el utilizado para este proyecto. Se destaca por su eficiencia al trabajar con problemas grandes que involucran una gran cantidad de datos o parámetros. Una de las más importantes ventajas de ADAM es que requiere menos memoria y es altamente eficiente.

Este método se fundamenta en una combinación de dos metodologías de descenso de gradiente:

- *Momentum*

  Utilizado para acelerar el algoritmo de descenso de gradiente teniendo en cuenta la "media ponderada exponencialmente"de los gradientes. El uso de promedios hace que el algoritmo converja hacia los mínimos a un ritmo más rápido.

$$w_{t+1} = w_t - \alpha m_t \tag{3.7}$$

  donde:

$$m_t = \beta m_{t-1} + (1-\beta)\left[\frac{\delta L}{\delta w_t}\right] \tag{3.8}$$

  siendo m_t el agregado de gradientes en el momento t (inicialmente, $m_t = 0$), m_{t-1} el conjunto de gradientes en el momento $t-1$ (anterior), W_t las ponderaciones en el momento t, W_{t+1} las ponderaciones en el momento $t+1$, α_t la tasa de aprendizaje en el tiempo t, $\frac{\partial L}{\partial W_t}$ la derivada de los pesos en el tiempo t y β el parámetro de media móvil (constante, 0,9).

- Algoritmo *RMSP*

  Root mean square prop (RMSprop) es un algoritmo que se caracteriza por hacer que la tasa efectiva sea inversamente proporcional al gradiente, es decir, cuando la η efectiva se reduce para los coeficientes que reciben un alto gradiente, crece para los que reciben pequeños gradientes o no se actualizan frecuentemente.

$$w_{t+1} = w_t - \frac{\alpha_t}{(v_t + \varepsilon)^{1/2}} * \left[\frac{\delta L}{\delta w_t}\right] \tag{3.9}$$

  donde:

$$v_t = \beta v_{t-1} + (1 - \beta) * \left[\frac{\delta L}{\delta w_t}\right]^2 \tag{3.10}$$

  siendo W_t las ponderaciones en el momento t, W_{t+1} las ponderaciones en el momento $t+1$, α_t la tasa de aprendizaje en el tiempo t, $\frac{\partial L}{\partial W_t}$ la derivada de los pesos en el tiempo t, V_t la suma del cuadrado de los gradientes pasados. (Es decir, $\sum\left(\frac{\partial L}{\partial W_{t-1}}\right)$) (Inicialmente, $V_t = 0$), β el parámetro de media móvil (constante, 0,9) y ϵ una pequeña constante positiva (10^{-8}).

Quedando por tanto la expresión matemática del optimizador de la siguiente manera.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\left[\frac{\delta L}{\delta w_t}\right] v_t = \beta_2 v_{t-1} + (1 - \beta_2)\left[\frac{\delta L}{\delta w_t}\right]^2 \tag{3.11}$$

De esta manera, el optimizador Adam hereda los puntos fuertes de los dos métodos anteriores y se basa en ellos para ofrecer un descenso de gradiente más optimizado.

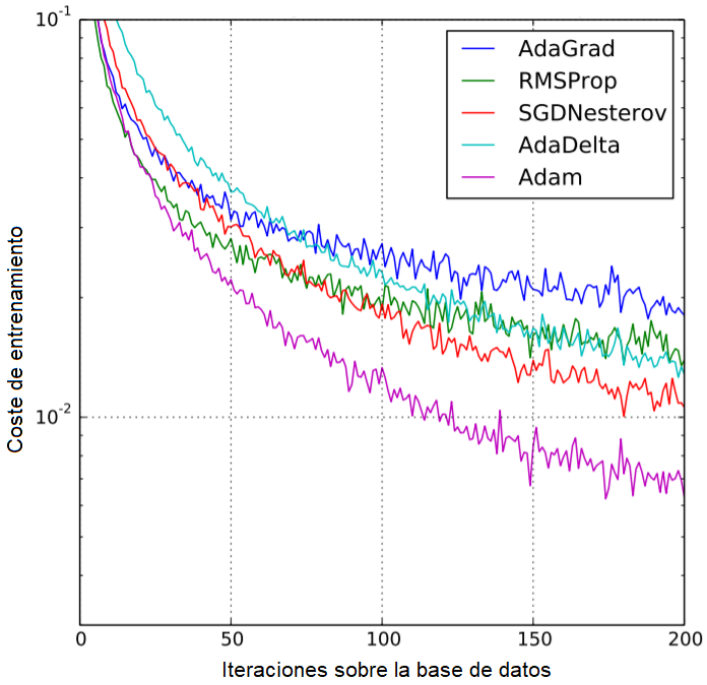


Figura 3.26: Rendimiento del optimizador Adam y otros usados comúnmente sobre la tradicional base de datos de dígitos escritos a mano MNIST [43]. Fuente: Elaboración propia

Capítulo 4

Resultados

4.1. Evaluación con datos de *test* externos

4.1.1. Conjunto de datos SWED

El conjunto de datos SWED se fraccionó en dos particiones completamente independientes; la que contiene la mayor parte del conjunto de las imágenes se dedicó al entrenamiento del modelo y las restantes a su evaluación. Tal y como se puede ver en la figura 4.1, el conjunto de entrenamiento se subdivide en dos pequeñas particiones, la de *train* y la de *validation*, con un 80 % y un 20 % de las imágenes del conjunto respectivamente.

Esta división en subconjuntos de *train* y *validation* se utiliza para entrenar el modelo en un conjunto de datos (conjunto de *training*) y evaluar su rendimiento en datos no vistos durante el entrenamiento (conjunto de *validation*). La partición de *train* se utiliza para ajustar los parámetros del modelo, mientras que la partición de validation se emplea para evaluar y ajustar el rendimiento del modelo a lo largo de las épocas, ayudando a seleccionar los mejores hiperparámetros y obtener estimaciones confiables del rendimiento en datos nuevos.

Esta agrupación a su vez se puede dividir en pequeños lotes, o *batches*, donde se agrupan las imágenes según un valor que se especifique. Por ejemplo, si se asigna un *batch size* de 16, cada iteración en una época tendrá como entrada 16 imágenes, de modo que la época finaliza cuando todos los *batches* realizan el *forward propagation* para la actualización de los coeficientes. Esto permite ganar eficiencia computacional y estabilidad en el modelo, ya que al calcular los gradientes en función de un conjunto más grande de ejemplos, se puede obtener una estimación más precisa del gradiente general, lo que puede ayudar a evitar fluctuaciones y mejorar la convergencia.

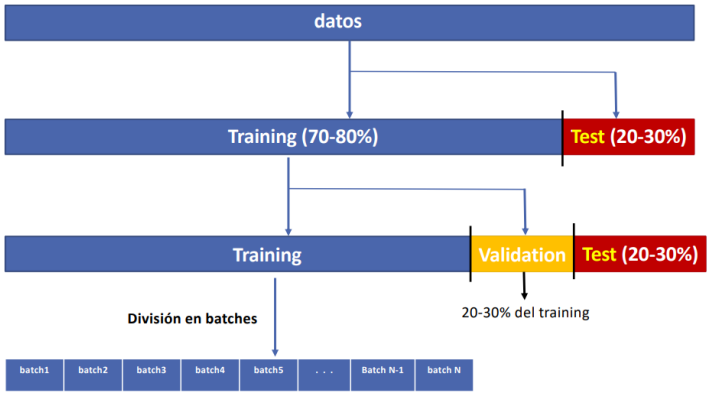


Figura 4.1: Particiones a aplicar en un conjunto de datos. Fuente: [31]

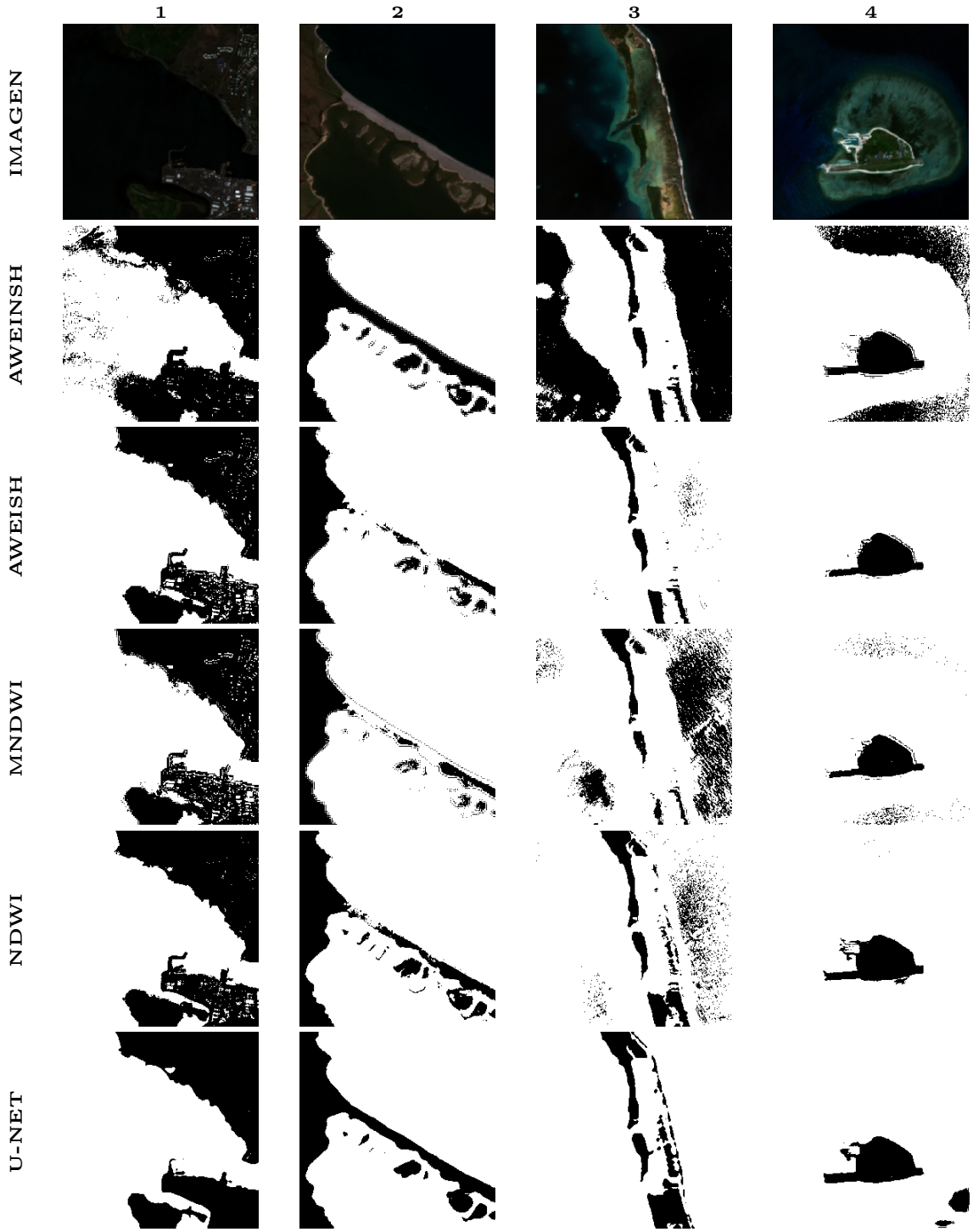


Figura 4.2: Resultados obtenidos con los índices de agua AWEINSH, AWEISH, MNDWI, NDWI y el modelo de red neuronal entrenado U-NET sobre las imágenes del conjunto de *test* del dataset SWED. La imagen 1 corresponde a una zona costera de Islandia, la 2 a una de Rusia, la 3 a una de Islas Cook y la 4 a una de Islas Spratly (mar de la China Meridional). Los umbrales empleados en cada índice ha sido el estándar, 0. Fuente: Elaboración propia

4.1.2. Conjunto de datos CGAT

Este conjunto de datos, a diferencia del visto anteriormente, fue usado en su totalidad para la evaluación del modelo y clasifica los píxeles en función de si pertenece a agua salada (mar) o no, al margen de si la zona con valores de píxel 0 corresponden a humedales, lagunas, lagos, ríos...

El modelo entrenado, al igual que los índices clásicos, detecta cualquier tipo de agua, por lo que para poder realizar una correcta evaluación se ha extraído de las máscaras verdad terreno una región de interés (ROI) que contuviera la línea de costa mediante técnicas de dilatación y erosión morfológicas.

(a) Máscara original (b) Máscara con ROI (c) Máscaras superpuestas

Figura 4.3: ROI generada sobre máscara verdad terreno. Fuente: Elaboración propia

Dilatación morfológica

En la dilatación morfológica, se utiliza un elemento estructurante, denotado como B, que corresponde a una forma geométrica predefinida. En este caso, esa forma se ha definido como un cuadrado.

Antes de aplicar la dilatación, B se refleja sobre sus 180 grados para obtener su versión invertida con el fin de expandir los bordes de interés de la imagen. Luego, se desplaza por toda la imagen de entrada comparando el elemento estructurante con los píxeles correspondientes de la imagen.

$$A \oplus B = \left\{ z \mid (\widehat{B})_z \cap A \neq \phi \right\} \tag{4.1}$$

La operación de dilatación, $A \oplus B$, resulta en un conjunto de todos los desplazamientos posibles del elemento B, indicado como z, donde al menos uno de los píxeles del elemento se encuentra contenido en el conjunto A.

Es decir, la dilatación morfológica expande las zonas blancas, que en este caso corresponde a las zonas de mar de la máscara.

Erosión morfológica

En la erosión morfológica, al igual que en la dilatación, se utiliza un elemento estructurante B de forma cuadrada. Sin embargo, en la erosión, no se realiza ningún reflejo o inversión del elemento.

B se desplaza por toda la imagen de entrada, al igual que en la dilatación. En cada posición, se compara el elemento estructurante con los píxeles correspondientes de la imagen. La operación de erosión, denotada como $A \ominus B$, resulta en el conjunto de todos los desplazamientos posibles de B, indicado como z, donde todos los píxeles del elemento estructurante se encuentran contenidos en el conjunto A.

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \tag{4.2}$$

Es decir, la erosión expande las zonas negras, que en este caso corresponde con las zonas de terreno de la máscara.

