Universidad de Valladolid

FACULTAD DE CIENCIAS

TRABAJO DE FIN DE GRADO

Grado en Física

SEGMENTACIÓN SEMÁNTICA MULTICATEGORÍA DE IMÁGENES TODO CIELO MEDIANTE REDES NEURONALES

Autor: Sergio García Pajares

Tutores: Roberto Román Diez y Juan Carlos Antuña Sánchez

Año 2024

Este trabajo fue realizado dentro del Grupo de Óptica Atmosférica de la Universidad de Valladolid.

Trabajo de Fin de Grado, Universidad de Valladolid

Depositado el día: 23 de mayo de 2024.
Defendido el día: 31 de mayo de 2024.

Recursos adicionales en línea: github.com/sergiogpajares/TFG-fis

All models are wrong,
but some are useful.

GEORGE E. P. BOX

Índice general

AGRADECIMIENTOS

Es muy difícil escribir los agradecimientos de un trabajo que cierra una gran etapa de la vida de una persona. En este camino he conocido a mucha gente y perdido el contacto con mucha otra. Un camino de altibajos y traspiés lleno de aciertos y errores que me han forjado como persona y me han traído hasta aquí. Le tengo mucho que agradecer a mucha gente y, aún a riesgo de olvidar alguien importante, me gustaría dar un agradecimiento especial:

A mi tutor, Roberto, al que seguramente deberían construir una estatua en la entrada de la facultad. Es de esas personas que, de alguna manera, me hacen creer en la docencia universitaria.

A mi hermana, Mária, por pelear por mi durante todos estos años. Te necesito más de lo que estoy dispuesto a admitir.

A mis padres, Celes y María Jesús, por enseñarme, cada uno a su manera, su forma de entender la vida.

A mi tía, María Cristina, por gritarme una y otra vez que hiciera los deberes de matemáticas.

A mi abuelo, Jesús, que un día decidió que eramos suficientemente mayores para caminar solos -cuanto te equivocabas- y decidió irse. Me enseñó que ante la adversidad no cabe otra cosa sino una sonrisa.

A mis abuelos; Juana, Celestino y Juliana; por enseñarme el significado de la palabra *amor*.

A los profesores del Bachillerato Internacional del IES Jorge Manrique de Palencia y, en especial, a Miguel Ángel y Arturo; por haber sido mis mentores en una de las épocas más difíciles de mi vida. Sin vosotros, con total certeza, no sería quien soy.

Aún recuerdo cuando Isidro González, profesor de física computacional en la Universidad de Oviedo, nos repetía incansablemente: «*Si trabajáis duro, al final de curso seréis un cuarto de físicos*». Desde aquel entonces han pasado seis años, he estado en cuatro universidades y he tenido siete pisos de estudiantes. Finalmente, creo que hoy ha llegado ese día del que Isidro tanto hablaba.

ABSTRACT

Clouds play a leading role in earth's climate due to their importance in the planet's energetic balance. Also, making very short-time forecast is crucial to an electric system that wants to relay more and more on photovoltaic systems. For those reasons, clouds properties need to be accurately and cost-effectively measured. An instrument that fulfills those requirements are ground-based all-sky cameras. Measurements with enough temporal and spatial resolution requires image analysis automatization. One step to archive it is semantic segmentation of the images.

We present a light-weighted model based on Convolutional Neural Networks. It follows an encoder-decoder architecture like U-Net using VGG16 as encoder. Transfer learning is used to train the model on a dataset from GOA-UVa research group. The dataset is built from six different cameras at two locations and consist of day and night whole-sky images with a variety of weather conditions. The images are labelled into five categories: N/A, clear sky, sun, thick clouds, and thin clouds. Also, data augmentation techniques are used.

The inner workings of the model are explained. A general approach on neural networks is given. Then, convolution, transposed convolution and pooling operations are explained with some visuals to promote intuition. Also, how these operations build up convolutional layers is shown.

The model achieves a True Positive rate (TP) up to 97% on some cases. It performs better on clear day images, but very good results are obtained with overcast conditions as well. It improves previous work on the field by 10% on TP. Cloud Cover can be derived from the segmentation mask. The model archives a ± 1 oktas confidence of up to 92%. This result is like the ones obtained by models trained specifically at Cloud Cover classification.

Keywords :
Cloud segmentation, All-sky images, Cloud Cover, Convolutional Neural Networks.

RESUMEN

Las nubes juegan un papel fundamental en el clima terrestre debido a su importancia en el balance energético de la tierra. Además, producir predicciones meteorológicas a muy corto plazo es crucial para una red eléctrica que quiere depender más y más de la energía fotovoltaica. Por estos motivos, las propiedades de las nubes deben ser medidas de manera precisa y asequible. Un instrumento que cumple estos requisitos son las cámaras todo cielo. Mediciones con suficiente resolución espacial y temporal requieren la automatización del análisis de las imágenes. Un paso para conseguirlo es la segmentación semántica de las mismas.

Presentamos un modelo ligero basado en Redes Neuronales Convolucionales. Seguimos una arquitectura *encoder–decoder* similar a U-Net con la red VGG16 como *enconder*. Técnicas de transferencia de aprendizaje son usadas para entrenar el modelo en un conjuto de datos del grupo de investigación GOA-UVa. El conjunto de datos se construye con seis cámaras en 2 ubicaciones distintas y consiste en imágenes todo cielo con condiciones meteorológicas variadas. Las imágenes se etiquetan en cinco categorías: N/A, cielo despejado, sol, nubes espesa y nube fina. Además, se emplean técnicas de aumentación de datos.

El funcionamiento del modelo se explica, proporcionando una aproximación general a las redes neuronales. Después, se explican las operaciones de convolución, convolución transpuesta y pooling, proporcionando ilustraciones para desarrollar la intuición. Finalmente se muestra como estas operaciones conforman las capas convolucionales.

El modelo consigue un ratio de positivos verdaderos (TP) de hasta el 97 % en algunos casos. Funciona mejor en imágenes diurnas despejadas, pero también se obtienen muy buenas condiciones con cielos totalmente cubiertos. Mejora trabajos previos en este campo en un 10 % en TP. La medida de la cobertura nubosa se puede obtener a partir de la máscara de segmentación. El modelo consigue una confianza de ± 1 octa de hasta el 92 %. Este resultado es similar a los obtenidos por modelos entrenados específicamente en la clasificación de la cobertura nubosa.

Palabras clave :
Segmentación de nubes, imágenes todo cielo, cobertura nubosa, redes neuronales convolucionales.

Capítulo 1

INTRODUCCIÓN

Los algoritmos de inteligencia artificial, como las redes neuronales, han llegado para quedarse y están marcando enormes hitos en la ciencia y la tecnología. Uno de sus muchos campos de aplicación son la meteorología y el estudio del cielo. Vamos a ver por qué el estudio de las nubes es importante, cómo se ha abordado hasta ahora y las nuevas metodologías basadas en inteligencia artificial.

1.1 NUBES

Las nubes cubren aproximadamente dos terceras partes de la superficie terrestre (Boucher *et al.*, 2013, p. 579) y juegan un papel esencial en la meteorología y el clima. El ciclo hidrológico, por ejemplo, es crucial para mantener nuestra forma de vida porque el agua es necesario, entre otras cosas, para las personas, los animales y la producción de alimentos. Todos hemos sufrido de diferentes maneras los efectos de las sequías de la última década en España (Fresneda *et al.*, 2020, p. 37), llegándose a producir cortes de agua al consumo humano en algunos municipios (Miñano, 2023).

Por otro lado, las nubes tienen un fuerte impacto en el clima por su interacción con la radiación. Las nubes pueden reflejar la radiación incidente del sol y también pueden absorber la radiación emitida por la tierra y re-emitirla, devolviendo parte a la superficie terrestre. Por ello, las nubes son cruciales en el balance radiativo de la tierra (Boucher *et al.*, 2013, p. 576–578). En función de sus características, pueden calentar y enfriar la tierra, teniendo, en promedio, un efecto refrigerante en el planeta (Forster *et al.*, 2023, p. 933–935).

Del mismo modo, una nube sobre un parque solar puede reducir drástica y repentinamente la producción eléctrica. La predicción a muy corto plazo (desde minutos hasta una hora) y corto plazo (hasta seis horas) de la irradianza solar es muy importante para la operación de una red eléctrica que dependa más y más de la energía fotovoltaica (Nie *et al.*, 2024).

La influencia que tienen las nubes depende de sus propiedades, por lo que es necesario poder medirlas de una manera confiable; eficiente; y, preferiblemente, económica. Algunas propiedades importantes son: cobertura nubosa (CC por sus siglas en inglés), profundidad

óptica nubosa y fase termodinámica (agua, hielo, vapor) (Peris-Ferrús, 2021, p. 67–72). Una de las propiedades más destacadas y que abordaremos en este trabajo es la cobertura nubosa (Peris-Ferrús, 2021; Nie *et al.*, 2024), es decir, la cantidad de cielo que cubren las nubes. Habitualmente se mide en octavos de cielo cubiertos, llamados *octas*.

Para ilustrar la complejidad y variedad de las diferentes nubes, vamos a ver someramente su clasificación. Las nubes se clasifican de acuerdo a las formas características que representan. La clasificación se subdivide en *géneros*, *especies* y *variedades* (Tapakis *et al.*, 2013), de forma similar a como se hace con animales y plantas. La figura 1.1 muestra los 10 géneros de nubes y la tabla 1.1 algunas de las especies y variedades más comunes. Existen otras clasificaciones de nubes en las que no entraremos, por ejemplo, en función de su altura sobre el suelo o de su fase termodinámica (Tapakis *et al.*, 2013).

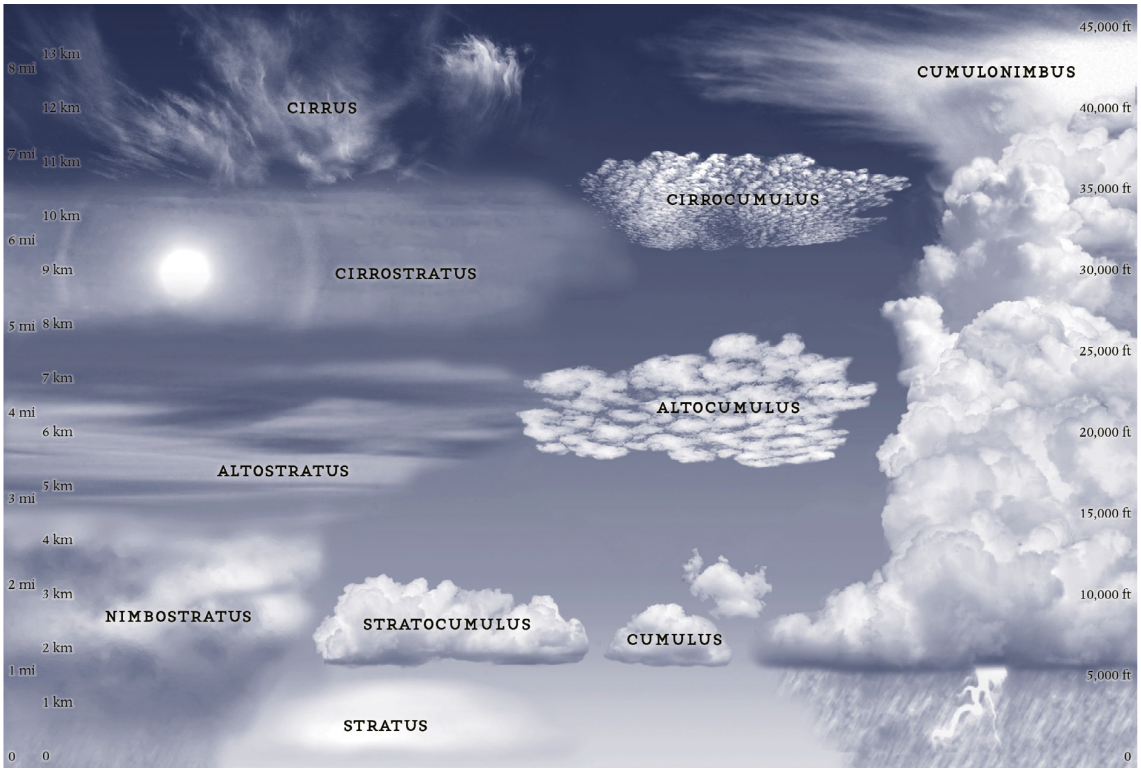


Figura 1.1: Los 10 géneros de nubes de acuerdo a la Organización Meteorológica Mundial. Crédito Tapakis *et al.* (2013).

Tabla 1.1: Los 10 géneros de nubes y sus especies y variedades más comunes. Entre paréntesis abreviaciones aceptadas. (Fitzgerald, 2017)

Género	**Especie**	**Variedad**
Cirrus	uncinus, fibratus, spissatus, castellanus	intortus, radiatus, vertebratus
Cirrostratus	nebulosus, fibratus	–
Cirrocumulus	castellanus, floccus, lenticularis	undulatus
Altocumulus	castellanus, floccus, lenticularis	tranlucidus, opacus, undulatus, perlucidus
Altoestratus	–	translucidus, opacus
Nimbostratus	–	–
Stratocumulus	castellanus, lenticularis	perlucidus, translucidus, opacus
Stratus	fractus, nebulosus	–
Cumulonimbus	calvus, capillatus	–
Cumulus	fractus, humilis, mediocris, congestus	–

1.1.1 EQUIPOS DE MEDIDA DE NUBES

La clasificación y detección de nubes se realiza con diversos instrumentos de medida que clasificaremos en dos tipos: terrestres y satélitales. Los instrumentos en tierra permiten medir datos para ubicaciones geográficas específicas, mientras que los satélites pueden monitorizar la formación y movimiento de nubes sobre una gran superficie (Tapakis *et al.*, 2013).

Los equipos terrestres se pueden dividir, a su vez, en tres grandes grupos: medidores de radiación incidente; emisores de radiación, que emiten un pulso electromagnético y miden la radiación reflejada; y cámaras espectrales en los espectros visible, infrarrojo o ultravioleta.

Entre los medidores de radiación incidente se encuentran, entre otros, los pirómetros y los fotómetros. Un pirómetro mide la radiación incidente en función de la longitud de onda y un fotómetro la cantidad total de radiación incidente (Tapakis *et al.*, 2013).

Entre los emisores de radiación se encuentran, entre otros: los celiometros, radares y LIDAR. Un celiometro consta de un láser vertical que mide normalmente en el infrarojo cercano. Un radar consta de una antena parabólica que emite microondas y permite detectar la posición, velocidad y tipo de precipitaciones. Un LIDAR hace un barrido de pulsos láser y está diseñado para medir nubes y estimar su altitud (Tapakis *et al.*, 2013).

Las cámaras espectrales pueden capturar toda la bóveda celeste o solo parte de esta. En el primero caso, estas cámaras se denominan *cámaras todo cielo*. Las cámaras infrarrojas IR se basan en la diferencia entre la radiación IR reflejada y emitida por distintos tipos de nubes y el cielo. Las cámaras del espectro visible, permiten estudiar las nubes en función de su color e intensidad. Las cámaras ultravioleta miden las diferente absorción de luz ultravioleta por distintas nubes (Tapakis *et al.*, 2013).

En este trabajo, vamos a usar imágenes del cielo en el espectro visible diurno y nocturno tomadas con cámaras todo cielo (véase la figura 1.2). Las imágenes provienen de una red de cámaras de cielo propiedad del Grupo de Óptica Atmoférica de la Universidad de Valladolid (GOA-UVa)[1]. Los detalles sobre estas cámaras y el proceso para la obtención de las imágenes lo trataremos en el capítulo 2.

Figura 1.2: Ejemplos de imágenes del cielo tomadas con cámaras todo cielo en el espectro visible. Crédito GOA-UVA.

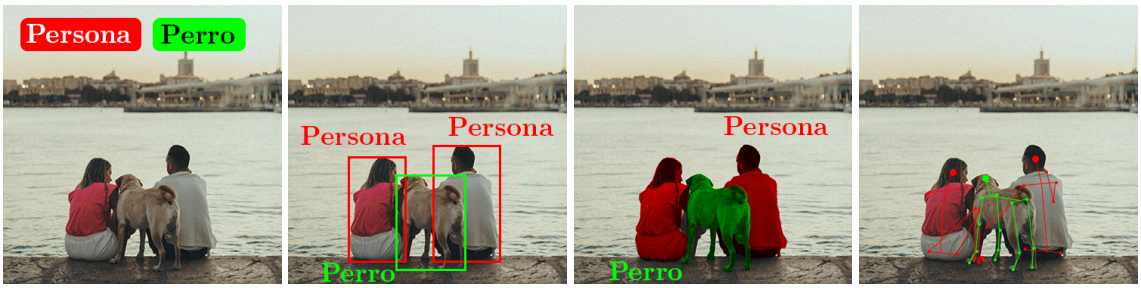
[1] Más información en su página web goa.uva.es.

Este tipo de cámaras, se emplean habitualmente para medir la cobertura nubosa (Tapakis *et al.*, 2013), pero también se han empleado para la medida de radiancia y luminiscencia (Román; Antón *et al.*, 2012), la predicción de la irradiancia solar (Hendrikx *et al.*, 2024), y la medición de propiedades de aerosoles (Román; Torres *et al.*, 2017; Román; Antuña-Sánchez *et al.*, 2022).

La determinación de la cobertura nubosa a partir de las cámaras se realiza habitualmente por observadores humanos entrenados al respecto. Esto requiere una gran cantidad de trabajo humano, lo que resulta en una baja resolución espacial y temporal de los datos disponibles. Además, es subjetivo, debido a las diferencias que puede haber entre diferentes observadores (WMO *et al.*, 2012, p. 8). Por ello, es de suma importancia la automatización de este proceso.

1.2 SEGMENTACIÓN SEMÁNTICA

La visión artificial se agrupa clásicamente[2] en cuatro grandes tareas: *clasificación*, *detección*, *segmentación* y *estimación de postura*. La clasificación busca decidir que elementos están presentes en una imagen. La detección persigue determinar la posición de un objeto en una imagen y la categoría de este objeto. La segmentación busca determinar qué píxeles corresponden a un objeto o categoría. La estimación de postura busca determinar la posición de las articulaciones de una persona o animal. La figura 1.3 muestra ejemplos de estas tareas.



(a) Clasificación (b) Detección (c) Segmentación (d) Postura

Figura 1.3: Tareas clásicas de visión artificial.

Nos vamos a centrar en el problema de la segmentación de una imagen. Existen tres tipos de tareas de segmentación: *segmentación de instancias*, *segmentación semántica* y *segmentación panóptica*. La segmentación de instancias busca indicar por separado los píxeles que corresponden a un objeto de los muchos que puede haber en la imagen. La segmentación semántica busca clasificar todos los píxeles de una imagen en una categoría, sin distinguir si existen uno o más objetos de esa categoría. La segmentación panóptica es una combinación de las dos anteriores. No solo queremos clasificar todos los píxeles, también queremos los distintos objetos dentro de una misma categoría. La figura 1.4 muestra ejemplos de estas tareas.

El caso que nos ocupa es el de la segmentación semántica de nubes. Tradicionalmente se han empleado diversos métodos para realizar esta tarea, todo ellos basados de un u otro modo en la comparación de colores y/o intensidades.

[2]Existen otras tareas como la *descripción* de una imagen o la generación de *mapas de profundidad*.

(a) Instancias (b) Semántica (c) Panóptica

Figura 1.4: Tipos de segmentación.

El método más sencillo consiste en clasificar los píxeles en base a su intensidad. Kegelmeyer (1994) propuso convertir las imágenes a escala de grises y aplicar los siguientes límites: 0: fuera del campo de visión, 1–99: Cielo despejado, 100–139: Nube fina, 140–200: Nube opaca y 201–255 fuera de escala/sin clasificar.

Un mejor procedimiento consiste en considerar el cociente rojo/azul. Estos métodos se basan en que en el cielo despejado el aire de las atmósfera dispersa principalmente en el espectro azul mientras que el agua y hielo de las nubes dispersan la misma cantidad en los espectros azul y rojo (Chow *et al.*, 2011; Huo *et al.*, 2009). Por tanto, se establece un umbral al cociente rojo/azul, típicamente entre 0.5–0.7, para distinguir entre nubes (por encima del umbral) y cielo despejado (por debajo del umbral) (Calbó y Sabburg, 2008; Kreuter *et al.*, 2009).

Existen variantes más sofisticadas del cociente rojo/azul. Esto sucede porque algunas nubes finas pueden no ser detectadas por no llegar al umbral. En condiciones meteorológicas especialmente adversas como la alta concentración de aerosoles o la presencia de calima, se puede confundir el cielo despejado con nubes (Huo *et al.*, 2009). También, por la dependencia azimutal de la dispersión que hace que el color del cielo despejado varíe con el ángulo de dispersión respecto al sol (Tapakis *et al.*, 2013). Para solucionar estos problemas se han propuesto diversos métodos como usar un umbral adaptativo (Cazorla *et al.*, 2015; Román; Cazorla *et al.*, 2017), usar propiedades de simetría del cielo en comparación con otros instrumentos (Román; Cazorla *et al.*, 2017) o comparar directamente con imágenes de cielos despejados (Ghonima *et al.*, 2012). Pese a las mejoras que estas variantes traen, este tipo de algoritmos basados en umbrales siguen siendo sensibles a las condiciones meteorológicas, las especificaciones y ajustes de la cámara y las condiciones de iluminación (Zhou *et al.*, 2022; Xie *et al.*, 2020).

Para solucionar las carencias de estos métodos basados en umbrales, surgen los métodos basados en redes neuronales convolucionales (CNN, por sus siglas en inglés) que estudiaremos en el capítulo 3. Debido a la escasez de datos, la mayoría de los artículos publicados trabajan con imágenes satélite (Zhou *et al.*, 2022) como Shi *et al.* (2016) o Yuan *et al.* (2017) y son muy pocos autores los que han tratado el uso de CNN con imágenes tomadas desde la superficie terrestre.

Dev; Lee *et al.* (2017) adaptaron el modelo U-Net (Ronneberger *et al.*, 2015) a la segmentación semántica binaria de imágenes todo cielo diurnas. Dev; Nautiyal *et al.* (2019) fueron más allá empleando imágenes de cielo diurnas y nocturnas. Emplearon imágenes

tomadas con cámaras todo cielo pero revirtiendo la distorsión usando la calibración de la cámara antes de procesar las imágenes. Su modelo es similar a U-Net. Xie *et al.* (2020) trabajaron con imágenes todo cielo usando también una arquitectura como la anterior. Zhou *et al.* (2022) tomaron varios modelos de segmentación semántica y los re-entrenanron (trataremos esto en la sección 3.3.2), encontrando el modelo DeepLabV3+ (Chen *et al.*, 2018) como el mejor. Sin embargo, no utilizaron imágenes todo cielo. Liu *et al.* (2022) utilizaron una aproximación más moderna que Dev; Nautiyal *et al.* (2019) y Zhou *et al.* (2022) mediante el uso de *transformers*[3]. Tampoco utilizaron imágenes todo cielo.

De todos los trabajos anteriormente mencionados, únicamente Dev; Nautiyal *et al.* (2019) usaron imágenes todo cielo, esto se debe a que los conjuntos de datos públicos para el entrenamiento de estos modelos son muy limitados. Existen cuatro conjuntos de datos públicos: HYTA (Li *et al.*, 2011), SWIMSEG (Dev; Lee *et al.*, 2017), WSISEG (Fa *et al.*, 2019) y GBCS (Zhou *et al.*, 2022). HYTA, SIWIMSEG y GBCS contienen imágenes de solo parte del cielo y evitan aquellas zonas cercanas al sol o al horizonte. Estos conjuntos de datos tienen 1000, 1013 y 1742 imágenes respectivamente. Sus máscaras son binarias: nube o cielo. WSISEG es el único con imágenes todo cielo, un total de 400. Sus máscaras contemplan 3 categorías: N/A, nube y cielo.

1.3 OBJETIVOS

Como hemos visto, la determinación de algunos parámetros importantes de las nubes, como la cobertura nubosa, se realiza con una baja resolución espacial y temporal. Los métodos basados en umbrales no son suficientemente precisos y, habitualmente, se termina por recurrir a observadores humanos para llevar a cabo estas tareas. Para atajar este problema, este trabajo busca investigar un algoritmo con la finalidad última de automatizar la segmentación semántica de imágenes todo cielo. Para ello, se establecen los siguientes objetivos básicos:

1. Diseñar y entrenar un modelo para la segmentación semántica de imágenes todo cielo en la línea del actual estado del arte.
2. Validar el modelo a partir de un conjunto diferente de imágenes de prueba, segmentadas manualmente.
3. Obtener medidas derivadas del modelo de segmentación: medida de la cobertura nubosa.

Debido al ya suficientemente ambicioso alcance de este trabajo, dejaremos fuera otra posible medida derivada: la oclusión solar. Trabajos futuros pueden partir del modelo de segmentación semántica para trabajar en la triangulación 3D de las nubes de las cámaras o del estudio del movimiento y deformación de las nubes.

[3]La arquitectura de un transformer está fuera del alcance de este trabajo. Puede encontrar más información en Khan *et al.* (2022).

Capítulo 2

INSTRUMENTACIÓN Y CONJUNTO DE DATOS

La construcción del conjunto de datos comporta tres pasos fundamentales: captura, etiquetado y aumento de datos. En la primera sección, veremos qué cámaras se han usado y cómo se procesan las imágenes en bruto del sensor para obtener imágenes finales. Después, veremos las diferentes categorías en las que se clasifican las imágenes y cómo se almacena esta información. Por último, veremos como se generan varias imágenes a partir de una única imagen original mediante la aplicación de distintas transformaciones.

2.1 IMÁGENES DEL CIELO

2.1.1 CÁMARAS TODO CIELO

Las *cámaras de todo cielo* o, simplemente, *cámaras de cielo* son capaces de capturar toda la bóveda celeste. Este tipo de imágenes son usadas con distintos propósitos como la predicción meteorológica, la predicción solar o la medida de la cobertura nubosa (Nie *et al.*, 2024). Más allá de la ciencia atmosférica, también son comunes en astronomía, por ejemplo, para medir la contaminación lumínica (Hänel *et al.*, 2018; Levin *et al.*, 2020). Existe una gran diversidad de cámaras de cielo (Antuña-Sánchez, 2021, p. 31–38). Nosotros hemos utilizado los siguientes modelos:

- SONA202-NF (*Sieltec Canarias S.L.*)
- OMEA-3C (*Alcor System*)
- OMEA-3C-TF (*Alcor System*)

Como se muestra en la figura 2.1, las cámaras están formados por una cúpula protectora, una lente de ojo de pez y un sensor CMOS bajo esta para registrar las imágenes del cielo.