Tanto para la dilatación como para la erosión se ha utilizado un elemento estructurante de tamaño 10x10, es decir, de 5 píxeles a cada lado de la banda, de modo que el ancho total de la banda sea de 100 metros (5 píxeles + línea de costa + 5 píxeles).

El resultado de ambas operaciones morfológicas se restan para obtener la máscara con la ROI definida, tal y como se puede apreciar en la siguiente figura.

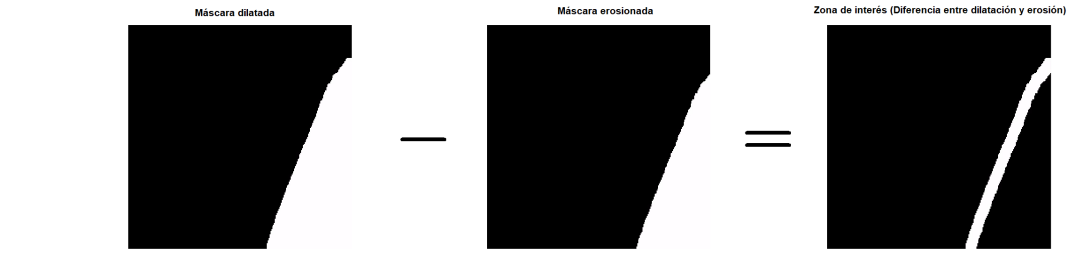


Figura 4.4: Diferencia entre la operación de dilatación y erosión morfológica para la definición de la máscara ROI. Fuente: Elaboración propia

Una vez el área de interés está delimitada, se multiplica las máscaras obtenidas mediante los índices y el modelo por la máscara ROI (figura 4.4), de modo que la evaluación solo se realizará con respecto a la zona deseada.

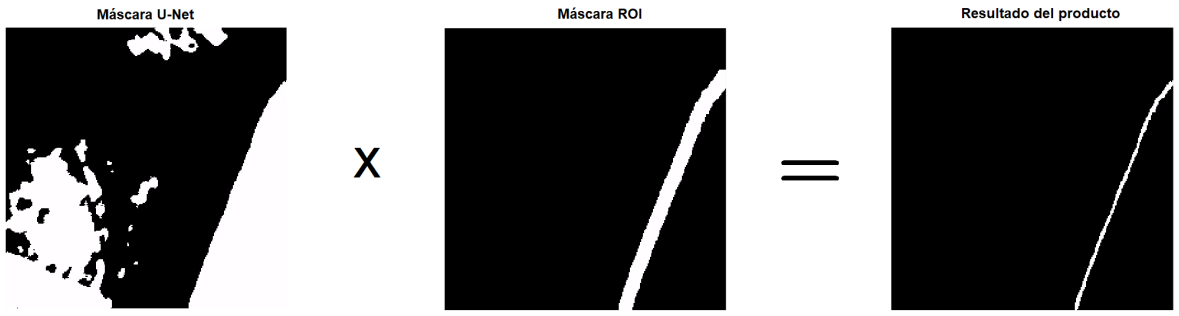


Figura 4.5: Extracción de la zona de interés sobre las máscaras generadas por el modelo para su evaluación. Fuente: Elaboración propia

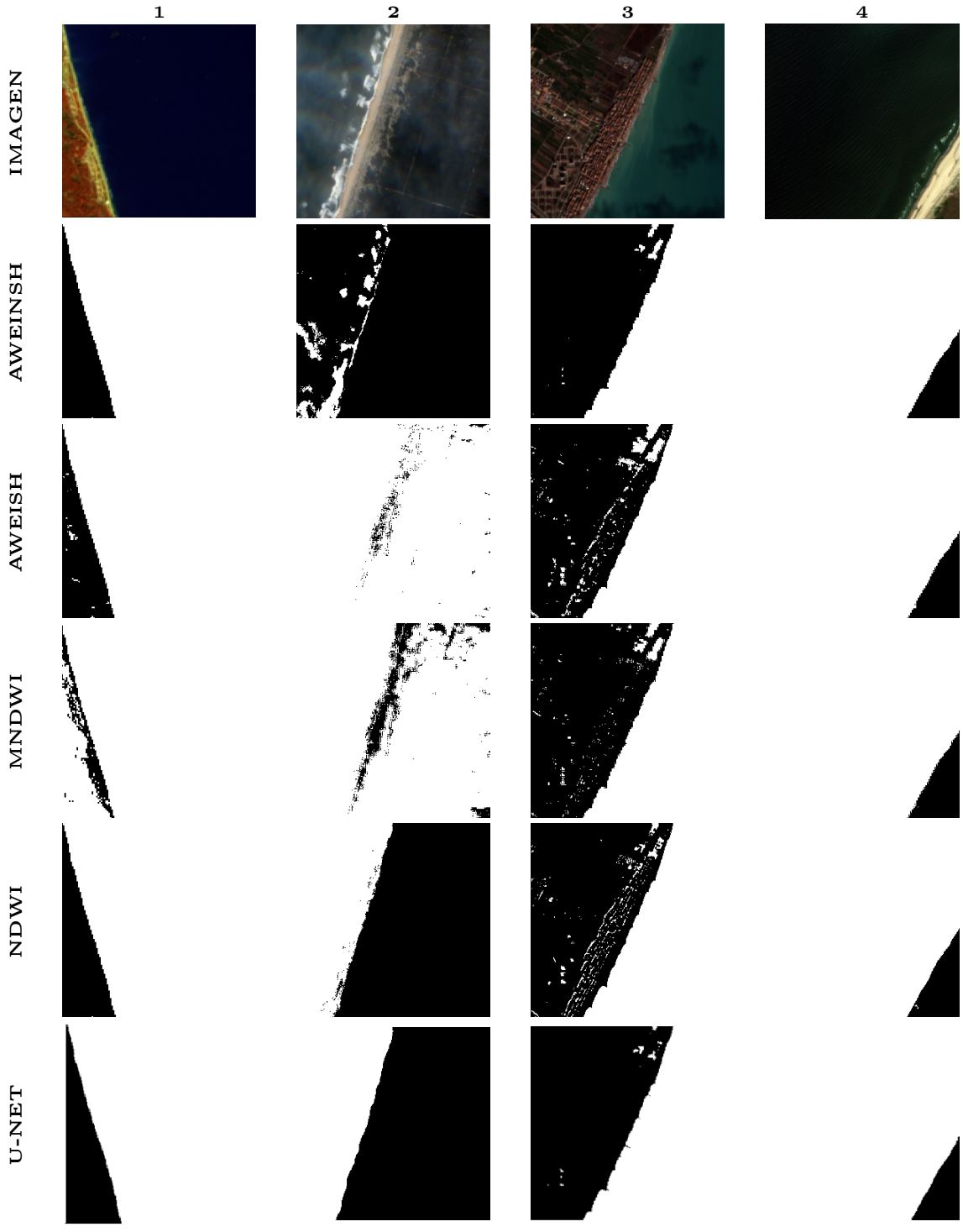


Figura 4.6: Resultados obtenidos con los índices de agua AWEINSH, AWEISH, MNDWI, NDWI y el modelo de red neuronal entrenado U-NET sobre las imágenes del conjunto de *test* del dataset CGAT. La imagen 1 corresponde a una zona costera de Valencia, la 2 a una de Portugal (con condiciones meteorológicas adversas), la 3 a una de Castellón y la 4 a una de Países Bajos. Los umbrales empleados en cada índice ha sido el estándar, 0. Fuente: Elaboración propia

4.2. Métricas de evaluación

Las métricas de evaluación cuantitativa usadas han sido:

- **Accuracy** (Exactitud). Esta métrica es comúnmente utilizada para evaluar la precisión general de un modelo. Se calcula dividiendo el número de predicciones correctas por el número total de muestras en el conjunto de datos, midiendo la proporción de predicciones correctas en comparación con todas las predicciones realizadas.

$$\frac{TP + TN}{TP + TN + FP + FN} \tag{4.3}$$

- **Precision** (Precisión). Indica la proporción de verdaderos positivos sobre el total de predicciones positivas realizadas por el modelo. En otras palabras, mide cuántas de las predicciones positivas realizadas por el modelo son realmente correctas.

$$\frac{TP}{TP + FP} \tag{4.4}$$

- **Recall** (Exhaustividad). Representa la proporción de verdaderos positivos sobre el total de ejemplos positivos en los datos de prueba. Indica qué tan bien el modelo puede recuperar o encontrar correctamente los ejemplos positivos.

$$\frac{TP}{TP + FN} \tag{4.5}$$

- **Cohen's kappa coefficient** (Coeficiente kappa de Cohen). Mide la concordancia entre las predicciones del modelo y las etiquetas reales ajustando la concordancia que podría esperarse por casualidad. Proporciona una medida más robusta al tener en cuenta el acuerdo esperado por azar. Un valor de kappa de 1 indica una concordancia perfecta, 0 indica una concordancia al azar y valores negativos indican una concordancia peor que la esperada al azar.

$$\frac{p_o - p_e}{1 - p_e} \tag{4.6}$$

- **F1-score** (Puntuación F1). Es una métrica que combina la precisión y el *recall* en un solo valor. Se calcula como la media armónica entre la precisión y el *recall*. El *F1-score* proporciona una medida equilibrada entre la precisión y la capacidad de recuperación del modelo.

$$\frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{4.7}$$

- **Dice coefficient** (Coeficiente de Dice). Utilizada principalmente en tareas de segmentación de imágenes. Calcula la similitud entre el área segmentada predicha y el área real. Un valor de 1 indica una coincidencia perfecta, mientras que un valor de 0 indica una falta total de coincidencia.

$$\frac{2 \cdot TP}{(TP + FP) + (TP + FN)} \tag{4.8}$$

- **Matthew's correlation coefficient** (Coeficiente de correlación de Matthew). Esta métrica es especialmente útil cuando las clases están desequilibradas en los datos. Toma en cuenta los verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos para calcular una medida de la calidad de la clasificación. Un valor de 1 indica una clasificación perfecta, 0 indica una clasificación al azar y -1 indica una clasificación inversa.

$$\frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{4.9}$$

Donde:
- TP: Verdaderos positivos (True Positives)
- FP: Falsos positivos (False Positives)
- FN: Falsos negativos (False Negatives)
- TN: Verdaderos negativos (True Negatives)
- p_o: Proporción observada de acuerdo (Observed Agreement)
- p_e: Proporción esperada de acuerdo (Expected Agreement)

Una muestra de los resultados obtenidos para este conjunto de *test* se pueden ver en la figura 4.2 y 4.6, así como las métricas obtenidas en la Tabla 4.2 y 4.4.

Resultados sobre SWED

Con respecto a la figura 4.2, tomando la métrica *Accuracy* como referencia, podemos ver como el modelo entrenado ha sido el que mejor resultados ha ofrecido junto con el índice NDWI.

	U-Net	**NDWI**	**MNDWI**	**AWEINSH**	**AWEISH**
Imagen 1	0.9838	0.9654	0.9390	0.8870	0.9481
Imagen 2	0.9858	0.9664	0.9113	0.9671	0.9275
Imagen 3	0.9513	0.8772	0.7108	0.4074	0.8952
Imagen 4	0.9728	0.9749	0.9504	0.8161	0.9628

Tabla 4.1: Métrica *Accuracy* obtenida sobre las imágenes de la figura 4.2

De este modo, la siguiente tabla muestra los resultados medios obtenidos sobre todo el conjunto de imágenes reservadas para la evaluación del modelo de la base de datos SWED, es decir, las métricas globales sobre las 98 imágenes reservadas para *test*. Para ello, se ha realizado la media aritmética de cada una de las métricas utilizadas.

Descripción	**Accuracy**	**Precision**	**Recall**	**Kappa**	**F1 score**	**Dice score**	**MCC**
U-NET	0.9296 ± 0.1238	0.9060 ± 0.1822	**0.9574 ± 0.1010**	0.8353 ± 0.2010	0.9139 ± 0.1652	0.9139 ± 0.1652	0.8426 ± 0.2063
NDWI	**0.9403 ± 0.1172**	**0.9469 ± 0.1030**	0.9345 ± 0.1395	**0.8478 ± 0.2060**	**0.9327 ± 0.1261**	**0.9327 ± 0.1261**	**0.8519 ± 0.2205**
AWEINSH	0.9215 ± 0.1300	0.9007 ± 0.1369	0.9544 ± 0.1221	0.8007 ± 0.2252	0.9177 ± 0.1324	0.9177 ± 0.1324	0.8079 ± 0.2426
MNDWI	0.8846 ± 0.1408	0.8563 ± 0.1921	0.9350 ± 0.1365	0.7166 ± 0.2645	0.8745 ± 0.1736	0.8745 ± 0.1736	0.7333 ± 0.2684
AWEISH	0.8686 ± 0.1937	0.9153 ± 0.1485	0.8374 ± 0.2632	0.7239 ± 0.2962	0.8385 ± 0.2414	0.8385 ± 0.2414	0.7357 ± 0.3032

Tabla 4.2: Resultados evaluación SWED

Resultados sobre CGAT

De igual manera, se ha calculado la métrica *Accuracy* para las imágenes de la figura 4.6. Sin embargo, esta evaluación, tal y como se ha explicado en el apartado 4.2.2, se ha realizado sobre una ROI centrada únicamente en la línea de costa.

	U-Net	**NDWI**	**MNDWI**	**AWEINSH**	**AWEISH**
Imagen 1	0.9978	0.9975	0.9928	0.9975	0.9962
Imagen 2	0.9979	0.9913	0.9764	0.9833	0.9745
Imagen 3	0.9956	0.9969	0.9973	0.9950	0.9979
Imagen 4	0.9991	0.9974	0.9958	0.9978	0.9955

Tabla 4.3: Métrica *Accuracy* obtenida sobre las imágenes de la figura 4.6

Quedando las métricas globales sobre el conjunto de datos del CGAT de la siguiente manera.