SONA202-NF La cámara SONA202-NF está fabricada por la empresa *Sieltec Canarias S.L.* Cuenta con un sensor CMOS SONY IMX249 que produce imágenes con una resolución de 1172×1158 y una profundidad de color de 10 bits. Está provista de tres filtros de color en forma de mosaico RGGB Bayer y un filtro tribanda, cuyas transmitancias espectrales podemos observar en la figura 2.2. El filtro tribanda adicional reduce la anchura espectral

(a) SONA202-NF

(b) OMEA-3C

Figura 2.1: Cámaras de cielo SONA202-NF y OMEA-3C usadas para la adquisición de datos. Crédito Antuña-Sánchez (2021, p. 39).

de los canales R, G y B, como puede observarse en la figura 2.2; esto es de gran utilidad para obtener colores más puros y para aplicaciones fotométricas con las imágenes del cielo (Antuña-Sánchez *et al.*, 2021; Román; Antuña-Sánchez *et al.*, 2022). Además, esta equipada con una lente ojo de pez que proporciona un ángulo de visión de 185° (Antuña-Sánchez, 2021, p. 39).

OMEA-3C La cámara OMEA-3C está fabricado por la empresa *Alcor System*. El sensor CMOS es el modelo IMX178 de SONY, que cuenta con una resolución de 3096 × 2080 y una profundidad de color de 14-bits. Sobre el sensor hay tres filtros de color (R, G y B) distribuidos en forma de mosaico RGGB Bayer. Además, tiene incorporado un filtro adicional que corta la luz infraroja. Las transmitancias espectrales de estos filtros las podemos observar en la figura 2.2. La lente de ojo de pez de este modelo proporciona un ángulo de visión de 180°. Además, cuenta con sistema de calefacción interno para evitar la condensación interior y evaporar rápidamente gotas de agua o rocío (Antuña-Sánchez, 2021, p. 39–40).

OMEA-3C-TF Denominamos así a la cámara OMEA-3C a la que, además del filtro que corta el infrarrojo, se la ha añadido un filtro tribanda similar al que trae incorporado la cámara SONA202-NF (véase figura 2.2).

2.1.2 UBICACIONES

Las imágenes utilizadas se han obtenido de 6 cámaras de cielo distintas, pertenecientes al GOA-UVa, ubicadas en dos localidades diferentes: la Facultad de Ciencias de Valladolid (41.66ºN, 4.71ºW, 705 m snm; Valladolid, España) y el Observatorio Metereológico *Richard Aßmann* de Lindenberg (52.21ºN, 14.12ºE, 122 m snm; Lindenberg, Alemania). Más información sobre qué modelos están en cada ubicación en la tabla 2.1.

Tabla 2.1: Cámaras empleadas dentro de la red de cámaras del GOA-UVa, ubicación y modelo de cámara disponible. Cada fila corresponde a una cámara distinta, pudiendo haber dos modelos en la misma ubicación.

Ubicación	**Modelo**
Valladolid	SONA202-NF
Valladolid	OMEA-3C
Valladolid	OMEA-3C
Valladolid	OMEA-3C-TF
Lindenberg	OMEA-3C
Lindenberg	OMEA-3C

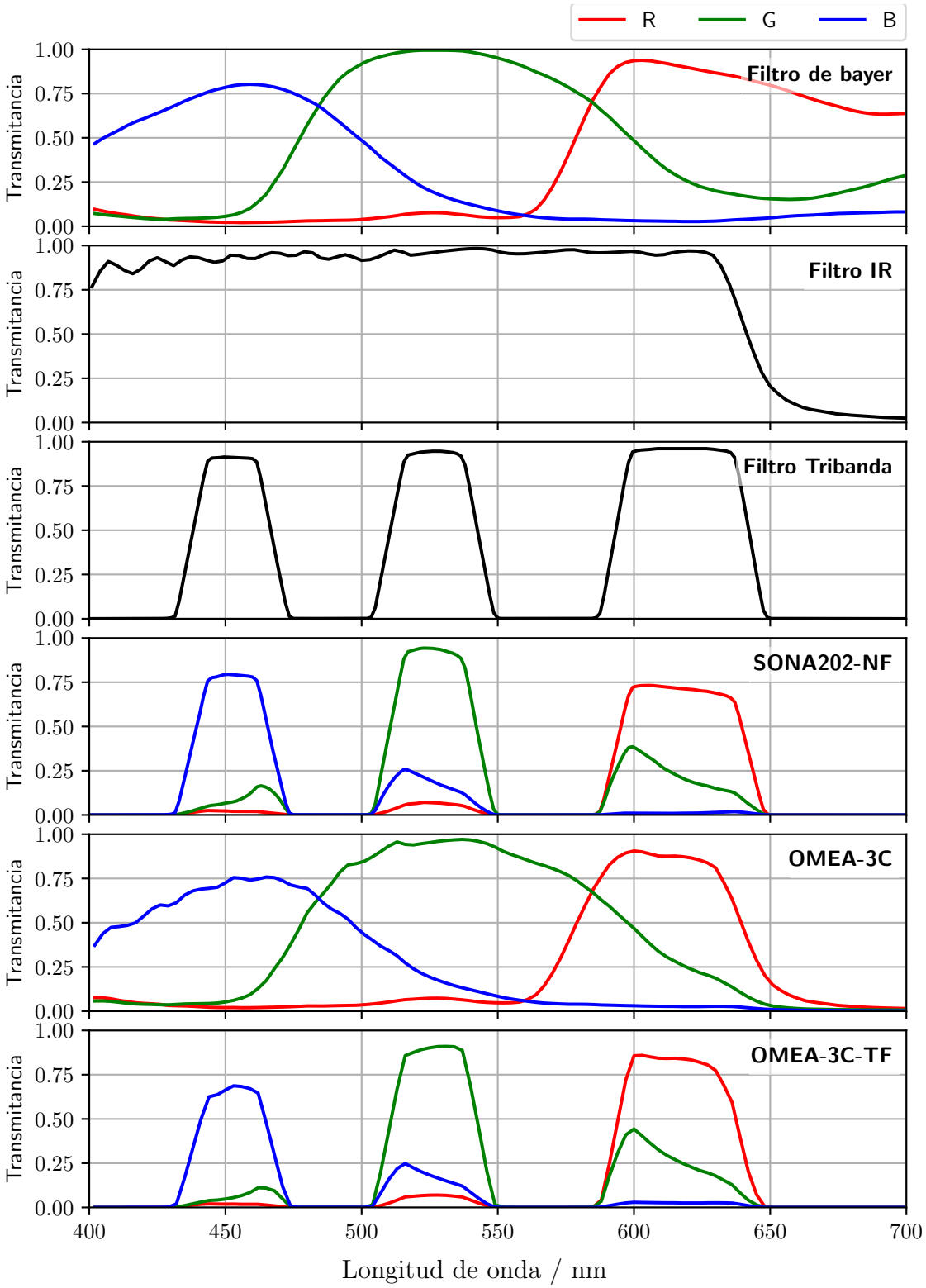


Figura 2.2: Los tres paneles superiores tienen las transmitancias normalizadas de los distintos filtros que tiene cada una de las cámaras. El filtro de bayer representado es el de la OMEA-3C. El filtro de bayer de la SONA202-NF no se representa por ser muy similar y puede consultarse en Antuña-Sánchez (2021, p. 40). Los tres paneles inferiores continen la transmitacia normalizada del conjunto de filtros que emplea esa cámara.

2.1.3 CAPTURA Y POST-PROCESADO

Todas las cámaras de cielo mencionadas se configuraron para capturar imágenes en Alto Rango Dinámico, o HDR por sus siglas en inglés. Esto consiste en tomar una secuencia consecutiva de imágenes, en nuestro caso en formato `raw`[1], bajo diferentes exposiciones[2]. Esta es una técnica común que, por ejemplo, la mayoría de teléfonos móviles emplean hoy en día para poder extraer información en imágenes con alto contraste. Estas secuencias de imágenes HDR se han realizado cada 5 minutos durante los periodos diurnos y cada 2 minutos en los nocturnos. Cabe destacar que, mientras que los tiempos de exposición en las imágenes diurnas van desde décimas de milisegundo a unos pocos milisegundos, haciendo la secuencia de imágenes cuasi-instantánea, los tiempos de exposición llegan alcanzar varias decenas de segundos en los periodos nocturnos, haciendo que la secuencia total se demore algo más de un minuto por las noches.

El proceso de combinación de las imágenes con diferentes exposiciones en una única imagen final se realiza empleando un algoritmo propio del GOA-UVa que consta de los siguientes puntos fundamentales: (Antuña-Sánchez, 2021, p. 49–52)

1. Se normalizan todas las imágenes de la secuencia HDR para que todas tengan la misma exposición aparente y se promedian.

2. Se reduce el contraste de la imagen empleando un mapeo de raíz cuadrada.

3. Se aplica un balance de blancos[3] fijo, el cual solo varía con el modelo de cámara.

4. Se normaliza de nuevo la imagen dividiendo la señal entre el percentil 99.5 %, para que la imagen siempre se vea iluminada.

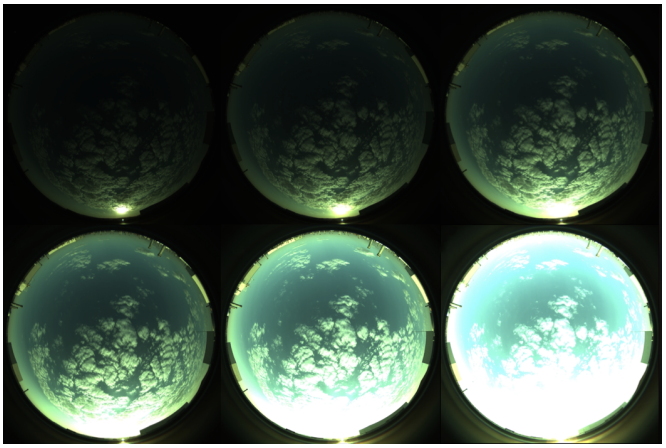
Como resultado obtenemos, para cada secuencia de imágenes, una imagen HDR. Con esta técnica, podemos apreciar perfectamente las partes más brillantes, cuya información se obtiene de las imágenes menos expuestas, y las más oscuras, cuya información se obtiene de las imágenes más expuestas. De esta forma podemos recuperar información de zonas claras y oscuras difiriendo en varios ordenes de magnitud. La figura 2.3 muestra una secuencia de imágenes junto a su imagen final. Las imágenes HDR se procesan automáticamente a partir de las secuencias recibidas de las cámaras[4].

[1] El formato `raw` no es un formato sino una familia de estos. Hablamos de formato `raw` cuando, en lugar de procesar los datos del sensor para generar una imagen final y almacenar la imagen final, se guardan directamente los datos brutos recogidos por el sensor de la cámara. De esta forma, almacenamos directamente una matriz de datos `uint8`, `uint16`,... junto con algunos metadatos de la imagen como la configuración de la captura o el *timestamp*. Los diferentes fabricantes deciden el formato preciso en que se combina esta información, dando lugar a una famila de formatos.

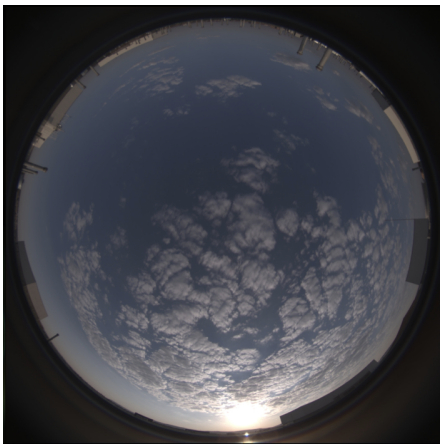
[2] La exposición de una imagen mide la cantidad de luz recogida y la amplificación de esta en el sensor. A más *exposición* más clara será una imagen. No confundir este concepto con el *tiempo de exposición* que es el tiempo durante el cual se expone el sensor de la cámara a la luz incidente. La exposición depende del tiempo de exposición, de la sensibilidad del sensor y de la apertura de la cámara. En las imágenes del GOA-UVa, las diferentes exposiciones se obtienen variando únicamente el tiempo de exposición.

[3] El balance de blancos es una adaptación de los canales de color de la imagen para producir unos colores que el ojo humano perciba como más realistas.

[4] Se pueden consultar las últimas imágenes capturas, así como un vídeo con las imágenes obtenidas durante el último día, en la web del GOA-UVa: goa.uva.es.



(a) Diferentes exposiciones



(b) HDR resultante

Figura 2.3: Proceso de obtención de una imagen HDR. (a) son las seis imágenes originales capturadas con diferentes exposiciones. (b) es la imagen obtenida, a partir de las otras seis, tras aplicar balance de blancos. Crédito: Antuña-Sánchez (2021).

2.2 ETIQUETADO MANUAL DE LAS IMÁGENES

Las imágenes se van a utilizar principalmente para identificar nubes. Sin embargo, como vimos en la figura 1.2, en una imagen del cielo de este tipo suelen aparecer otros elementos, por lo que se han definido cinco categorías distintas a las que puede corresponder de forma excluyente cada píxel:

- N/A (no cielo)
- Nube fina
- Sol
- Cielo despejado
- Nube espesa

Esta clasificación es habitual en la literatura (Kegelmeyer, 1994; Ghonima *et al.*, 2012). Una vez definidas las categorías que puede tener cada píxel, se selecciona qué píxeles corresponden a cada categoría. Esta clasificación se ha realizado por miembros del GOA-UVa. Como resultado, para cada imagen se tiene una máscara con el mismo ancho y alto que la imagen original. Las mascaras, al igual que las imágenes, se almacenan en formato `jpg`. La figura 2.4 muestra algunos ejemplos de máscaras.

Este etiquetado tiene dos dificultades principales: determinar el borde de una nube y determinar si considerarla como nube fina o espesa. Ninguna de estas dos preguntas tiene una respuesta clara, inequívoca ni consensuada. Por ello, se crean carencias en la elaboración del conjunto de datos que, como veremos más adelante, pueden afectar a los resultados del modelo.

Como veremos en el capítulo 3, trabajaremos con estas imágenes pero reduciendo su resolución a 224×224. Un aspecto muy importante es que, cuando tengamos que reducir el tamaño de las imágenes deberemos aplicar algún algoritmo de interpolación. Es de vital importancia para las mascaras, emplear un algoritmo de tipo *vecino más cercano*[5] (*nearest neighbor*). Este algoritmo no interpola al uso, sino que toma el valor del píxel más cercano.

[5] **¡Atención!** Algunas de las principales librerías para trabajar con imágenes, como `OpenCV` (Bradski, 2000) o `TensorFlow` (Abadi *et al.*, 2016) implementan la interpolación lineal como valor por defecto.

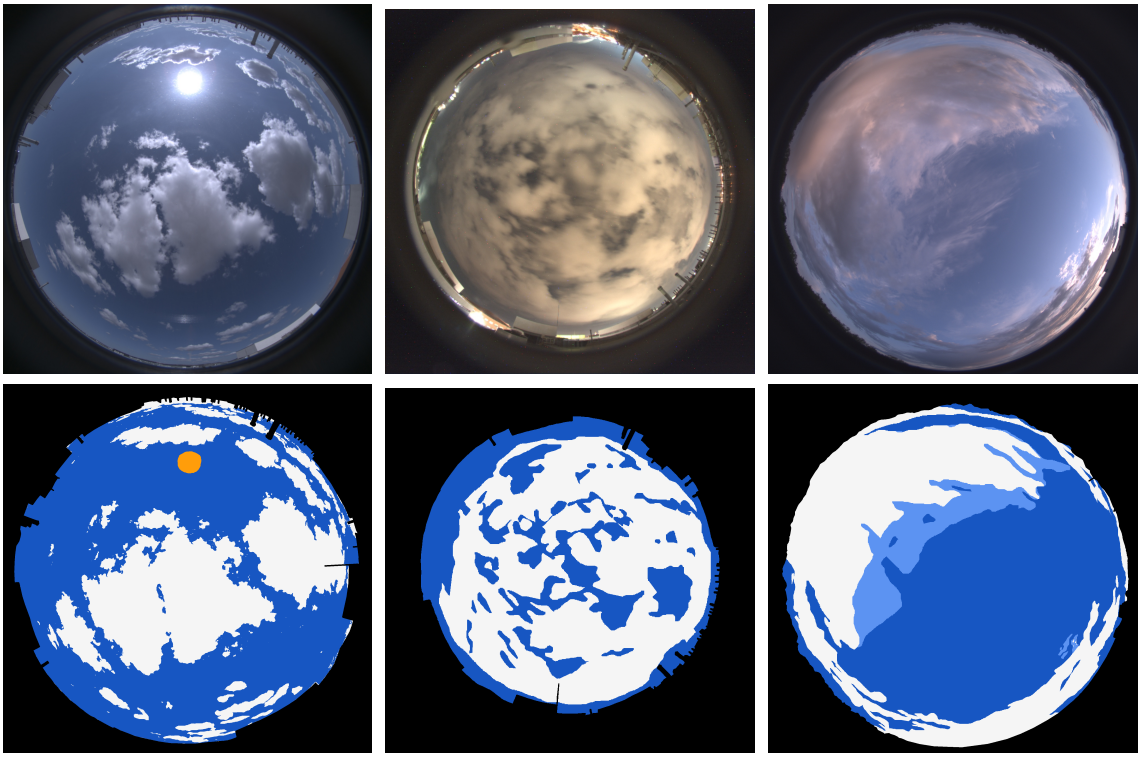


Figura 2.4: Ejemplos de imágenes de cielo (superior) con sus respectivas máscaras etiquetadas manualmente (inferior). Las etiquetas se han recodificado a distintos colores: N/A (negro), cielo despejado (azul), sol (naranja), nube espesa (blanco) y nube fina (gris azulado).

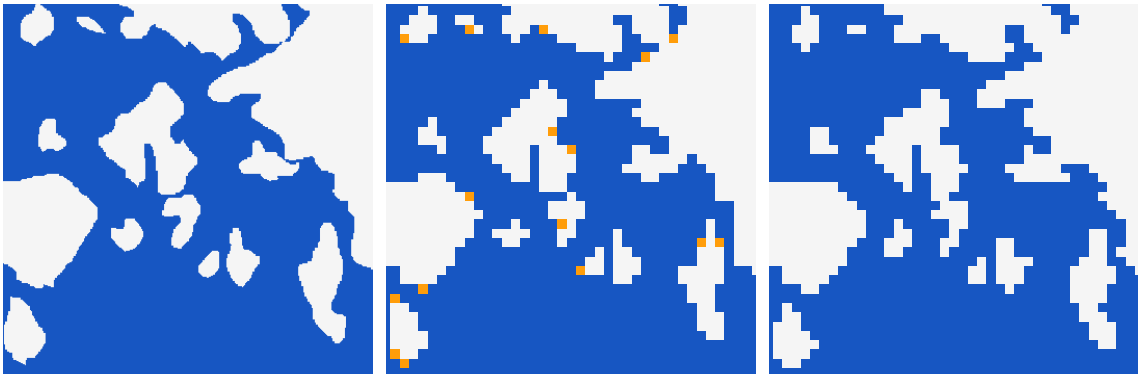
Si aplicamos otro algoritmo, como la interpolación lineal, es posible que obtengamos píxeles cuyo valor no coincide con la etiqueta original. La figura 2.5b muestra como la interpolación lineal hace aparecer valores de sol (almacenados como 2) entre el cielo (almacenado como 1) y la nube espesa (almacenada como 3). En la figura 2.5c, con interpolación de vecino más cercano, no aparecen estos artefactos.

2.3 PARTICIONADO Y AUMENTACIÓN DEL CONJUNTO DE IMÁGENES

El conjunto de imágenes está formado por un total de 346 imágenes, de las cuales, solo un $\sim 10\,\%$ son nocturnas. Las imágenes están tomadas entre los años 2021 y 2023. Existe variedad de condiciones atmosféricas: completamente cubierto, parcialmente cubierto, presencia de calima, niebla... aunque no en la misma proporción. La figura 2.6 muestra el número de imágenes en función de la cobertura nubosa expresada en octas (octavos de cielo cubiertos). Tampoco tenemos la misma cantidad de píxeles de todas las etiquetas, por ejemplo, hay muchos menos píxeles de sol y de nube fina que del resto. La figura 2.7 muestra esta distribución.

El conjunto de datos se ha dividido aleatoriamente en tres subconjuntos más pequeños: entrenamiento, validación y rendimiento[6]. El conjunto de entrenamiento consta de 254

[6] A este conjunto es habitual referirse en la literatura como *conjunto de prueba*. La nomenclatura es confusa, pues el *conjunto de validación* se usa junto con el de entrenamiento y el *conjunto de rendimiento*



(a) Mascara tamaño original (b) Interpolación lineal (c) Interpolación vecino más cercano

Figura 2.5: Comparación de dos métodos de interpolación para reducir el tamaño de la máscara. (b) aparecen artefactos en la imágen (como puntos naranjas de sol entre las nubes). (c) la mascara aparece sin artefactos.

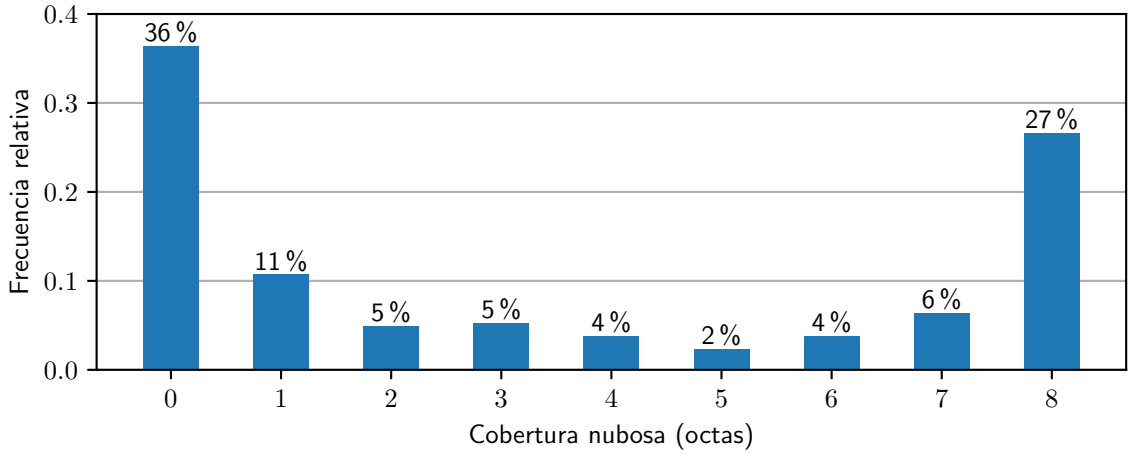


Figura 2.6: Imágenes de todo el conjunto de datos en función de la cobertura de cielo calculada según (4.1).

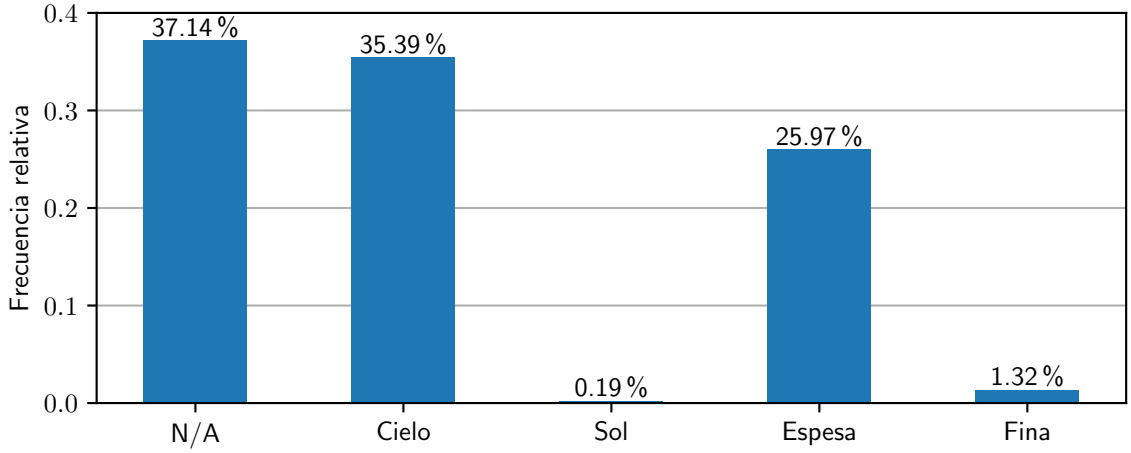


Figura 2.7: Distribución de los píxeles por categoría de todas las imágenes.

imágenes. El conjunto de validación consta de 46 imágenes y se emplea para monitorizar el entrenamiento, como veremos en la sección 3.3.2. El conjunto de rendimiento consta de 46 imágenes para calcular métricas sobre el desempeño del modelo. Estos subconjutos tienen una proporción aproximada de 74 – 13 – 13.

Contamos con muy pocas imágenes[7], tan solo 346. Por ello, si queremos incrementar la cantidad de imágenes necesitamos generar imágenes inéditas a partir de las originales. Esto se consigue mediante la aplicación a las imágenes originales de una o varias transformaciones digitales (*Data Augmentation*). Nosotros hemos hecho uso de la librería de código abierto `Albumentations` (Buslaev *et al.*, 2020) que contiene infinidad de transformaciones. En concreto, hemos usado:

1. Deformación
2. Rotación
3. Desplazamiento RGB
4. Ajuste de brillo y contraste

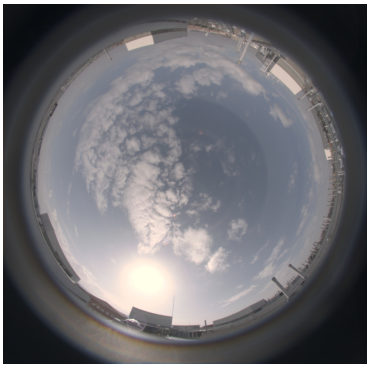
Cada vez que se genera una imagen con este procedimiento, esta se rota siempre. Las demás transformaciones se aplican con una probabilidad del 50 %. Una o más transformaciones pueden ser aplicadas a una misma imagen. La transformaciones no son conmutativas y se aplican siempre en el orden expuesto. La figura 2.8 muestra un ejemplo de cada una de estas transformaciones. Como se puede comprobar, las transformaciones son sutiles para que no se pierda la esencia de la imagen original. Aplicando esta técnica se han obtenido 15 nuevas imágenes por cada imagen original. También se hacen pruebas recortando las imágenes para eliminar el exterior y con transformaciones elásticas[8] pero se ha decidido no incluir estas transformaciones finalmente.