Descripción	**Accuracy**	**Precision**	**Recall**	**Kappa**	**F1 score**	**Dice score**	**MCC**
U-NET	**0.9965 ± 0.0029**	**0.9185 ± 0.1131**	0.9097 ± 0.1152	**0.9071 ± 0.0713**	**0.9089 ± 0.0707**	**0.9089 ± 0.0707**	**0.9097 ± 0.0648**
NDWI	0.9957 ± 0.0033	0.9036 ± 0.0863	0.9092 ± 0.1224	0.8939 ± 0.0801	0.8961 ± 0.0794	0.8961 ± 0.0794	0.8990 ± 0.0686
AWEINSH	0.9953 ± 0.0034	0.9107 ± 0.0759	0.8749 ± 0.1699	0.8741 ± 0.0757	0.8763 ± 0.1135	0.8763 ± 0.1135	0.8819 ± 0.0972
MNDWI	0.9921 ± 0.0051	0.7522 ± 0.1296	0.9811 ± 0.0355	0.8392 ± 0.0766	0.8431 ± 0.0752	0.8431 ± 0.0752	0.8513 ± 0.0657
AWEISH	0.9917 ± 0.0063	0.7460 ± 0.1358	**0.9887 ± 0.0203**	0.8382 ± 0.0879	0.8423 ± 0.0857	0.8423 ± 0.0857	0.8511 ± 0.0757

Tabla 4.4: Resultados evaluación CGAT

4.3. Experimentos de ablación

En esta sección se comentarán los diversos experimentos realizados y resultados obtenidos, siendo el objetivo evaluar el impacto de los parámetros en el rendimiento del modelo y conseguir una configuración óptima. La selección de distintas composiciones de canales en la imagen de entrada, los ajustes de hiperparámetros y entrenamiento de otros modelos son las ramas principales que componen este estudio conocido como análisis o experimentos de ablación.

4.3.1. Parámetros de entrenamiento

A pesar de que el proyecto se ha desarrollado utilizando un portátil convencional como dispositivo principal, ha sido necesario realizar una conexión de acceso remoto a una tarjeta gráfica propiedad del laboratorio CVBLab (donde he realizado mis prácticas y se ha llevado a cabo el proyecto), la NVIDIA Titan V, para poder llevar a cabo los diferentes entrenamientos del modelo con duraciones medias de entre 2 y 4 horas, las cuales varían en función del número de imágenes utilizadas. Con las características del dispositivo local, estos entrenamientos no habrían sido posibles debido a la gran cantidad de recursos que se consumen durante el proceso. De esta manera y después de probar diversas combinaciones de parámetros, estos fueron los valores que demostraron ofrecer el mejor rendimiento y resultados en el entrenamiento del modelo.

Se realizaron 40 épocas de entrenamiento, donde cada época representa un ciclo completo de procesamiento de los datos de entrenamiento. Para cada ciclo, se utilizó un tamaño de lote (*batch size*) de 16, lo que significa que se procesaron 16 muestras a la vez. La tasa de aprendizaje (η) se estableció en 0.00005. Además, se aplicó un decremento exponencial a la tasa de aprendizaje a partir de la época 31, con el objetivo de optimizar el modelo de la mejor manera posible. Por último, para aumentar la variabilidad y la robustez del modelo, se realizaron 5 transformaciones diferentes en cada imagen durante el entrenamiento para aumentar los datos (sección 3.2).

4.3.2. Selección de bandas de la imagen Sentinel

Los primeros entrenamientos se realizaron con la resolución espectral completa de las imágenes Sentinel. Sin embargo, la existencia de bandas cuya respuesta espectral no favorecía la detección de agua, como la de aerosoles, añadía un factor de ruido que empeoraba los resultados de la predicción, por lo que se tomó la decisión de seleccionar aquellas que fueran de mayor interés.

Como se ha explicado, el codificador del modelo fue previamente entrenado utilizando imágenes de *ImageNet*. Estas imágenes están compuestas por los canales RGB, por lo que en un intento inicial de mejora, se seleccionaron las bandas del rojo, verde y azul de cada imagen en el conjunto de datos. Sin embargo, los resultados obtenidos no fueron satisfactorios. De este modo, se decidió mantener los 3 canales de entrada y filtrar nuevamente las imágenes seleccionando esta vez las bandas del verde, NIR y SWIR. Estas son ampliamente utilizadas en los índices de detección de agua clásicos debido a la valiosa información que proporcionan, por lo que la red, en este caso, interpreta de una manera óptima las estructuras y patrones de textura relacionadas con el agua existente en las imágenes, obteniendo el mejor de los resultados en los experimentos de ablación.

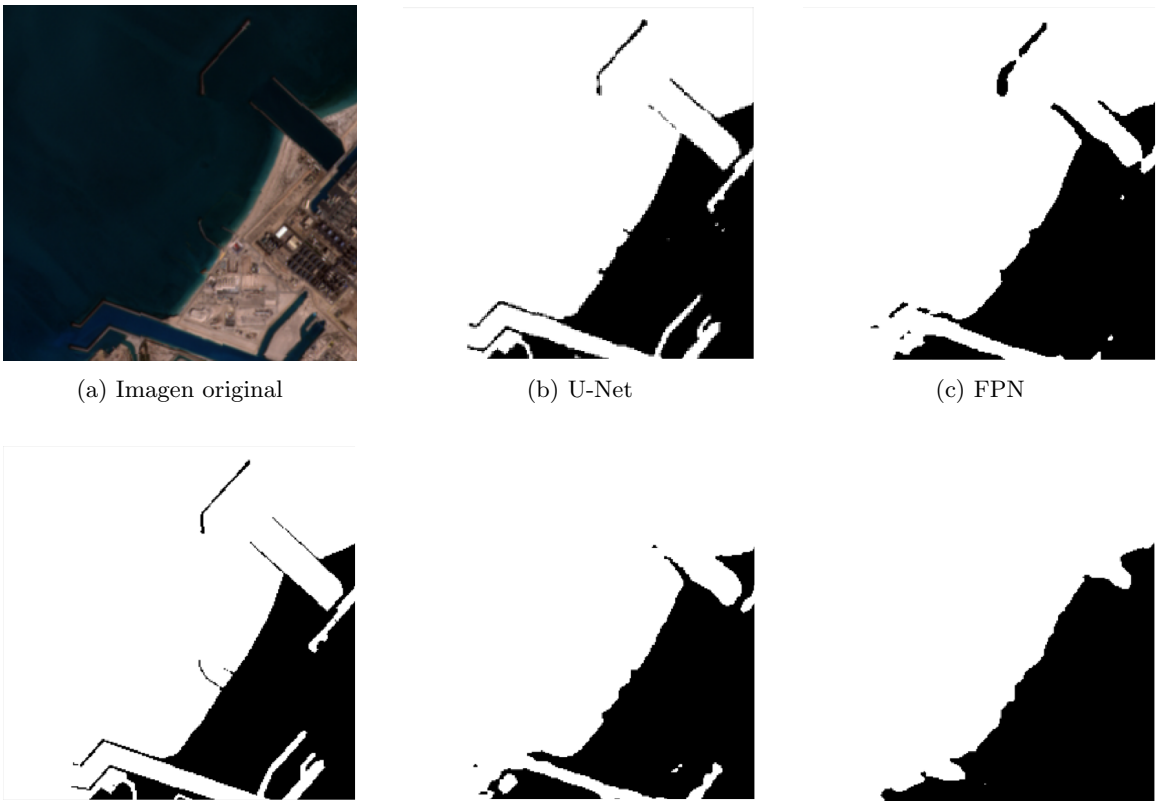
Descripción	Accuracy	Precision	Recall	Kappa	F1-score	Dice score	MCC
G-NIR-SWIR1	**0.9296**	**0.9059**	**0.9574**	**0.8331**	**0.9138**	**0.9138**	**0.8426**
12 bandas	0.8901	0.8627	0.9285	0.7483	0.8745	0.8745	0.7593
RGB	0.6206	0.6674	0.6043	0.2327	0.5851	0.5851	0.2480

Tabla 4.5: Comparación de los resultados obtenidos a partir del modelo *U-Net* entrenado con las 12 bandas espectrales, RGB y G-NIR-SWIR1 sobre el conjunto de *test* de SWED.

4.3.3. Comparación con otros modelos *encoder-decoder*

El modelo *U-Net* ha sido el que mejor resultados ha ofrecido a lo largo de los distintos experimentos realizados. Sin embargo, no ha sido el único entrenado para este proyecto.

Aunque sus estructuras son muy similares entre si, compuestos por un *encoder* y un *decoder*, existen otros modelos de segmentación de imágenes en el estado del arte que proporcionan buenos resultados como la *Feature Pyramid Network* (FPN) [44], *DeepLabV3Plus+* [45] y *Path Integral Based Convolution* (PAN) [46]. Estos han sido los tres modelos adicionales que se han entrenado a la par que la *U-Net*. En cambio, tal y como se podrá ver a continuación, los resultados que han ofrecido no han conseguido mejorar a los del modelo definitivo.



(a) Imagen original (b) U-Net (c) FPN

(d) Verdad terreno (e) DeepLabV3+ (f) PAN

Figura 4.7: Muestra de los resultados obtenidos con los distintos modelos entrenados sobre una imagen del conjunto de *test* de SWED. Fuente: Elaboración propia

Descripción	Accuracy	Precision	Recall	Kappa	F1 score	Dice score	MCC
U-NET	**0.9296**	**0.9059**	**0.9574**	**0.8331**	**0.9138**	**0.9138**	**0.8426**
FPN	0.9028	0.8902	0.9287	0.7676	0.8880	0.8880	0.7830
DeepLabV3+	0.9018	0.8858	0.9235	0.7659	0.8844	0.8844	0.7772
PAN	0.8984	0.8937	0.8870	0.7502	0.8734	0.8734	0.7613

Tabla 4.6: Resultados obtenidos con los modelos U-Net, FPN, DeepLabV3+ y PAN sobre las imágenes del conjunto de *test* del dataset SWED

Capítulo 5

Conclusiones y propuestas de futuro

A lo largo del proyecto se ha explicado el desarrollo y evaluación del modelo de red neuronal entrenado para la detección de la línea de costa y segmentación de la interfaz agua-no agua a partir de imágenes de Sentinel-2, el cual ha logrado generar resultados prometedores en términos de exactitud y consistencia a nivel píxel. Sentinel ha demostrado ser una fuente de datos adecuada para esta tarea, ya que proporciona de manera efectiva características costeras al modelo gracias tanto a su resolución espectral como espacial.

Además, el uso de la red neuronal ha permitido capturar de manera automática y eficiente los patrones y características distintivas de la línea de costa, eliminando así la necesidad de un proceso manual de delineación. Esto no solo ahorra tiempo y recursos, sino que también garantiza una mayor objetividad en los resultados. Sin embargo, existe mucho margen aún de mejora.

Aunque el modelo ha demostrado en general un rendimiento sólido, existen algunos aspectos que pueden tenerse en cuenta para mejorar su precisión y aumentar aún más su capacidad de generalización y actuación en nuevas imágenes. En primer lugar, la base de datos usada para el entrenamiento, SWED, parte de unas verdades terreno que carecen de una precisión óptima. Si el modelo fuese entrenado con máscaras de alta precisión, como las del CGAT, los resultados mejorarían cuantitativamente. Además, podrían incorporarse al conjunto de entrenamiento imágenes donde existan diferentes condiciones atmosféricas, como la vista en la figura 4.6 (imagen 2), para conseguir que el modelo fuera capaz de aprender las estructuras y patrones de textura necesarios para realizar una segmentación precisa aún en esas situaciones.

De la misma manera, se pueden realizar pares de imágenes de Sentinel-1 y Sentinel-2 con el fin de nutrir al modelo con imágenes multiespectrales y sus correspondientes imágenes radar para poder disponer de información auxiliar, como la topografía del terreno, que enriquezca la segmentación y proporcione una visión más completa de la zona de estudio.

También, al igual que se realizó para la evaluación del modelo con el conjunto de datos del CGAT, se pueden generar máscaras de bordes (máscaras donde los píxeles con valor 1 correspondieran solo a la línea de costa) con el fin de calcular unas pérdidas más precisas en lo que a contornos se refiere, consiguiendo que el modelo preste una mayor atención a estas zonas sensibles.

Estos son solo algunos ejemplos de cómo podrían mejorarse los resultados obtenidos, con miras a explorar técnicas subpíxel para extraer una línea de costa vectorial y compararla con una línea de referencia terrestre de alta precisión. El campo del aprendizaje profundo avanza rápidamente, y existe también la posibilidad de que surjan nuevos algoritmos con capacidades superiores en períodos cortos de tiempo. Por lo tanto, es de vital importancia mantener actualizadas las metodologías de detección y segmentación mediante el uso de la Teledetección para ser lo más rigurosos y precisos posible.

Bibliografía

[1] R. Nicholls et al. «Coastal systems and low-lying areas». En: *Faculty of Science - Papers* (ene. de 2007).

[2] Marius Kuschnerus, Roderik Lindenbergh y Sander Vos. «Coastal change patterns from time series clustering of permanent laser scan data». En: *Earth Surface Dynamics* 9 (2021), págs. 89-103. DOI: 10.5194/esurf-9-89-2021.

[3] IPCC. «Climate Change 2023: Synthesis Report. A Report of the Intergovernmental Panel on Climate Change. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change». En: (2023). Ed. por Core Writing Team, H. Lee y J. Romero. (in press).

[4] Elizabeth H. Boak y Ian L. Turner. «Shoreline Definition and Detection: A Review». En: *Journal of Coastal Research* 2005.214 (2005), págs. 688-703. DOI: 10.2112/03-0071.1. URL: https://doi.org/10.2112/03-0071.1.

[5] Kayleigh J. Wyles et al. «Are Some Natural Environments More Psychologically Beneficial Than Others? The Importance of Type and Quality on Connectedness to Nature and Psychological Restoration». En: *Environment and Behavior* 51.2 (2019), págs. 111-143. DOI: 10.1177/0013916517738312. eprint: https://doi.org/10.1177/0013916517738312. URL: https://doi.org/10.1177/0013916517738312.

[6] Quantao Zhu et al. «Spatiotemporal Changes of Coastline over the Yellow River Delta in the Previous 40 Years with Optical and SAR Remote Sensing». En: *Remote Sensing* 13.10 (2021). ISSN: 2072-4292. DOI: 10.3390/rs13101940. URL: https://www.mdpi.com/2072-4292/13/10/1940.