Únicamente se aumentan los conjuntos de entrenamiento y de validación, dejando intacto el de rendimiento. Además, no se mezclan las imágenes originales. Es decir, una imagen que se generó a partir de otra que originalmente estaba en el conjunto de entrenamiento, pertenecerá al conjunto de entrenamiento y no al de validación. Después de este proceso, obtenemos un conjunto de entrenamiento aumentado con 5190 imágenes, un conjunto de validación aumentado con 690 imágenes y un conjunto de rendimiento de 46 imágenes. A los conjuntos aumentados nos referiremos indistintamente que a los conjuntos originales.

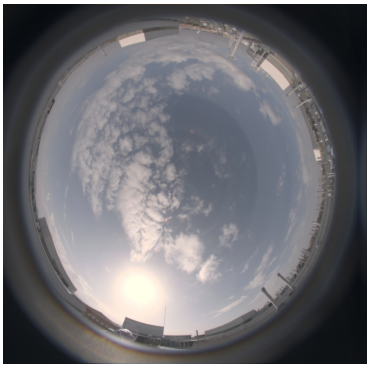
se usa para validar el modelo.

[7] El tamaño habitual de los grandes conjuntos de datos para visión artificial es del orden 10^3–10^5 imágenes. A modo ilustrativo, Zhou *et al.* (2022) parten de 1742 imágenes.

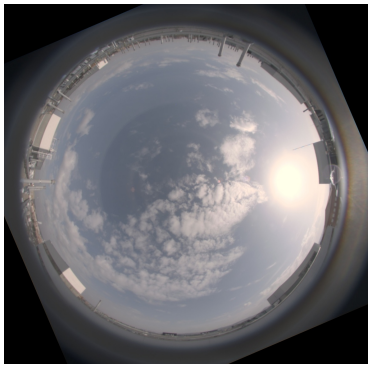
[8] Las transformaciones distorsionan la imagen haciendo que parezca vista a través de una superficie de agua.



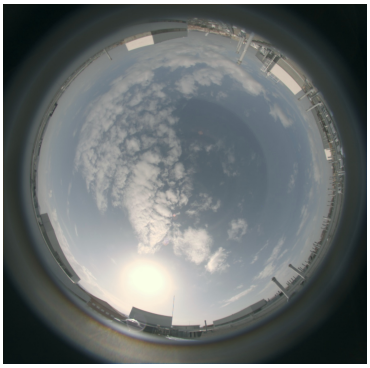
(a) Original



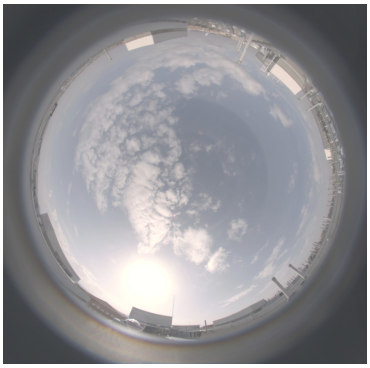
(b) Distorsión



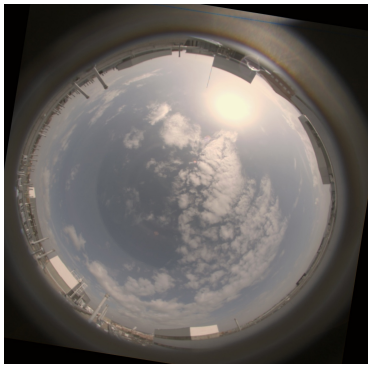
(c) Rotación



(d) RGBshift



(e) Brillo y contraste



(f) Todas

Figura 2.8: Transformaciones aplicadas a las imágenes para el aumento de datos. (b)-(e) muestran las transformaciones individuales. (f) muestra la aplicación de todas las transformaciones sobre una misma imagen.

Capítulo 3

MODELO DE VISIÓN ARTIFICIAL

Proponemos un modelo propio para la segmentación semántica multi-clase de las imágenes del cielo. El modelo, es un modelo estadístico basado en redes neuronales convolucionales. Como el lector puede no estar familiarizado con estas, trataremos brevemente en qué consisten estos algoritmos para después centrarnos en la operación de convolución y sus variantes, de las que haremos uso en el modelo. Después, veremos los detalles del modelo y su entrenamiento.

3.1 REDES NEURONALES

Las redes neuronales son una familia de funciones $h_{G,\boldsymbol{\theta},\mathscr{F}} : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ construidas mediante la concatenación, fundamentalmente, de funciones elementales $g_{\mathbf{w},f} : \mathbb{R}^d \longrightarrow \mathbb{R}$ llamadas *neuronas*. La concatenación se realiza de acuerdo a un grafo G dirigido, conexo y acíclico. Las neuronas $g_{\mathbf{w},f}$ dependen de una serie de parámetros $\mathbf{w}_j$ que constituyen el vector de parámetros $\boldsymbol{\theta} = (\mathbf{w}_1, \ldots, \mathbf{w}_r)$ del que depende $h_{G,\boldsymbol{\theta},\mathscr{F}}$ y de una serie de funciones $f : \mathbb{R} \longrightarrow \mathbb{R}$ que constituyen $\mathscr{F}$. Estas funciones se llaman *funciones de activación*.

La figura 3.1 muestra un ejemplo del grafo G de una red neuronal. No todas las conexiones tienen por qué agruparse en capas ni todas las neuronas entre capas están necesariamente conectadas.

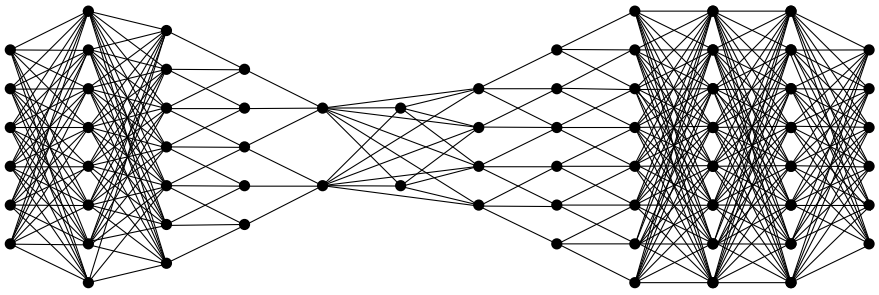


Figura 3.1: Ejemplo del grafo asociado a una red neuronal. Cada nodo representa una neurona, las aristas son conexiones de izquierda a derecha.

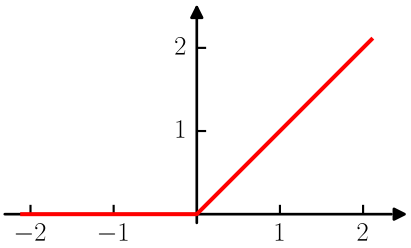
Las neuronas $g_{\mathbf{w},f}$ se obtienen realizando la suma ponderadas por $\mathbf{w} = (w_1, \ldots, w_{d_j})$ de las entradas y aplicando después la función de activación f,

$$g_{\mathbf{w},f}(\mathbf{x}) = f\left(\sum_{i=1}^{d_j} x_i w_i\right).$$

Algunos ejemplos de funciones de activación comunes son:

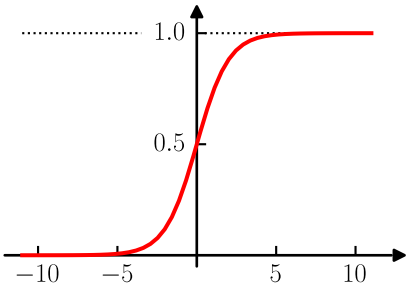
1. ReLU (Rectified Linear Unit)

$$f(x) = \begin{cases} 0 & \text{si } x \leq 0 \\ x & \text{si } x > 0 \end{cases}$$



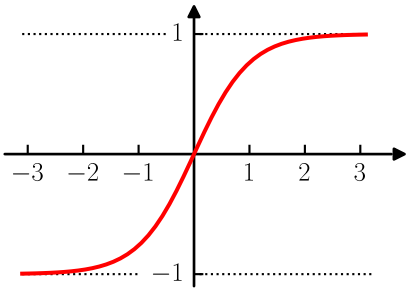
2. Logística o Sigmoide

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



3. Tangente hiperbólica

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



Además, es importante la función *Softmax* $\boldsymbol{\sigma} : \mathbb{R}^{d_j} \longrightarrow [0,1]^{d_j}$

$$\boldsymbol{\sigma}_i(x_1, \ldots, x_{d_j}) = \frac{e^{x_i}}{\sum_{k=1}^{d_j} e^{x_k}}$$

aunque no entra exactamente en este esquema. Nótese que la suma de todas las componetes de $\boldsymbol{\sigma}$ suma exactamente 1:

$$\sum_{i=1}^{d_j} \boldsymbol{\sigma}_i(\mathbf{x}) = \frac{\sum_{i=1}^{d_j} e^{x_i}}{\sum_{k=1}^{d_j} e^{x_k}} = 1.$$

Por tanto, la salida de una función softmax es una distribución de masa de probabilidad sobre cada una de sus componentes.

3.1.1 REDES NEURONALES CONVOLUCIONALES

Las redes neuronales convolucionales (CNN, por sus siglas en inglés) son aquellas que incluyen al menos una capa convolucional (normalmente muchas). Junto a la convolución, son necesarias otras dos operaciones fundamentales: el *MaxPooling* y la convolución transpuesta, que trataremos en sucesivas secciones. En nuestro caso, nos centraremos en las capas de dos dimensiones, pues estamos tratando con imágenes.

Convolución

Una convolución 2D es una operación que nos permite transformar una matriz $A = (a_{ij})$ en otra matriz $B = (b_{ij})$ con diferentes dimensiones (que pueden ser mayores o menores) mediante la aplicación de un filtro. El filtro es una matriz $K = (k_{ij})$ más pequeña que deslizamos sobre la entrada para conformar la salida. La figura 3.2 muestra una convolución que parte de una matriz 4×4 (azul inferior) con un filtro de tamaño 3×3 (sombreado inferior) para producir una matriz de salida 2×2 (verde superior).

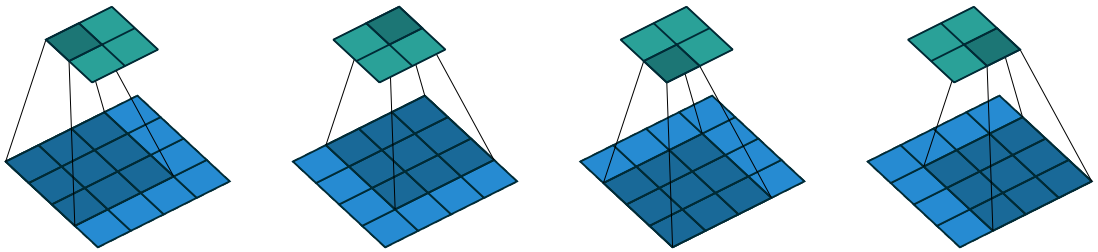


Figura 3.2: Ejemplo esquemático de una convolución 2D que transforma una matriz 4×4 (azul inferior) en una matriz 2×2 (verde superior) con un filtro de tamaño 3×3 (sombreado inferior). El filtro se desliza por toda la matriz de entrada para generar la salida. Crédito Dumoulin *et al.* (2016).

El filtro se aplica haciendo la suma ponderada por la matriz del filtro K del menor de la matriz de entrada A cubierta por el filtro durante cada movimiento. Siguiendo con el ejemplo de la figura 3.2:

$$B = \begin{pmatrix} \sum_{i=1}^{3}\sum_{j=1}^{3} k_{ij}a_{ij} & \sum_{i=1}^{3}\sum_{j=1}^{3} k_{ij}a_{i,j+1} \\ \sum_{i=1}^{3}\sum_{j=1}^{3} k_{ij}a_{i+1,j} & \sum_{i=1}^{3}\sum_{j=1}^{3} k_{ij}a_{i+1,j+1} \end{pmatrix}.$$

Los valores del filtro, serán los parámetros de nuestro modelo que deberemos ajustar durante el proceso de entrenamiento.

En el ejemplo de la figura 3.2, el tamaño de la matriz de salida se ha reducido. Esto puede ser indeseado en algunas ocasiones. Para conseguir mantener la dimensión, podemos ampliar la matriz de entrada introduciendo ceros. Estos ceros se conocen como *relleno* (del inglés *padding*). La figura 3.3 muestra un ejemplo de este tipo de convolución.

Como veremos más adelante, no trabajaremos con una única matriz de entrada A que queremos convertir en una única matriz de salida B, sino que tendremos muchas matrices de entrada con las mismas dimensiones $A_1, \ldots, A_n$ que queremos convertir en muchas matrices de salida $B_1, \ldots, B_m$[1] con las mismas dimensiones, usando varios filtros $K_1^{(1)}, \ldots, K_n^{(1)}, \ldots\ldots, K_1^{(m)}, \ldots, K_n^{(m)}$. Para cada una de estas matrices de salida $B^{(j)}$,

[1] Podemos pensar en estos como si, en lugar de las habituales imágenes RGB de 3 canales, tuviésemos imágenes de n y m canales respectivamente.