[7] European Space Agency. *Sentinel-2 User Handbook*. Inf. téc. European Space Agency, 2015. URL: https://sentinel.esa.int/documents/247904/685211/Sentinel-2_User_Handbook.

[8] European Space Agency. *Sentinel-2B User Handbook*. Inf. téc. European Space Agency, 2017. URL: https://sentinel.esa.int/documents/247904/685211/Sentinel-2_User_Handbook.

[9] CREODIAS. *Sentinel-2 L1 Collection Dataset*. https://discovery.creodias.eu/dataset/sentinel-2-l1-collection. Accedido el 14 de junio de 2023.

[10] Copernicus Sentinel Data. *Overview of the Sentinel-2 Mission*. https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-2/overview. Accedido el 2 de mayo de 2023.

[11] Nobuyuki Otsu. «A Threshold Selection Method from Gray-Level Histograms». En: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (1979), págs. 62-66. DOI: 10.1109/TSMC.1979.4310076.

[12] J. Kittler y J. Illingworth. «Minimum error thresholding». En: *Pattern Recognition* 19.1 (1986), págs. 41-47. ISSN: 0031-3203. DOI: https://doi.org/10.1016/0031-3203(86)90030-0. URL: https://www.sciencedirect.com/science/article/pii/0031320386900300.

[13] John Canny. «A Computational Approach to Edge Detection». En: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8.6 (1986), págs. 679-698. DOI: 10.1109/TPAMI.1986.4767851.

[14] Akhlaq Husain et al. «Fractal dimension of coastline of Australia». En: *Scientific Reports* 11 (mar. de 2021). DOI: 10.1038/s41598-021-85405-0.

[15] J. Kittler y J. Illingworth. «Minimum error thresholding». En: *Pattern Recognition* 19.1 (1986), págs. 41-47. ISSN: 0031-3203. DOI: https://doi.org/10.1016/0031-3203(86)90030-0. URL: https://www.sciencedirect.com/science/article/pii/0031320386900300.

[16] Tri Dev Acharya, Anoj Subedi y Dong Ha Lee. «Evaluation of Water Indices for Surface Water Extraction in a Landsat 8 Scene of Nepal». En: *Sensors* 18.8 (2018). ISSN: 1424-8220. DOI: 10.3390/s18082580. URL: https://www.mdpi.com/1424-8220/18/8/2580.

[17] Cheng Qiao et al. «An Adaptive Water Extraction Method from Remote Sensing Image Based on NDWI». En: *Journal of the Indian Society of Remote Sensing* 40.3 (2012), págs. 421-433. ISSN: 0974-3006. DOI: 10.1007/s12524-011-0162-7. URL: https://doi.org/10.1007/s12524-011-0162-7.

[18] Y. Lecun et al. «Gradient-based learning applied to document recognition». En: *Proceedings of the IEEE* 86.11 (1998), págs. 2278-2324. DOI: 10.1109/5.726791.

[19] Chang Yuqing et al. «Soft Sensor Modeling of Process Industrial Process with Uncertain Information». En: *2020 Chinese Control And Decision Conference (CCDC)*. 2020, págs. 3342-3347. DOI: 10.1109/CCDC49329.2020.9163948.

[20] Catherine Seale et al. «Coastline detection in satellite imagery: A deep learning approach on new benchmark data». En: *Remote Sensing of Environment* 278 (2022), pág. 113044. ISSN: 0034-4257. DOI: https://doi.org/10.1016/j.rse.2022.113044. URL: https://www.sciencedirect.com/science/article/pii/S0034425722001584.

[21] The European Space Agency. *Sentinel-2 MSI User Guides - Processing Levels*. https://sentinels.copernicus.eu/web/sentinel/user-guides/sentinel-2-msi/processing-levels/level-2. Accedido el 20 de mayo de 2023.

[22] Magdalena Main-Knorn et al. «Sen2Cor for Sentinel-2». En: *Image and Signal Processing for Remote Sensing XXIII*. Ed. por Lorenzo Bruzzone. Vol. 10427. International Society for Optics y Photonics. SPIE, 2017, pág. 1042704. DOI: 10.1117/12.2278218. URL: https://doi.org/10.1117/12.2278218.

[23] Bernhard Mayer et al. *libRadtran - A Software Package for Radiative Transfer Calculations*. http://www.libradtran.org/doku.php. Accedido el 20 de mayo de 2023. 2020.

[24] Carmen Valdivieso-Ros, Francisco Alonso-Sarria y Francisco Gomariz-Castillo. «Effect of Different Atmospheric Correction Algorithms on Sentinel-2 Imagery Classification Accuracy in a Semiarid Mediterranean Area». En: *Remote Sensing* 13.9 (2021). ISSN: 2072-4292. DOI: 10.3390/rs13091770. URL: https://www.mdpi.com/2072-4292/13/9/1770.

[25] *Sentinel-2 MSI User Guide: Resolutions*. Copernicus Open Access Hub. Consultado el 16 de mayo de 2023.

[26] David Roy et al. «Best practices for the reprojection and resampling of Sentinel-2 Multi Spectral Instrument Level 1C data». En: *Remote Sensing Letters* 7 (nov. de 2016), págs. 1023-1032. DOI: 10.1080/2150704X.2016.1212419.

[27] Olivier Rukundo y Hanqiang Cao. «Nearest Neighbor Value Interpolation». En: *International Journal of Advanced Computer Science and Applications* 3 (nov. de 2012), 25:30. DOI: 10.14569/IJACSA.2012.030405.

[28] angeljohnsy.blogspot.com. *Nearest Neighbor Interpolation*. https://www.imageeprocessing.com/2017/11/nearest-neighbor-interpolation.html. Accedido el 15 de mayo de 2023. 2017.

[29] Krystian Wojcicki. *Nearest Neighbour Interpolation*. https://kwojcicki.github.io/blog/NEAREST-NEIGHBOUR. Accedido el 15 de mayo de 2023. 2020.

[30] Pankaj Parsania y Dr Virparia. «A Comparative Analysis of Image Interpolation Algorithms». En: *IJARCCE* 5 (ene. de 2016), págs. 29-34. DOI: 10.17148/IJARCCE.2016.5107.

[31] Valery Naranjo Ornedo - CVBLab. *Deep Learning aplicado al análisis de señales e imágenes. Fundamentos de redes neuronales*. Accedido el 14 de junio de 2023.

[32] Numerentur. *Gradiente Descendente*. https://numerentur.org/gradiente-descendente/. Accedido el 20 de mayo de 2023.

[33] Sai Balaji. *Binary Image Classifier CNN using TensorFlow*. https://medium.com/techiepedia/binary-image-classifier-cnn-using-tensorflow-a3f5d6746697. Accedido el 2 de mayo de 2023. 2020.

[34] KeepCoding. *Capas Pooling en Redes Neuronales Convolucionales*. https://keepcoding.io/blog/capas-pooling-red-neuronal-convolucional/. Accedido el 2 de mayo de 2023. 2022.

[35] Jason Brownlee. *Rectified Linear Activation Function for Deep Learning Neural Networks*. https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks. Accedido el 18 de mayo de 2023. 2019.

[36] OpenGenus IQ. *ReLU Activation*. https://iq.opengenus.org/relu-activation/. Accedido el 18 de mayo de 2023.

[37] Jeremy Zhang. *UNet — Line by Line Explanation*. https://towardsdatascience.com/unet-line-by-line-explanation-9b191c76baf5. Accedido el 22 de mayo de 2023.

[38] Mingxing Tan y Quoc V. Le. «EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks». En: *CoRR* abs/1905.11946 (2019). arXiv: 1905.11946. URL: http://arxiv.org/abs/1905.11946.

[39] Jia Deng et al. «ImageNet: A large-scale hierarchical image database». En: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, págs. 248-255. DOI: 10.1109/CVPR.2009.5206848.

[40] Muhammad Khalil et al. «Multi-Scale Network for Thoracic Organs Segmentation». En: *Computers, Materials and Continua* 70 (ene. de 2022), págs. 3251-3265. DOI: 10.32604/cmc.2022.020561.

[41] Mina Amiri, Rupert Brooks y Hassan Rivaz. *Fine tuning U-Net for ultrasound image segmentation: which layers?* 2020. arXiv: 2002.08438 [eess.IV].

[42] Diederik Kingma y Jimmy Ba. «Adam: A Method for Stochastic Optimization». En: *International Conference on Learning Representations* (dic. de 2014).

[43] Li Deng. «The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]». En: *IEEE Signal Processing Magazine* 29.6 (2012), págs. 141-142. DOI: 10.1109/MSP.2012.2211477.

[44] Tsung-Yi Lin et al. «Feature Pyramid Networks for Object Detection». En: *CoRR* abs/1612.03144 (2016). arXiv: 1612.03144. URL: http://arxiv.org/abs/1612.03144.

[45] Liang-Chieh Chen et al. «Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation». En: *Computer Vision – ECCV 2018*. Ed. por Vittorio Ferrari et al. Cham: Springer International Publishing, 2018, págs. 833-851. ISBN: 978-3-030-01234-2.

[46] Zheng Ma, Ming Li y Yu Guang Wang. «PAN: Path Integral Based Convolution for Deep Graph Neural Networks». En: *CoRR* abs/1904.10996 (2019). arXiv: 1904.10996. URL: http://arxiv.org/abs/1904.10996.

Parte II

Presupuesto

Capítulo 1

Presupuesto

1.1. Objetivo

En este apartado se pretende dar a conocer los costes, directos e indirectos, que han sido necesarios para la elaboración del proyecto, así como el tiempo dedicado por cada uno de los componentes del grupo.

1.2. Costes directos

Para el cálculo de los costes directos se tendrá en cuenta el sueldo de los empleados implicados en la elaboración del proyecto y el material, tanto componentes físicos (*hardware*) como programas informáticos (*software*). Así, en la elaboración de este Trabajo Fin de Grado han participado:

- Luis Correas Naranjo. Estudiante en prácticas del grado en Ingeniería en Geomática y Topografía y autor del proyecto.
- Jesús Manuel Palomar Vázquez. Profesor titular de universidad e investigador del Grupo de Cartografía Geoambiental y Teledetección y tutor de este trabajo.
- Francisco Pastor Naranjo. Ingeniero en Tecnologías y Servicios de Telecomunicaciones e investigador en el CVBLab y cotutor de este trabajo.

Con respecto al material, tanto los *softwares* como las librerías y módulos usados a lo largo del proyecto son de uso libre, por lo que no han requerido ningún otro coste más allá del *hardware* necesario para hacerlo funcionar.

En este caso, el ordenador que se ha usado tanto para la programación como para la redacción del documento es un *HP ProBook 640*.

1.3. Costes indirectos

Como coste indirecto se ha considerado el servidor donde se conectan los investigadores del Computer Vision and Behaviour Analysis Lab, así como las *Graphics processing unit (GPU)* a la que está conectado.

1.4. Costes totales

Así, el coste total del proyecto sumando los costes directos e indirectos desglosados en la Tabla 1.1, ha sido de 3.768,24€.

Aplicando el IVA del 21 % a los materiales usados (HP ProBook G1 Core i7 4600M 2.9 GHz 16GB — 128 SSD, Tarjeta gráfica NVIDIA Titan V y NAS Synology DS918) el presupuesto final asciende a **3.829,82€**.

Descripción	Uds.	Cantidad	Precio	Período de amortización (meses)	Período de uso (meses)	Coste
Doctor	€/h	30h	25	-	-	750,00
Graduado	€/h	50h	9,5	-	-	475,00
Estudiante	€/h	300h	7,5	-	-	2.250,00
HP ProBook G1 Core i7 4600M 2.9 GHz 16GB — 128 SSD	€	1	493,06	24	4	82,18
Tarjeta gráfica NVIDIA Titan V	€	1	3300,00	48	3	206,25
NAS Synology DS918	€	1	77,00	48	3	4,81
TOTAL	€	-	-	-	-	3.768,24

Tabla 1.1: Presupuesto total del proyecto

Parte III

Anejo I - Relación del trabajo con los objetivos de desarrollo sostenible de la Agenda 2030

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar**			X	
ODS 4. **Educación de calidad**				X
ODS 5. **Igualdad de género**				X
ODS 6. **Agua limpia y saneamiento**				X
ODS 7. **Energía asequible y no contaminante**				X
ODS 8. **Trabajo decente y crecimiento económico**			X	
ODS 9. **Industria, innovación e infraestructuras**			X	
ODS 10. **Reducción de las desigualdades**				X
ODS 11. **Ciudades y comunidades sostenibles**			X	
ODS 12. **Producción y consumo responsables**				X
ODS 13. **Acción por el clima**		X		
ODS 14. **Vida submarina**		X		
ODS 15. **Vida de ecosistemas terrestres**		X		
ODS 16. **Paz, justicia e instituciones sólidas**				X
ODS 17. **Alianzas para lograr objetivos**		X		

Tal y como se puede apreciar en la tabla anterior, los Objetivos de Desarrollo Sostenible que más importancia han tenido a lo largo de este proyecto han sido:

- **ODS 13.** Como se mencionó anteriormente, el cambio climático está causando un aumento del nivel del mar, provocando cambios rápidos e imprevistos en la línea costera. El modelo creado actúa contra este efecto monitorizando de forma instantánea las fluctuaciones que puedan ocurrir.

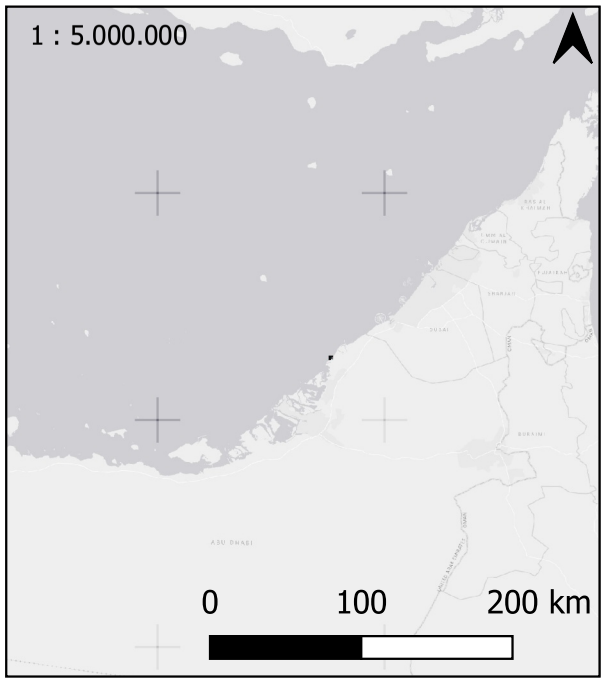
- **ODS 14.** Al igual que ocurre con el ODS 15, una mejor detección de la línea de costa puede ayudar a reducir peligros contra especies acostumbradas a vivir cerca de la costa.

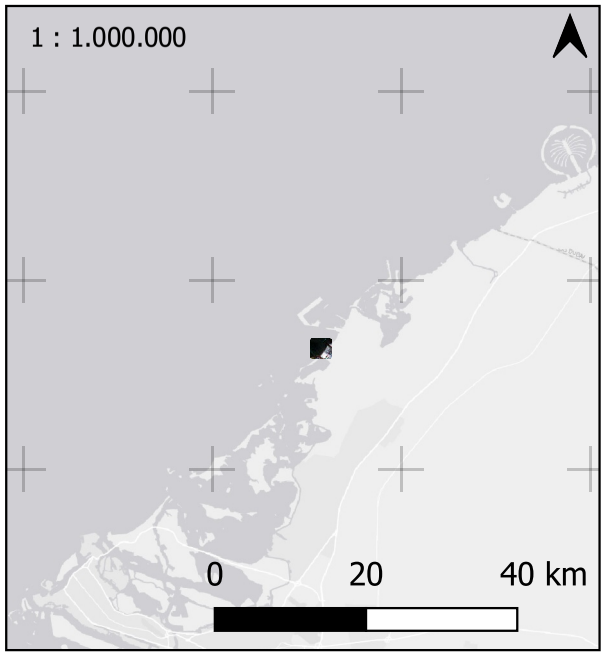
- **ODS 15.** En esta memoria se ha explicado la importancia de la detección de la línea de costa para el desarrollo de las zonas más próximas a la costa. Con este proyecto se puede mejorar la vigilancia y protección de las áreas más vulnerables.

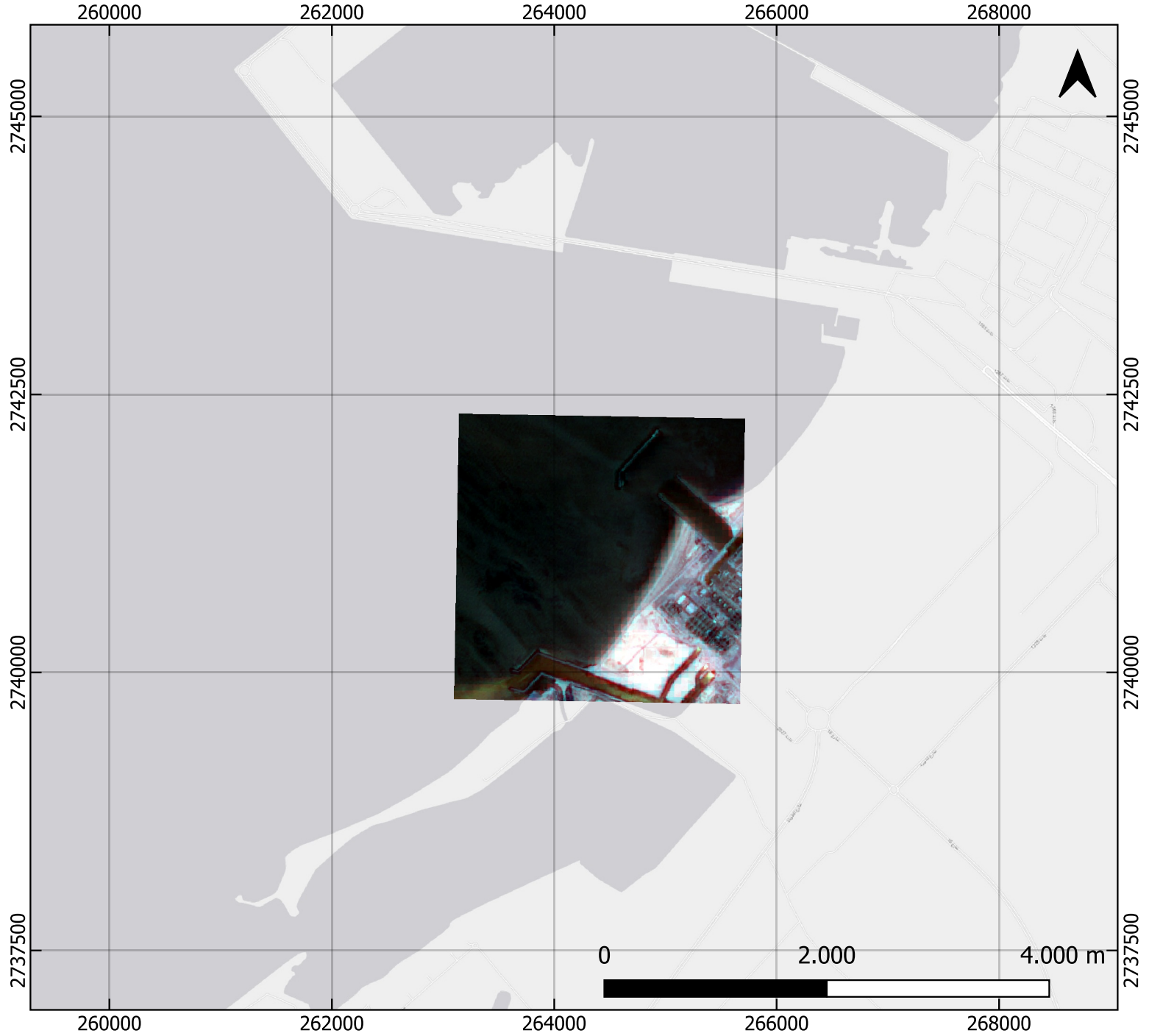
- **ODS 17.** Para llevar a cabo este trabajo, se estableció una colaboración profesional entre el CVBLab (laboratorio especializado en Inteligencia Artificial) y el laboratorio CGAT (especializado en Teledetección).

Parte IV

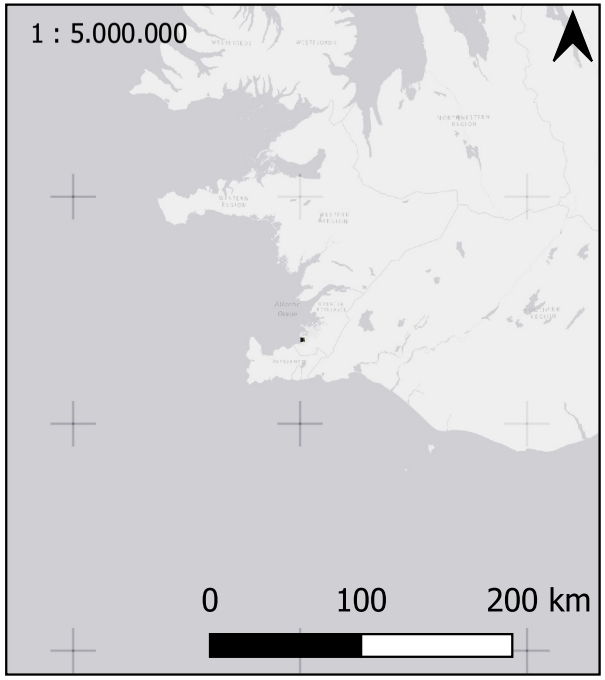
Anejo II - Mapas

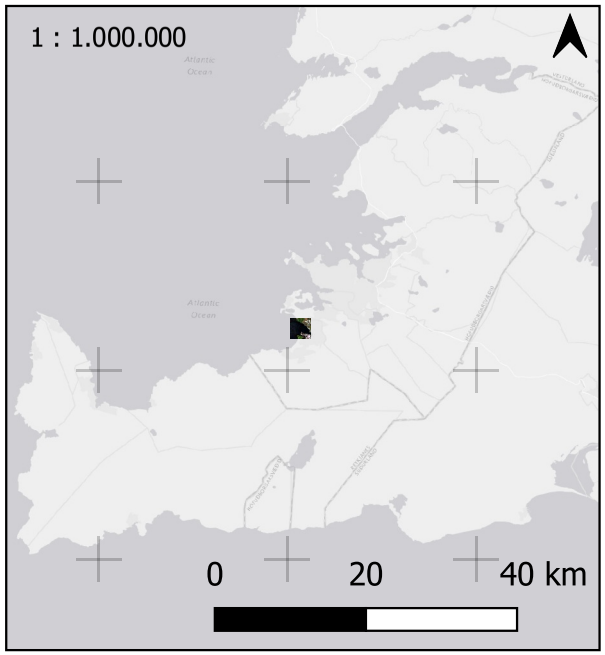


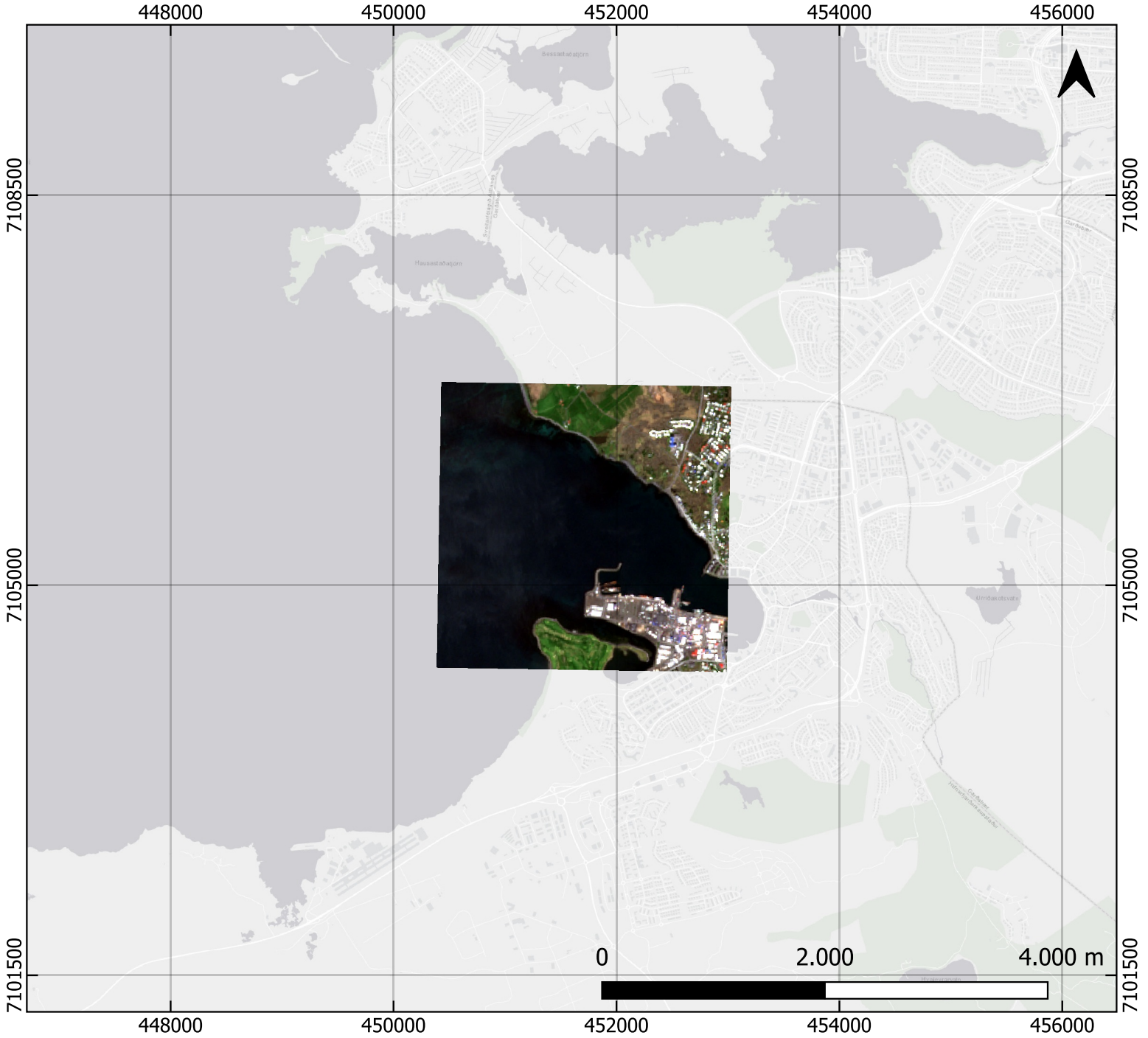




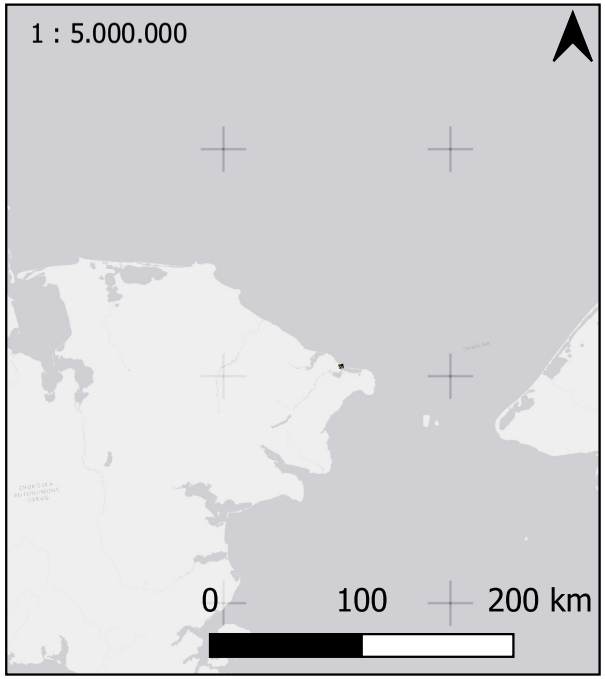
Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 40 N EPSG:32640 / WGS84	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - EAU**	**Escala** 1 : 50.000 **Fecha** 06/06/2023	**SWED** **0**