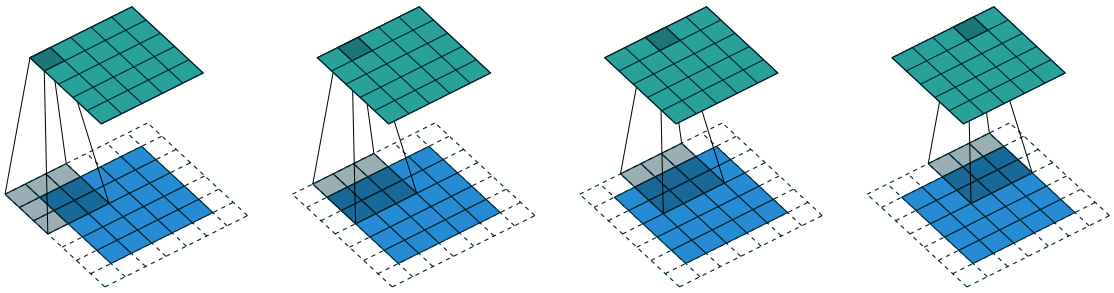


Figura 3.3: Ejemplo esquemático de una convolución 2D con relleno tipo *igual*, que transforma una matriz 5×5 (azul inferior) en una matriz 5×5 (verde superior) con un filtro de tamaño 3×3 (sombreado inferior). Las casillas blancas representan ceros. El filtro se desliza por toda la matriz de entrada para generar la salida. Crédito Dumoulin *et al.* (2016).

deslizaremos los filtros $K_1^{(j)}, \ldots, K_n^{(j)}$ sobre las entradas $A_1, \ldots, A_n$ y sumaremos los resultados. La figura 3.4 describe esquemáticamente esta operación. Esto es a lo que nos referiremos cuando hablemos de una *capa convolucional* con m filtros.

Existen otra dos nociones importantes sobre las capas convolucionales: el *paso* (del inglés *stride*) y las dilataciones. Se omiten por no ser necesarios en este trabajo[2].

MaxPooling

Junto a las capas convolucionales, es habitual la presencia de otra operación fundamental con el objetivo de reducir el tamaño de las imágenes: el *MaxPooling*.

Esta es una operación similar a la convolución, que transforma una matriz de entrada A en una matriz de salida B. Al igual que en la convolución, tendremos una ventana que se desliza por la matriz de entrada. En lugar de realizar una suma ponderada por los valores del filtro, tomaremos el máximo de todos ellos. La figura 3.5 muestra un ejemplo de esta operación.

Convolución transpuesta

Si con la operación *MaxPooling* se reduce el tamaño de la matriz de entrada A, con la *convolución transpuesta* queremos aumentar este tamaño. Para ello, realizaremos *convoluciones dilatadas*. Una convolución dilatada es similar a una convolución con relleno, en lugar de extender con ceros la matriz de entrada A, lo que haremos será intercalar estos ceros entre los valores de A. La figura 3.6 muestra un ejemplo de la convolución transpuesta.

A diferencia de las convoluciones, esta operación la aplicaremos para cada matriz por separado. Por tanto, si partimos de n matrices de entrada $A_1, \ldots, A_n$ obtendremos n matrices de salida $B_1, \ldots, B_n$ respectivamente.

Es habitual hablar de *paso* (del inglés *stride*) y *relleno* en este contexto para calcular el número de ceros añadidos y sus posiciones. Nosotros no usaremos esta nomenclatura[3] por ser innecesaria para lo comprensión de la operación.

[2] Puede encontrar información sobre ellos en Dumoulin *et al.* (2016).
[3] Puede encontrar más información sobre esta nomenclatura en Dumoulin *et al.* (2016).

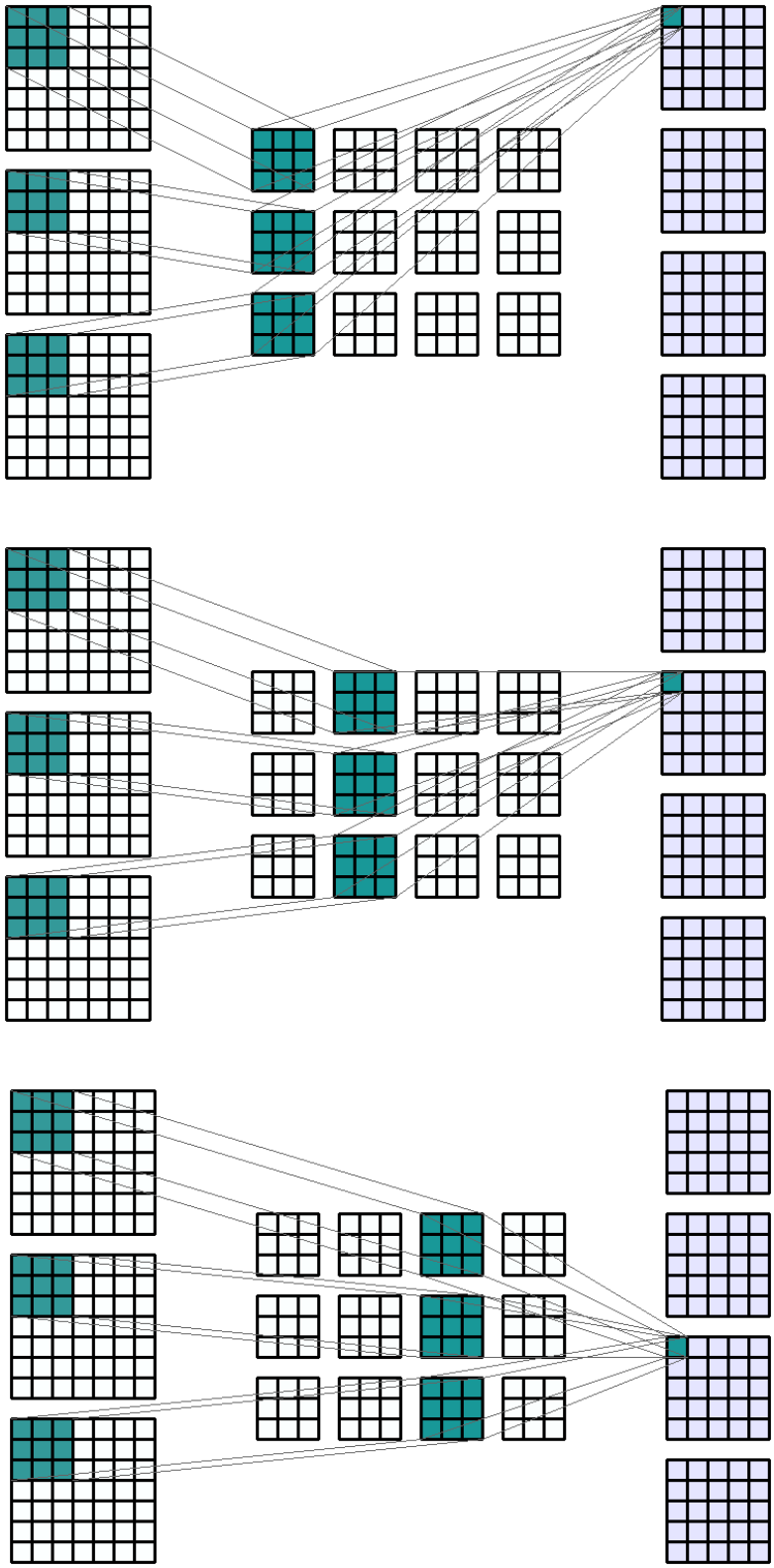


Figura 3.4: Capa convolucional 2D aplicada a una entrada $7 \times 7 \times 3$ (izquierda) para dar una salida $5 \times 5 \times 4$ (derecha) usando filtros de tamaño $3 \times 3 \times 3$ (centro). Crédito Thevenot (2020).

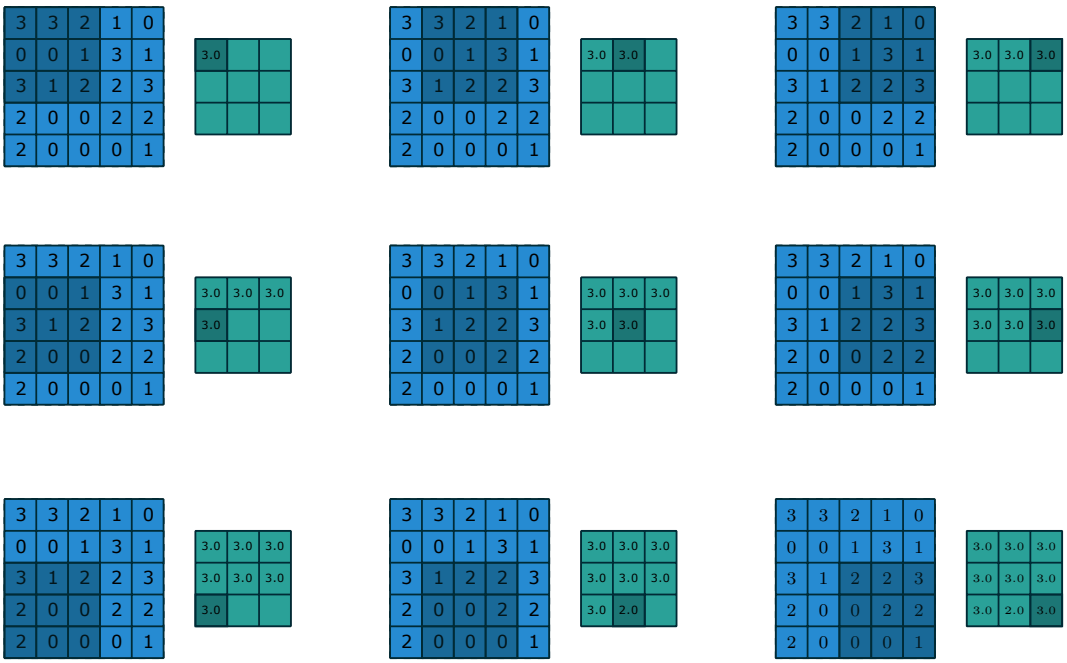


Figura 3.5: Ejemplo esquemático de *MaxPooling* que transforma una matriz 5×5 (azul inquierda) en una matriz 3×3 (verde derecha) con una ventana de tamaño 3×3 (sombreado inquierda). El filtro se desliza por toda la matriz de entrada para generar la salida. Crédito Dumoulin *et al.* (2016).

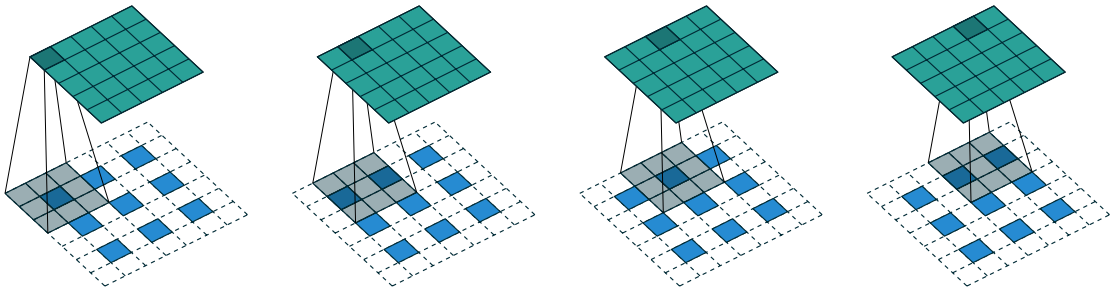


Figura 3.6: Ejemplo de convolución transpuesta. La matriz de entrada tiene dimensiones 3×3 (azul inferior), el filtro 3×3 (sombreado inferior) y la salida 5×5 (verde superior). Las casillas blancas representan ceros. El filtro se desliza por toda la entrada para generar la salida. Crédito Dumoulin *et al.* (2016).

3.2 ARQUITECTURA DEL MODELO

Muchas redes destinadas a la segmentación semántica adoptan una arquitectura de tipo *enconder–decoder*. Esto significa que la red esta compuesta de dos partes fundamentales: una primera parte capaz de analizar la imagen y entender su estructura y atributos más importantes, y una segunda parte capaz de interpretar estos atributos y generar una máscara para la imagen original. La máscara predice una categoría para cada píxel de la imagen original.

En el modelo, de elaboración propia, seguimos el siguiente esquema. Empleamos la red VGG16[4] (Simonyan *et al.*, 2014) como encoder, que consta de 5 bloques formados por dos o tres capas convolucionales y una capa de MaxPooling. Como decoder, se emplean 4 bloques formados por una convolución transpuesta seguida de 2 convoluciones. Además, la salida de cada bloque del encoder, se concatena con la salida de cada convolución transpuesta. La figura 3.7 presenta un esquema de la red[5].

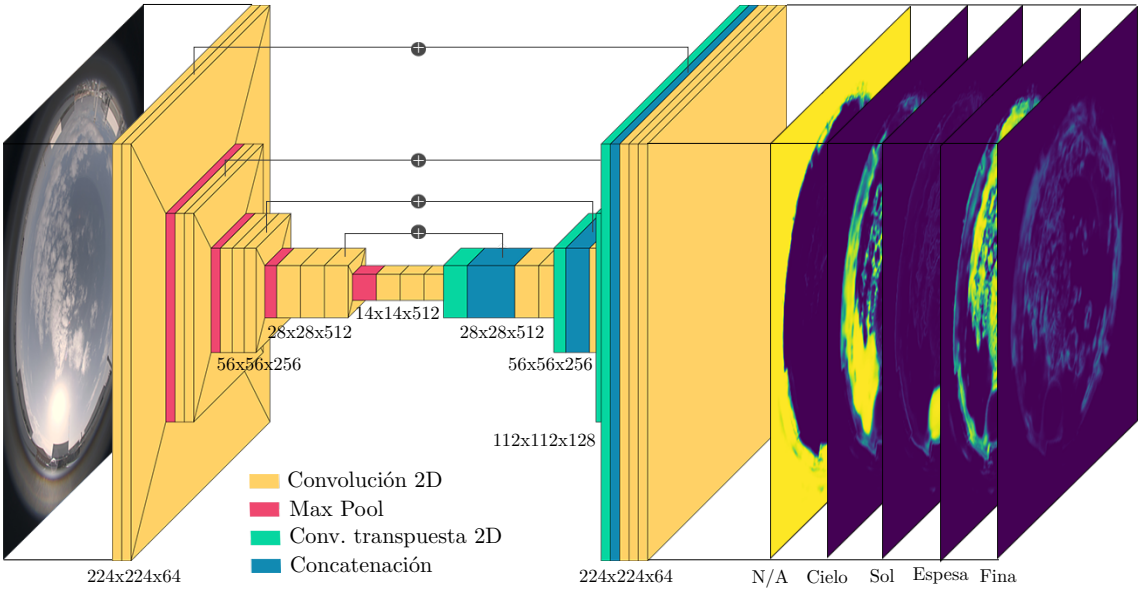


Figura 3.7: Representación esquemática de las capas usadas en el modelo. Las dimensiones representadas son aquellas a la salida de la capa. Todas las capas emplean la función de activación ReLU, a excepción de la última que usa Softmax. El modelo se alimenta con la imagen todo cielo (izquierda) y genera 5 mapas de probabilidad, uno para cada categoría (derecha).