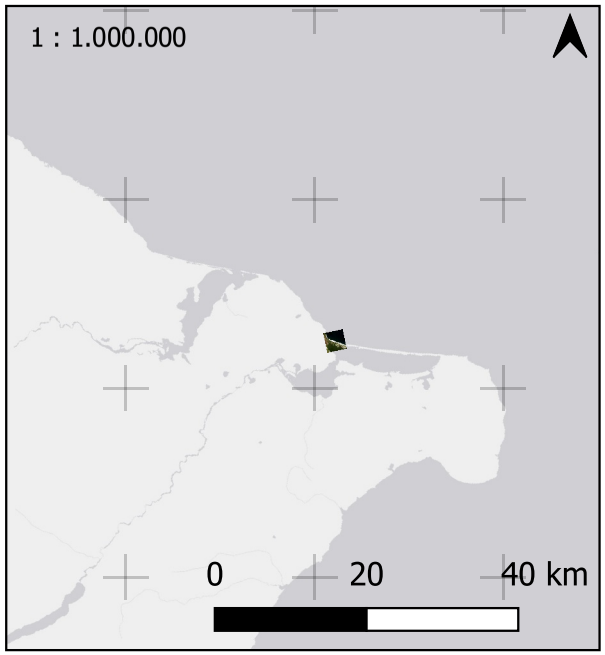


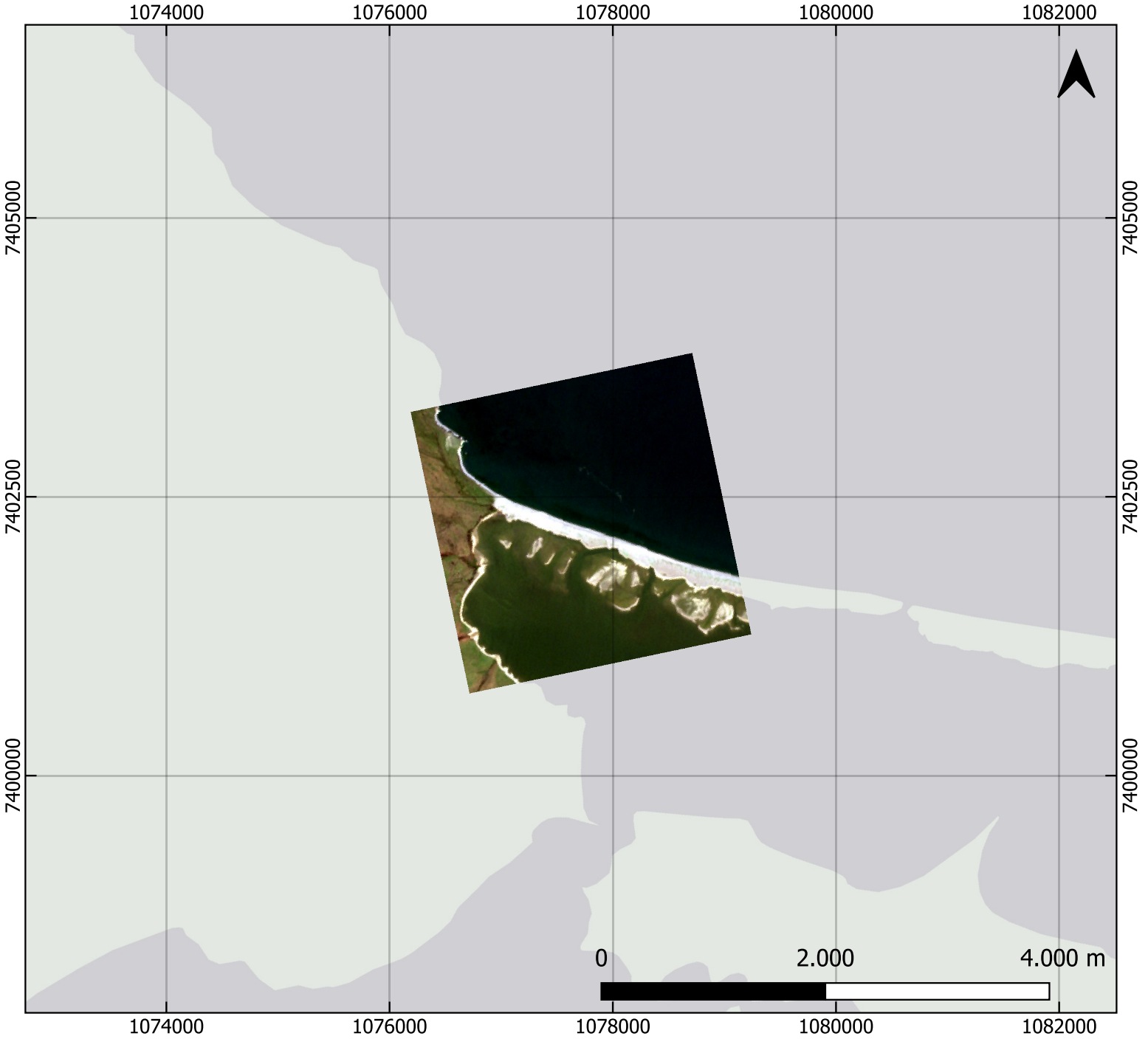




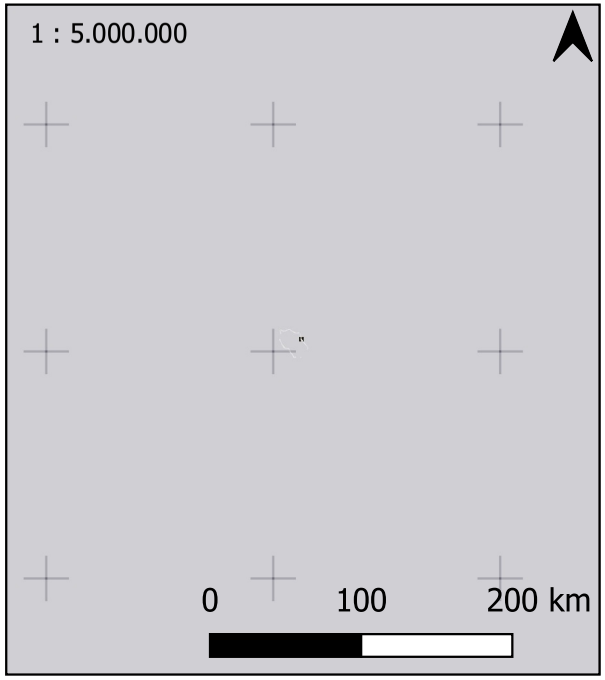
Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 27 N EPSG:32627 / WGS84	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - ISLANDIA**	**Escala** 1 : 50.000 **Fecha** 06/06/2023	**SWED** **1**

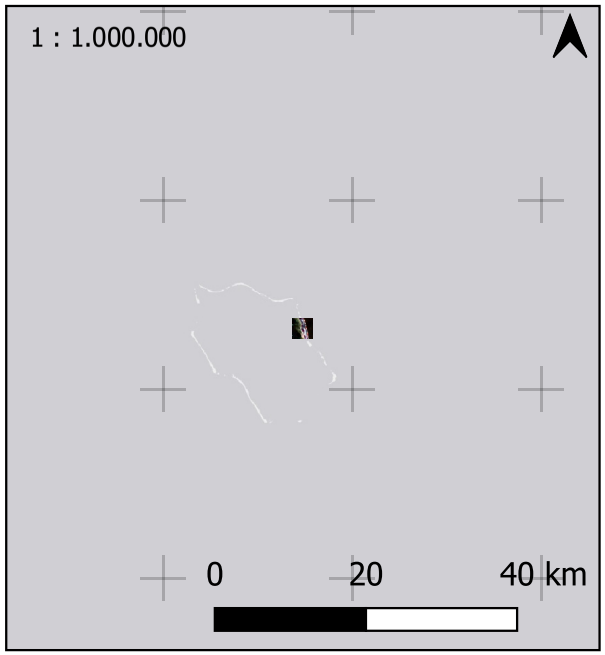


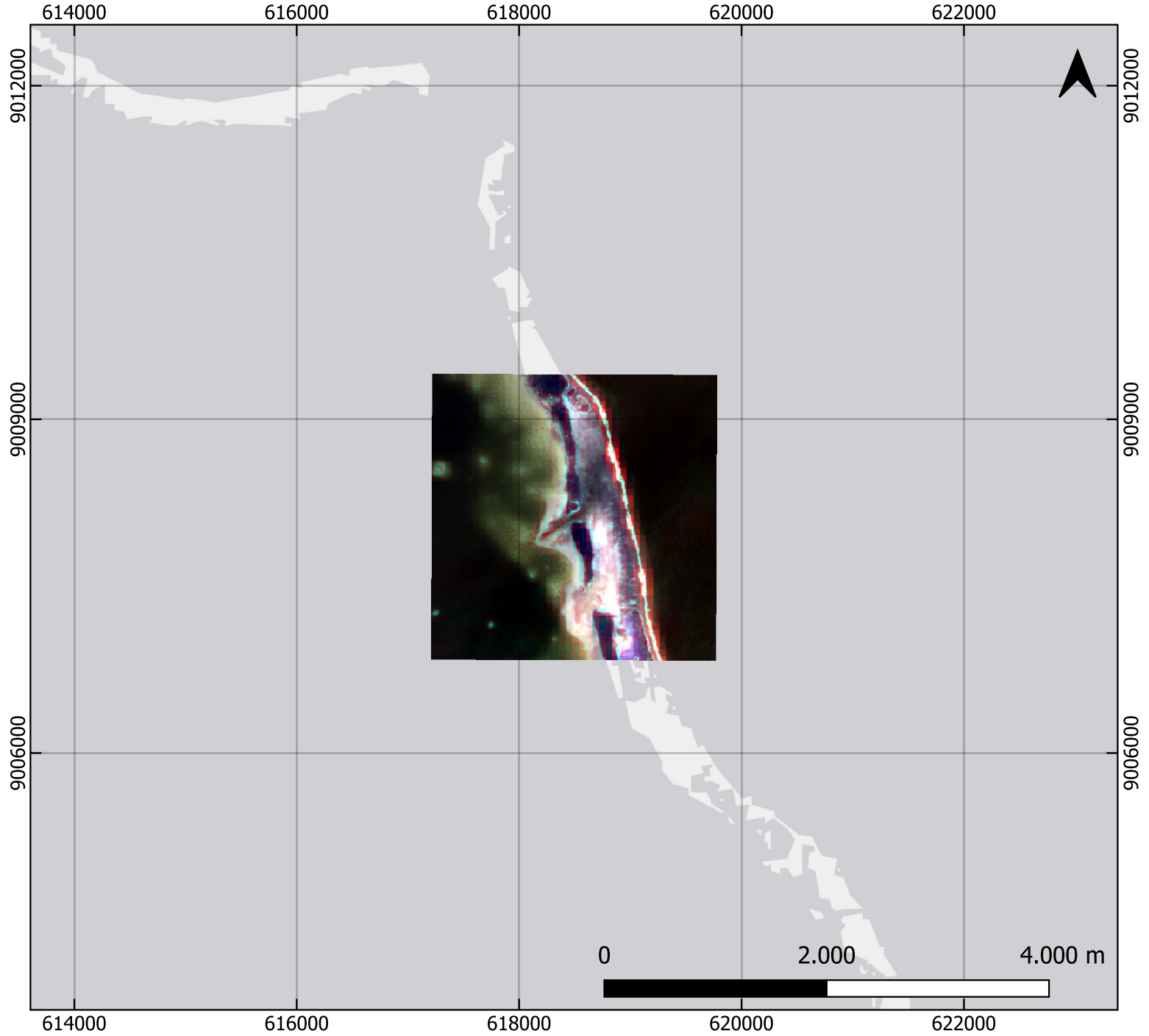




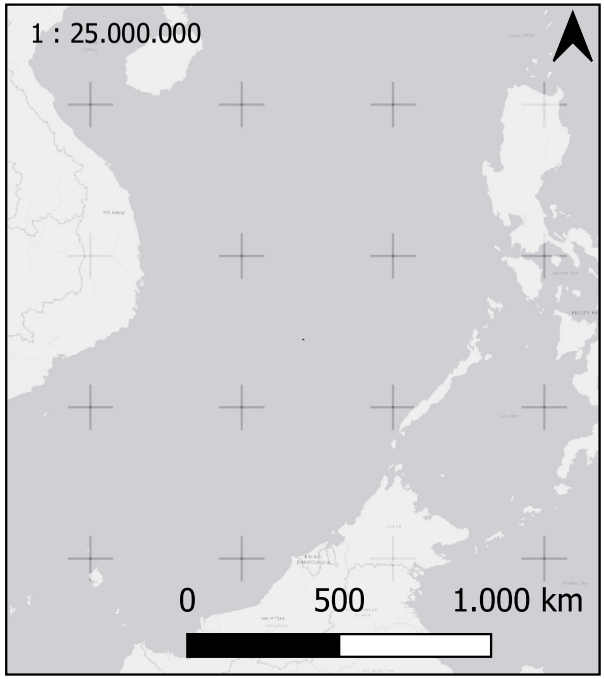
Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 60 N EPSG:32660 / WGS84	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - RUSIA**	**Escala** 1 : 50.000 **Fecha** 06/06/2023	**SWED** **2**

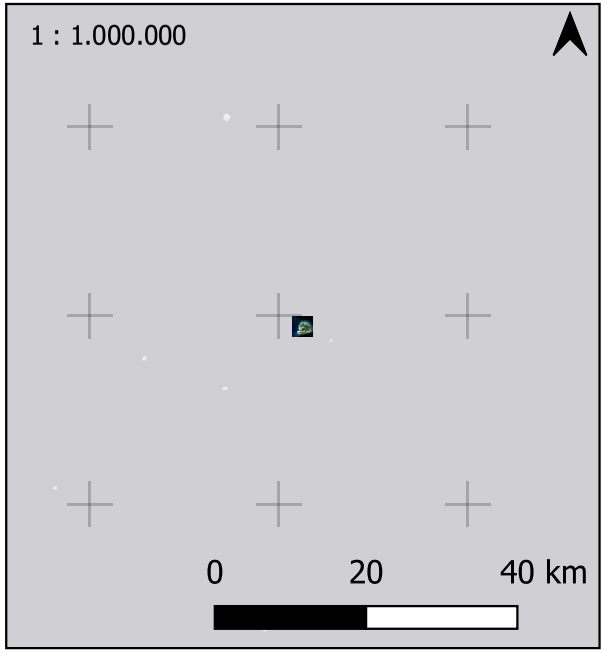


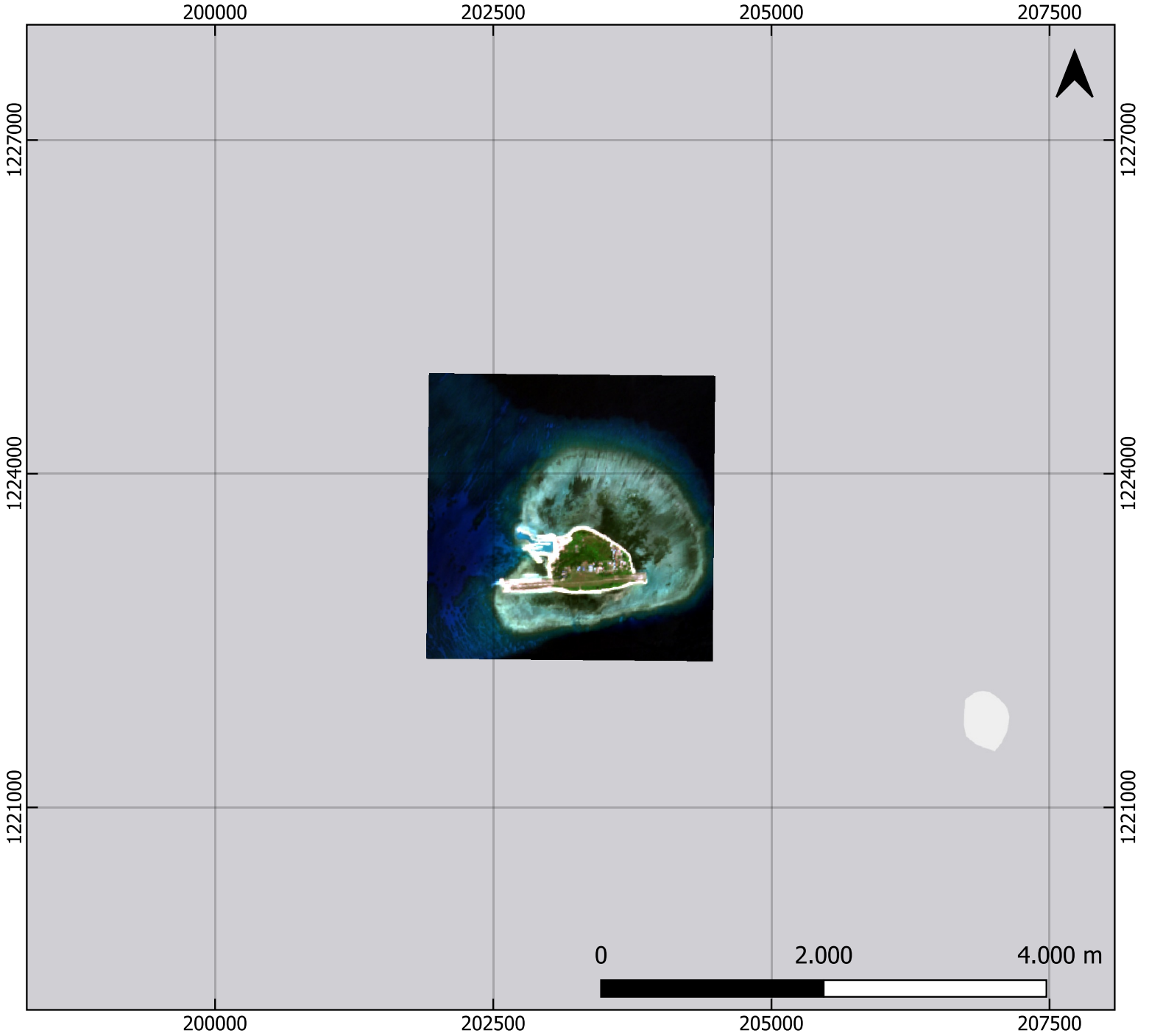




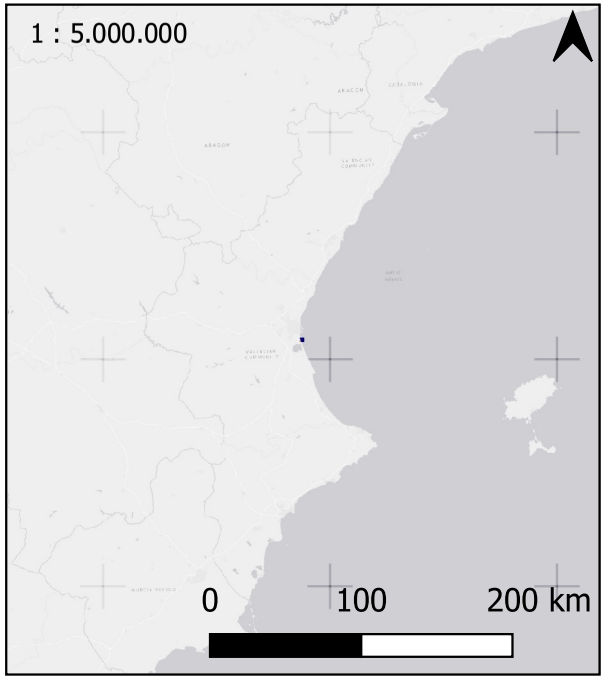
Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 4 S EPSG:32704 / WGS84	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - ISLAS COOK**	**Escala** 1 : 50.000 **Fecha** 06/06/2023	**SWED** **3**

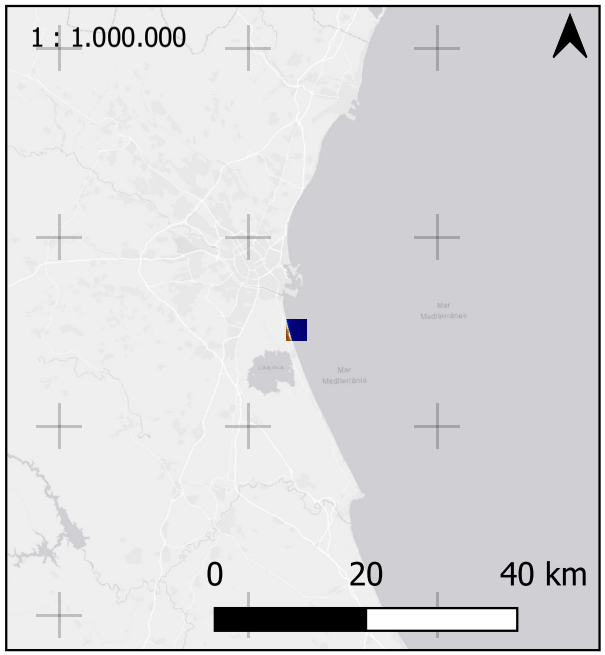


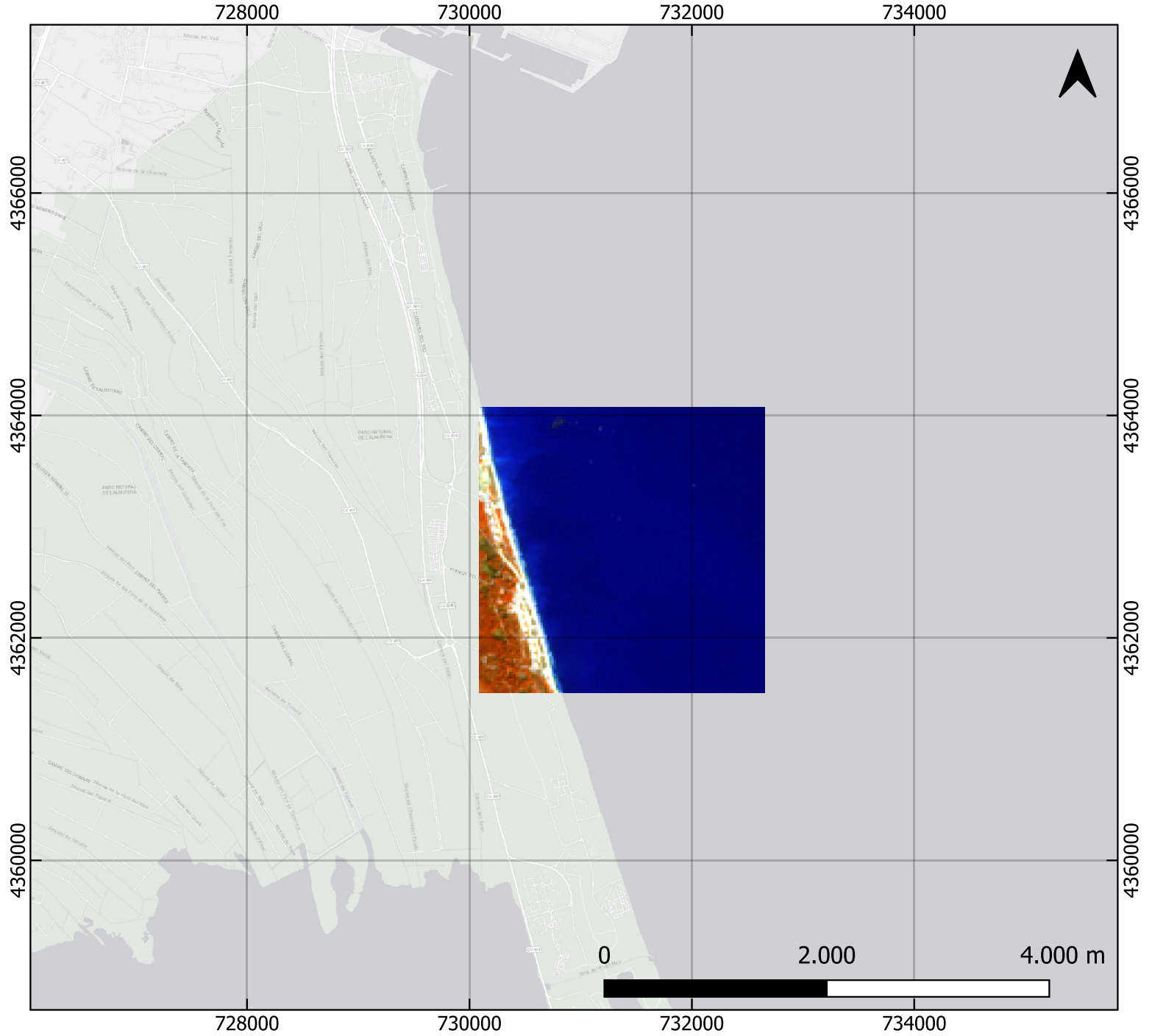




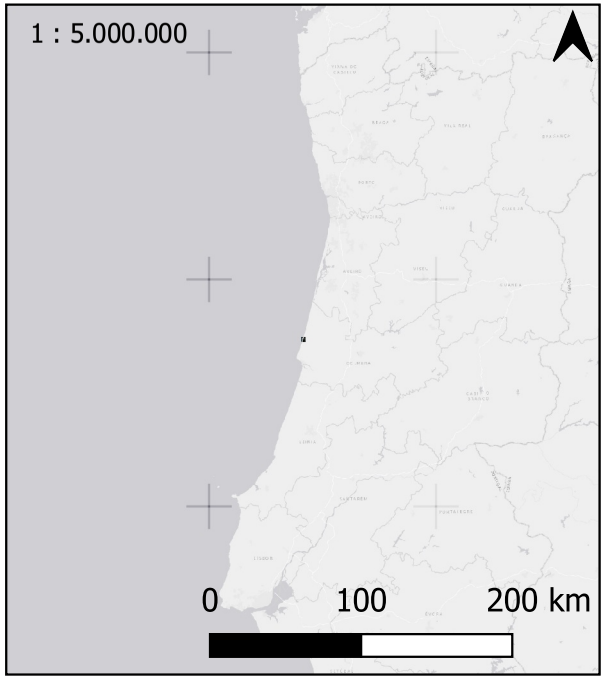
Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 50 N EPSG:32650 / WGS84	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - ISLAS SPRATLY**	**Escala** 1 : 50.000 **Fecha** 06/06/2023	**SWED** **4**