La capa de entrada es una imagen $224 \times 224 \times 3$, es decir, una imagen RGB a la que hemos reducido su tamaño original. Todas las capas, a excepción de la última, emplean la función de activación ReLU debido a su bajo coste computacional. En la última capa, empleamos la función *Softmax*. Se toman 5 canales de salida para esta función, de manera

[4]La red VGG (Simonyan *et al.*, 2014) es una red neuronal orientada a la clasificación de imágenes compuesta por 16 capas convolucionales y 4 capas *MaxPooling* intercaladas. Es una red muy conocida en el campo de la visión artificial por demostrar que el aumento del número de capas de la red podía aumentar drásticamente el rendimiento de los modelos.

[5]Además, se emplean capas de *Batch Normalization* cuya explicación se omite por no considerarse fundamental para la comprensión de este trabajo. Estas capas están relacionadas con el proceso de entrenamiento y su naturaleza estadística. Puede encontrar más detalles sobre ellas en Goodfellow *et al.* (2016, p. 317–321).

que cada canal corresponda a cada una de las 5 etiquetas. Como la salida esta normalizada, podemos tomar los valores como la *probabilidad* o *confianza* que el modelo tiene sobre si el píxel pertenecerá a una categoría concreta. De esta manera, la salida de la red tiene dimensiones $224 \times 224 \times 5$. La predicción final, se hace tomando como categoría aquella que tiene confianza máxima para cada píxel como ilustra la figura 3.8. Esta elección se justifica en la sección 3.3.1.

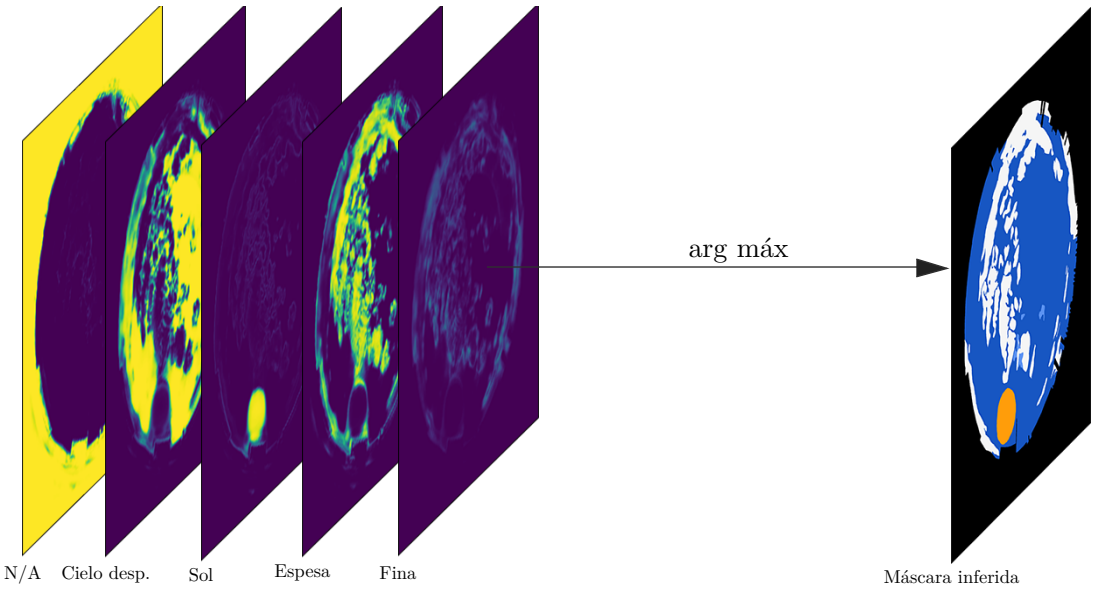


Figura 3.8: Obtención de la máscara inferida a partir de los mapas de probabilidad para cada categoría. Para cada píxel, la categoría se obtiene como aquella con máxima probabilidad.

Teniendo en cuenta todas las capas, el modelo tiene un número total de 26M parámetros independientes, de los cuales 11M pertenecen al encoder y 15M al decoder. Este modelo está inspirado en el artículo de Dev; Nautiyal *et al.* (2019), que presenta una red neuronal convolucional para la segmentación semántica binaria de imágenes de cielo y nubes.

3.3 ENTRENAMIENTO DEL MODELO

Recordemos que la red neuronal usada es una función $h_{G,\boldsymbol{\theta},\mathscr{F}}$ parametrizada por el grafo G que hemos fijado en la figura 3.7, las funciones de activación ReLU y Softmax, y los parámetros $\boldsymbol{\theta}$. Deberemos encontrar el conjunto de parámetros $\boldsymbol{\theta}$ que sea capaz de clasificar adecuadamente las imágenes. El problema de encontrar los parámetros correctos se trata como un problema de optimización y no difiere en su esencia del problema de encontrar una recta que mejor ajuste un conjunto de puntos.

Para resolver este problema, debemos medir el error cometido por la red en la predicción de las etiquetas de cada píxel. Vamos a definir en la sección 3.3.1 una función de pérdida $\ell(Y, \hat{Y})$ que dependa de la etiqueta de referencia Y y de la etiqueta predicha $\hat{Y}$ que de cuenta del error cometido por la red. Por ejemplo, en el problema de regresión por mínimos cuadrados, la función de pérdida es $\ell(\mathbf{y}, \hat{\mathbf{y}}) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$.

Ahora, tomando n imágenes de nuestro conjunto de entrenamiento

$S = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, podemos evaluar el error cometido a través del *riesgo empírico* $\mathcal{R} = \mathcal{R}(\boldsymbol{\theta})$, que se define como:

$$\mathcal{R}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \ell\big(Y_i, h_{G,\boldsymbol{\theta},\mathscr{F}}(X_i)\big). \tag{3.1}$$

Es decir, $\mathcal{R}(\boldsymbol{\theta})$ es el error que en promedio comete $h_{G,\boldsymbol{\theta},\mathscr{F}}$ en predecir las etiquetas. El problema de entrenamiento consiste encontrar $\hat{\boldsymbol{\theta}} = \arg\min \mathcal{R}(\boldsymbol{\theta})$ que hace mínimo $\mathcal{R}$. La red neuronal que usaremos será entonces $h_{G,\hat{\boldsymbol{\theta}},\mathscr{F}}$.

3.3.1 FUNCIÓN DE PÉRDIDA

Recordemos que el modelo predice, para cada píxel (i, j), la confianza $p_{ij}^{(c)}$ que tiene de que este píxel pertenezca a una cada categoría $c \in \mathscr{C}$. Se verifica que $\sum_{c\in\mathscr{C}} p_{ij}^{(c)} = 1$ y $p_{ij}^{(c)} \in (0, 1]$ para todos i, j, c.

Como función de pérdida, vamos a usar la función *entropía cruzada multicategoría* con pesos, que se define como:

$$\ell = \sum_{c\in\mathscr{C}} \left(-\lambda_c \sum_{i=1}^{W} \sum_{j=1}^{H} \mathbb{1}_{ij}^{(c)} \log\left(p_{ij}^{(c)}\right) \right), \tag{3.2}$$

donde $\mathbb{1}_{ij}^{(c)}$ es 1 si el píxel (i, j) pertenece a la categoría c-ésima y 0 en el resto de casos, y $\lambda_c > 0$ es un peso que se asigna a cada categoría. Nótese que, como función $p_{ij}^{(c)}$, ℓ es una función diferenciable, no acotada superiormente y su mínimo absoluto se alcanza y vale 0.

Esta función solo penaliza los $p_{ij}^{(c)}$ para los que el píxel (i, j) pertenece a la categoría c correcta. No dice nada sobre el resto de confianzas predichas para las categorías incorrectas. Además, ℓ es monótona decreciente con el valor de $p_{ij}^{(c)}$. Al minimizar esta función, si la etiqueta correcta del píxel (i, j) es C, estamos imponiendo que $p_{ij}^{(C)}$ sea cercano a 1. Como las confianzas son una distribución de masa de probabilidad, también estamos imponiendo que $p_{ij}^{(c)}$ sea cercano a 0 para los $c \neq C$. De esta forma, justificamos tomar como predicción c_{ij} la categoría

$$c_{ij} = \arg\max_{c\in\mathscr{C}} p_{ij}^{(c)}.$$

3.3.2 PROCESO DE OPTIMIZACIÓN

Para encontrar $\arg\min \mathcal{R}(\boldsymbol{\theta})$ no existe un procedimiento general. En algunos casos, como el de la regresión lineal, el problema tiene una solución analítica sencilla. En el caso de las redes neuronales, no existe solución analítica. Debemos recurrir a un método numérico que permita aproximar la solución. El método numérico empleado se conoce en la jerga de la inteligencia artificial como *optimizador*.

Los métodos empleados se basan de una u otro forma en el descenso del gradiente[6]. Este puede calcularse de manera eficiente mediante la aplicación reiterada de la regla de

[6] Quizá deberíamos hablar de métodos de descenso del *subgradiente*. Nótese que la red neuronal no es

la cadena[7,8]. En concreto, nosotros usaremos el algoritmo `Adam` (Kingma *et al.*, 2014). El parámetro principal[9] de ese algoritmo es la *tasa de aprendizaje* γ que representa cuanto nos movemos en la dirección del gradiente en cada iteración.

En cada iteración, no se pueden usar todas las imágenes de entranamiento para calcular el gradiente de (3.1) debido al privativo coste computacional de calcular derivadas con respecto de millones de parámetros para miles de imágenes distintas. Por ello, en cada iteración solo se procesa un pequeño *lote* de imágenes (del inglés *batch*). En nuestro caso, debido a los límitados recursos computacionales de los que disponemos, usamos 4 imágenes por lote. Estos lotes se conforman tomando imágenes de manera aleatoria del conjunto de datos. Una vez hayamos dado suficientes pasos como para haber iterado con todas las imágenes del conjunto de datos diremos que hemos completado una *época* de entrenamiento.

Para reducir el coste computacional, no se entrenan los 11M de parámetros del encoder; en su lugar, se toman parámetros preentrenados para resolver un problema de clasificación multiclase con el conjunto de datos públicos ImageNet (Russakovsky *et al.*, 2015). Este conjunto de datos contiene más de 1.2M de imágenes cotidianas con 1 000 categorías generales como perro, gato, coche,... Los parámetros preentrenados los facilita la librería `TensorFlow` (Abadi *et al.*, 2016). Este es el motivo último de usar VGG16 como encoder.

Emplear parte de una red entrenada con un propósito, en otra red entrenada con otro propósito se conoce como *transferencia de aprendizaje*. Diversos autores como Redmon *et al.* (2016) aseguran que esta aproximación puede mejorar los resultados obtenidos.

Ejecución del entrenamiento

La definición y entrenamiento del modelo se realizan usando el lenguaje de programación `Python 3.10` y empleando la librería de Inteligencia Artificial de propósito general `TensorFlow 2.15`[10] (Abadi *et al.*, 2016). Como hardware se emplea un ordenador portátil `Lenovo IdeaPad 330` equipado con un procesador `Intel Core i7-8750H` y una tarjeta gráfica `NVIDIA 1050Ti Mobile` con 4 Gb de memoria. Como sistema operativo se emplea `Ubuntu 22.04.4 LTS` equipado con el driver `535.161.08` de NVIDIA y la librería `CuDNN 8.9.9`. No se dispone de la librería `TensorRT`. Además, se hace uso del sistema de control de versiones `Git`. El código empleado está disponible públicamente en `GitHub` en la dirección

una función diferenciable respecto de todos los valores de $\boldsymbol{\theta}$ porque la función ReLU no es diferenciable en 0. Por ello, se recurre al concepto de subgradiente. En los puntos en los que la función es diferenciable, el gradiente y el subgradiente coinciden. Cuando la función no es diferenciable, se toma como gradiente la pendiente de un subgradiente, es decir, el de una recta que localmente corta a la función en un único punto. Puede encontrar más detalles en Shalev-Shwartz *et al.* (2014, p. 188–191).

[7] Los detalles de este cálculo se omiten debido a su larga extensión. Puede encontrar más detalles en Goodfellow *et al.* (2016, p. 204–219).