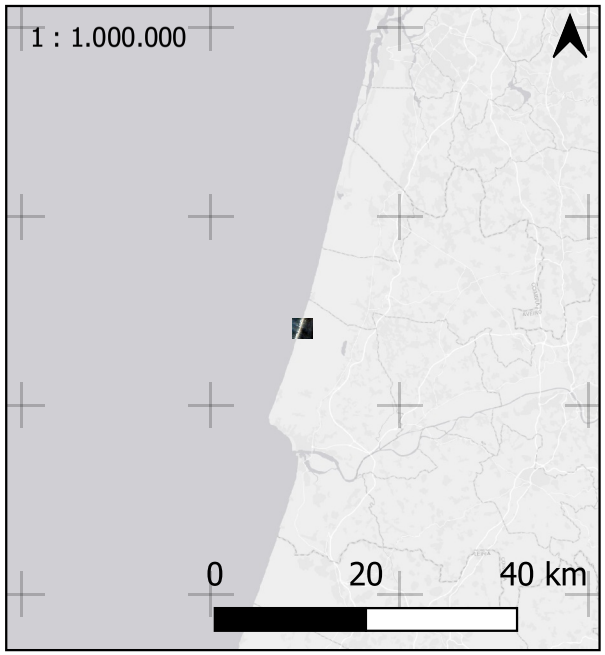


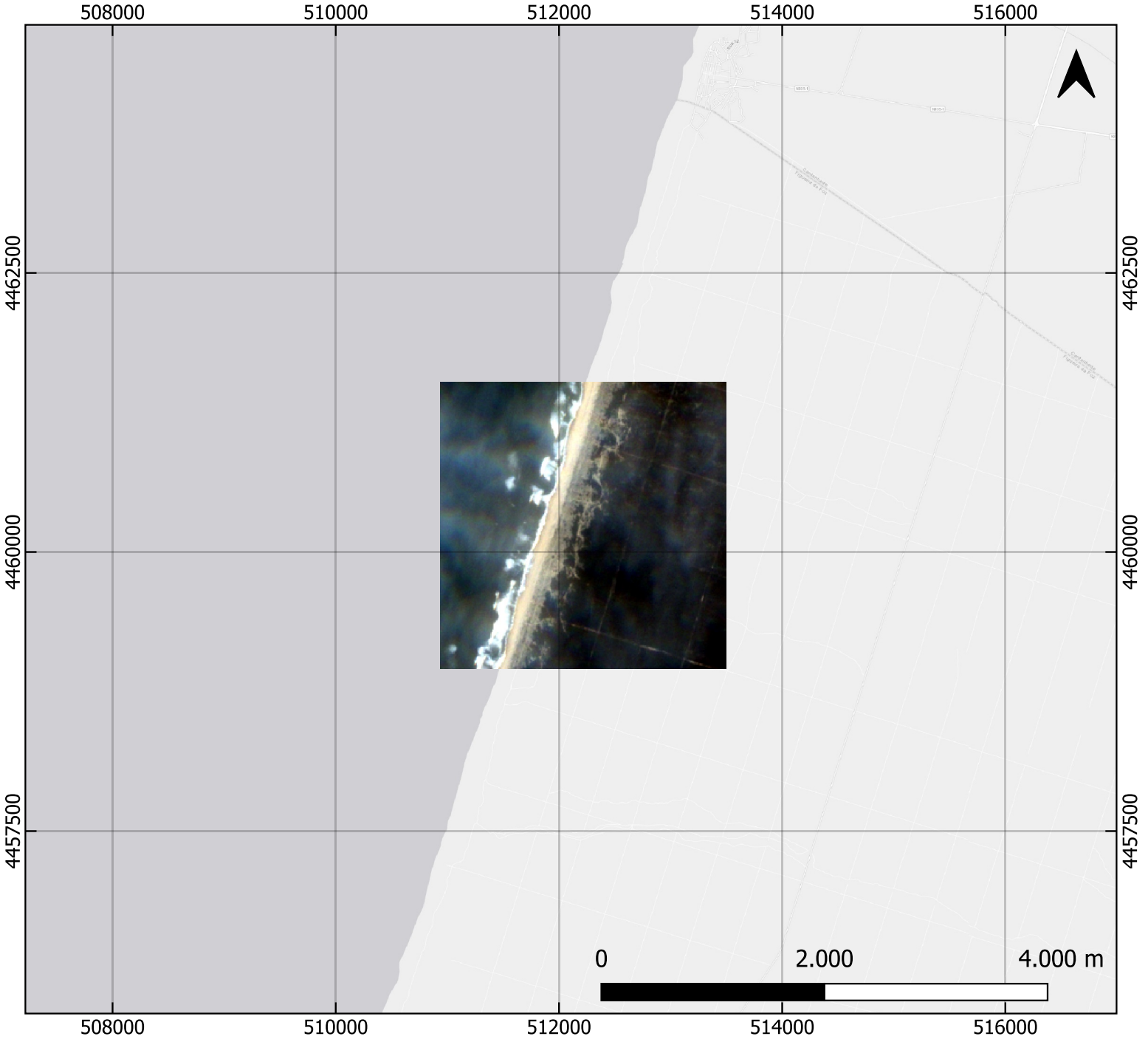




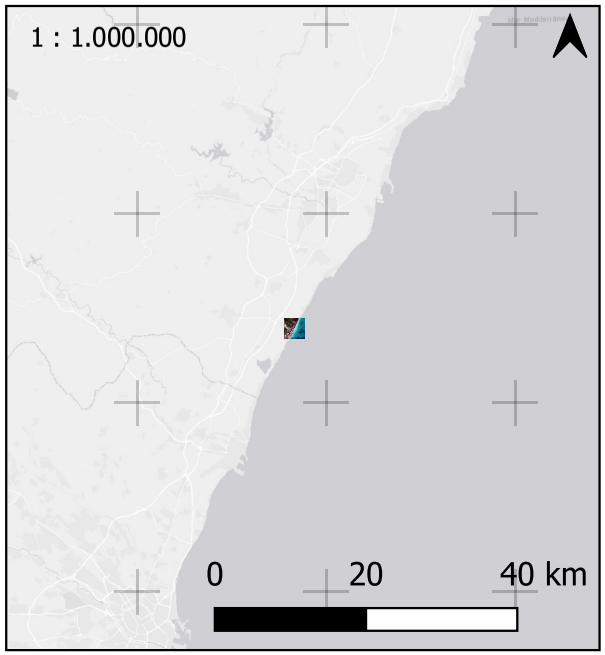
	Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 30 N EPSG:25830 / ETRS89	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - VALENCIA**	**Escala** 1 : 50.000 **Fecha** 07/06/2023	**CGAT** **1**

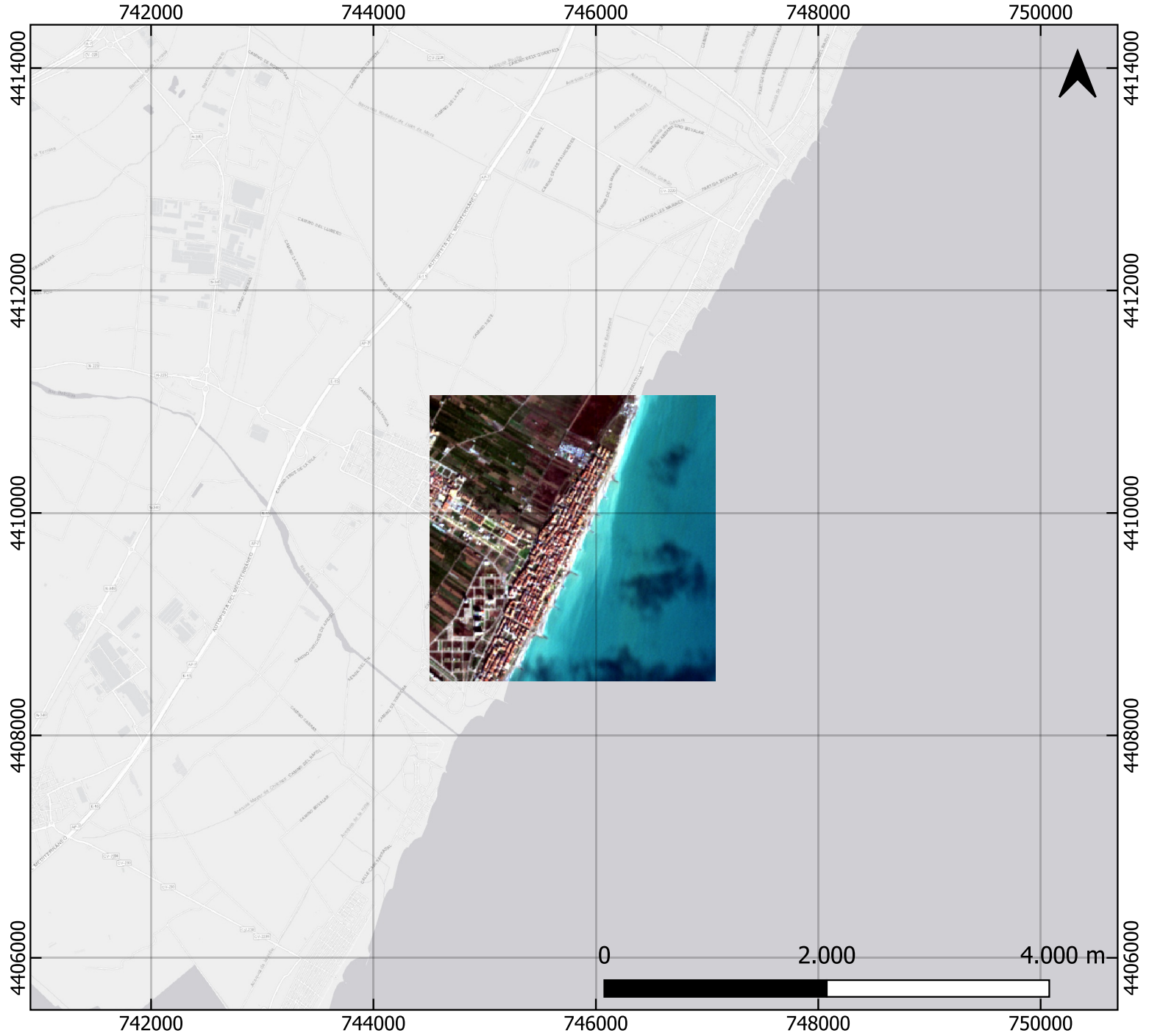




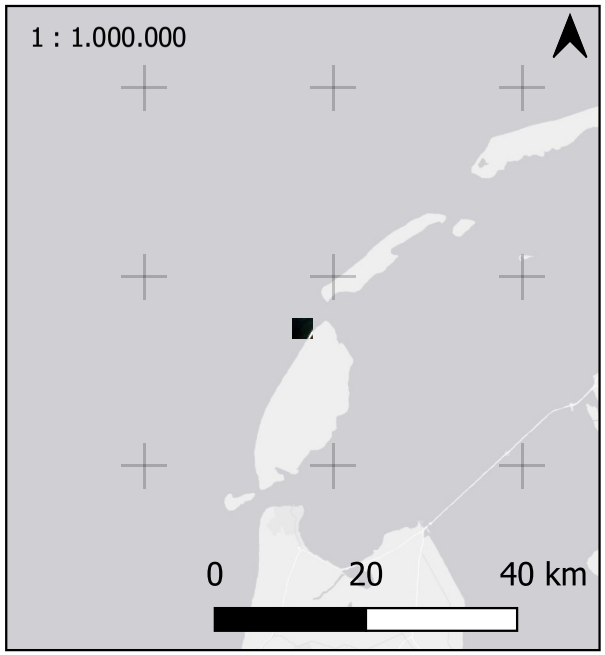


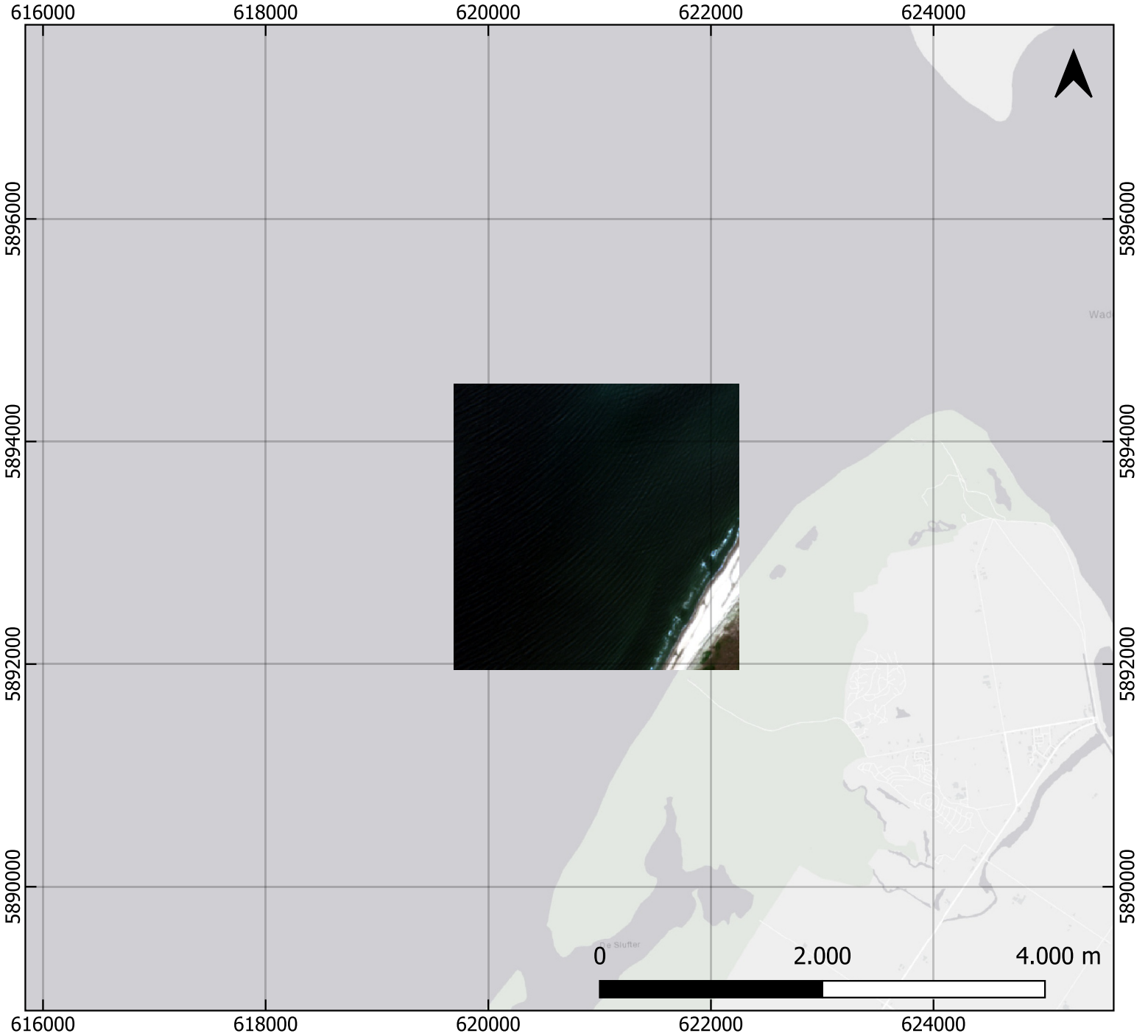
	Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 29 N EPSG:32629 / WGS84	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - PORTUGAL**	**Escala** 1 : 50.000 **Fecha** 07/06/2023	**CGAT** **2**





	Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 30 N EPSG:32360 / WGS84	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - CASTELLÓN**	**Escala** 1 : 50.000 **Fecha** 07/06/2023	**CGAT** **3**





	Autor Luis Correas Naranjo	**Sistema de referencia** COORDENADAS UTM ZONA 31 N EPSG:32631 / WGS84	**MAPA DE SITUACIÓN Y EMPLAZAMIENTO - PAISES BAJOS**	**Escala** 1 : 50.000 **Fecha** 07/06/2023	**CGAT** **4**