[8] Como las derivadas se obtienen aplicado la regla de la cadena sucesivamente, la derivada con respecto de los parámetros de una capa se obtiene a partir de las derivadas de la capa posterior. De esta forma, todas las derivadas pueden calcularse desde la última capa hacia la primera. Por este motivo, este procedimiento se conoce con el nombre de *retropropagación* (del inglés *backpropagation*).

[9] `Adam` es un algoritmo del tipo: descenso estocástico del gradiente con momentos de primer y segundo orden. El uso de momentos significa que, en cada iteración, además de tener en cuenta el gradiente calculado en esa iteración, tiene en cuenta la dirección y magnitud de los movimientos que ha realizado en iteraciones anteriores. Por tanto, depende de otros dos parámetros fundamentales β_1 y β_2 que dan cuenta de las iteraciones anteriores. Se fijan en los valores habituales de $\beta_1 = 0.9$ y $\beta_2 = 0.999$ (Kingma *et al.*, 2014) y no se modifican en nigún momento.

[10] Después del entrenamiento se migró el modelo a la versión `2.16.1`, que tiene `Keras 3` en lugar de `Keras 2`.

github.com/sergiogpajares/TFG-fis.

Para el entrenamiento, calculamos las derivadas de (3.1) usando las 5190 imágenes de entrenamiento aumentadas. Después de cada época, se evalúa la función de pérdida sobre todo el conjunto de 690 imágenes de validación y sobre todo el de entrenamiento. Si la función de pérdida en el conjunto de validación no ha mejorado después de 3 épocas, entonces la tasa de aprendizaje se reduce un factor 10. El valor inicial de la tasa de aprendizaje es 10^{-3}. La red se entrena durante 50 épocas con los pesos asignados a cada categoría (véase tabla 3.1). Nótese que se usan distintos pesos al principio y al final.

Tabla 3.1: Pesos λ_c aplicados a cada clase durante el entrenamiento. Los pesos solo se tienen en cuenta en en el conjunto de entrenamiento.

	Épocas	**N/A**	**Cielo desp.**	**Sol**	**Nube espesa**	**Nube trans.**
Peso	1–39	1	1	5	1	0.5
Peso	40–50	1	1	2	1	0.8

Los pesos se asignan para tener en cuenta la diferente representación en las imágenes de las diferentes categorías (véase figura 2.7). Por ello, aumentamos mucho el peso del sol sobre las demás etiquetas. Aunque la nube fina también esta infrarepresentada, sus bordes son mucho más difusos y su etiquetado más subjetivo. Se decide reducir su peso respecto de las demás etiquetas.

La figura 3.9 muestra la evolución de la función de pérdida durante el entrenamiento. El entrenamiento ha durado 5 horas. Téngase en cuenta que para el conjunto de validación, la función de perdida que se evalúa tiene todos los pesos iguales a 1[11] haciendo distantes los valores de ambas funciones. Podemos ver como la función de pérdida decae rápidamente para ambos conjuntos. Sin embargo, para el conjunto de validación se estanca mientras que para el conjunto de entrenamiento continua decreciendo. Este efecto se conoce como sobreajuste (del inglés *overfitting*).

El sobreajuste sucede cuando la red se vuelve demasiado específica en clasificar las imágenes del entrenamiento y pierde la capacidad de generalizar para imágenes que no ha visto nunca. Para evitar el sobreajuste, detenemos el entrenamiento del modelo una vez hemos comprobado que la función de pérdida sobre el conjunto de validación no disminuye más. Puede encontrar información detallada sobre el sobreajuste en Goodfellow *et al.* (2016, p. 110–116).

Otro aspecto importante de la figura 3.9 es la aparente falta de suavidad de la evolución de la función de pérdida. Esto se debe al entrenamiento por lotes. De esta forma, los pasos del algoritmo iterativo no son siempre en la dirección correcta, sino que solo son correctos en promedio.

Un último aspecto importante es la bajada repentina en validación en la época 39. Esto se debe a que, en esa época, paramos el entrenamiento para cambiar los pesos. Como los pesos de entrenamiento se acercaron más a los pesos de validación, la diferencia entre ambas evoluciones se reduce rápidamente.

[11] Hubiese sido preferible haber empleado los mismos pesos para validación y entrenamiento a fin de poder dar una comparación más realista de ambas funciones de pérdida.

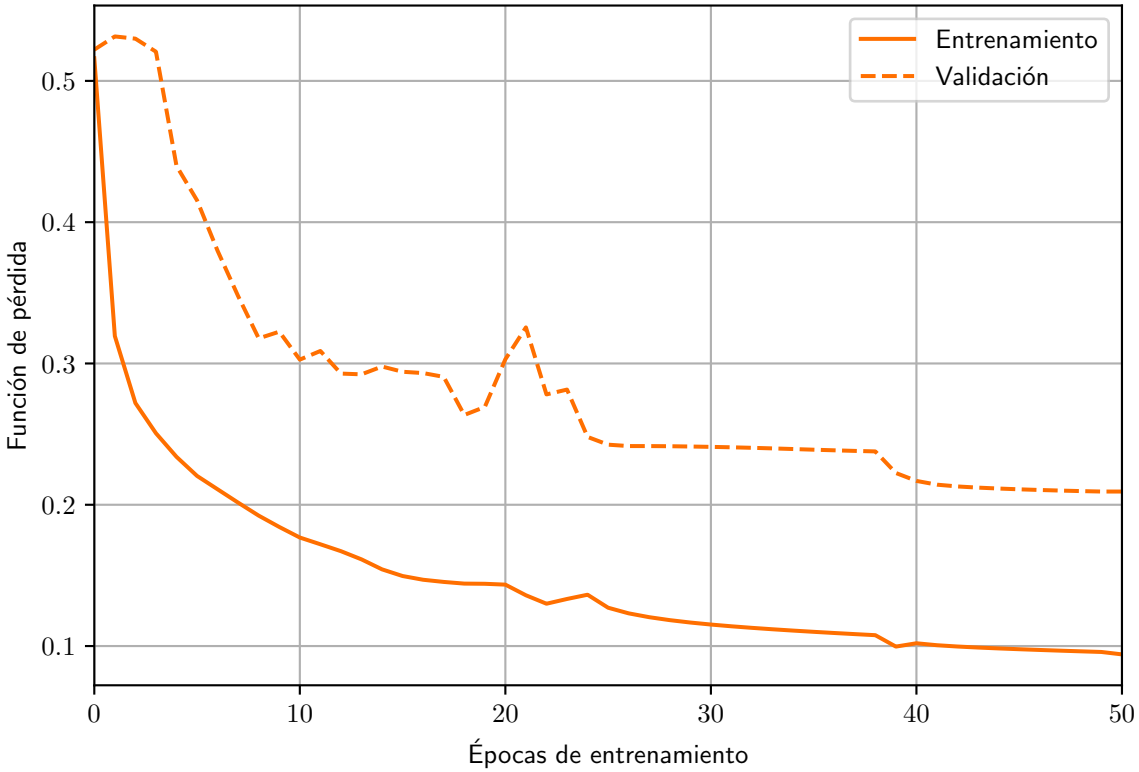


Figura 3.9: Evolución de la función de pérdida (3.2). Los pesos se aplican a la función de pérdida del conjunto de entrenamiento, en el conjunto de validación los pesos son todos iguales a 1.

Capítulo 4

RESULTADOS

Separamos los resultados en dos partes: rendimiento del modelo de segmentación semántica y aplicación directa a la medida de la cobertura nubosa. Para medir el rendimiento en la tarea de segmentación vamos a estudiar, a nivel de píxel, la precisión del modelo en clasificar la categoría correcta. Para la medida de la cobertura nubosa vamos a estudiar el error cometido en la estimación del número de octas de cielo cubiertas.

4.1 SEGMENTACIÓN SEMÁNTICA

Con el modelo ajustado, podemos evaluar su desempeño sobre el conjunto de 46 imágenes de rendimiento. Lo primero es hacer una comparación cualitativa de las máscaras de referencia y las máscaras inferidas por el modelo. La figura 4.1 muestra esta comparación para 4 ejemplos aleatorios[1]. Se puede comprobar como la red reconoce el cielo despejado con total claridad y también es capaz de distinguir de manera general la localización de las grandes masas nubosas. Como era de esperar, el error es cometido en los bordes de las nubes, pues hasta para el meteorólogo más experto[2] puede resultar difícil determinar donde empieza y termina una nube. El sol se detecta sin problemas y la etiqueta N/A, fuera de la bóveda celeste, es muy cercana. La red falla decidiendo que nubes son espesas y cuales finas, como se observa en el último ejemplo de la figura 4.1.

Lo segundo es obtener una medida cuantitativa del desempeño del modelo. Una primera aproximación podría ser fijarse en los valores de la función de pérdida (3.2), pues definimos esta para medir el error cometido por la red. Sin embargo, existen dos problemas asociados: la función de pérdida mide el error cometido en la distribución de masa de probabilidad que genera la red, no dice nada sobre la máscara de segmentación; y distintos modelos emplean distintas funciones de perdida, los resultado no serían comparables.

Como medida cuantitativa del desempeño del modelo vamos a usar la matriz de confusión del modelo, representada en la figura 4.2. La matriz de confusión C_{ij} presenta el número de píxeles que perteneciendo a la categoría i fueron clasificados en la categoría j.

[1] La comparación entre la máscara inferida y la máscara de referencia para todas las imágenes puede consultarse en la información suplementaria online, concretamente en: github.com/sergiogpajares/TFG-fis.

[2] Existe el concepto de la zona de transición entre nube y aerosol que alude a la falta de definición del borde de la nube (Jahani *et al.*, 2019; Calbó; González *et al.*, 2024).

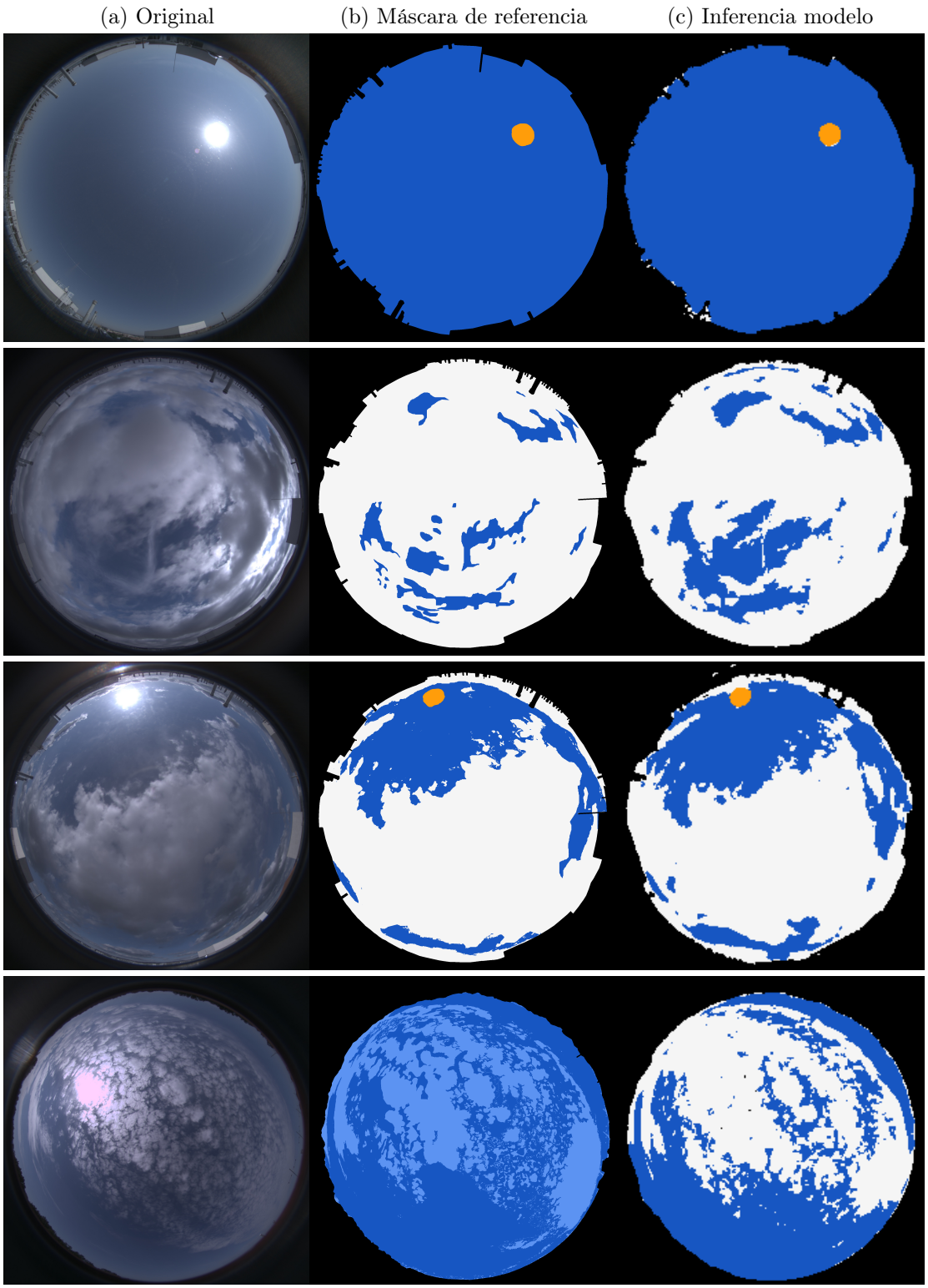


Figura 4.1: Comparativa entre la imagen original (izquierda), mascara de referencia (centro) e inferencia del modelo (derecha) para cuatro imágenes aleatorias del conjunto de 46 imágenes de rendimiento.

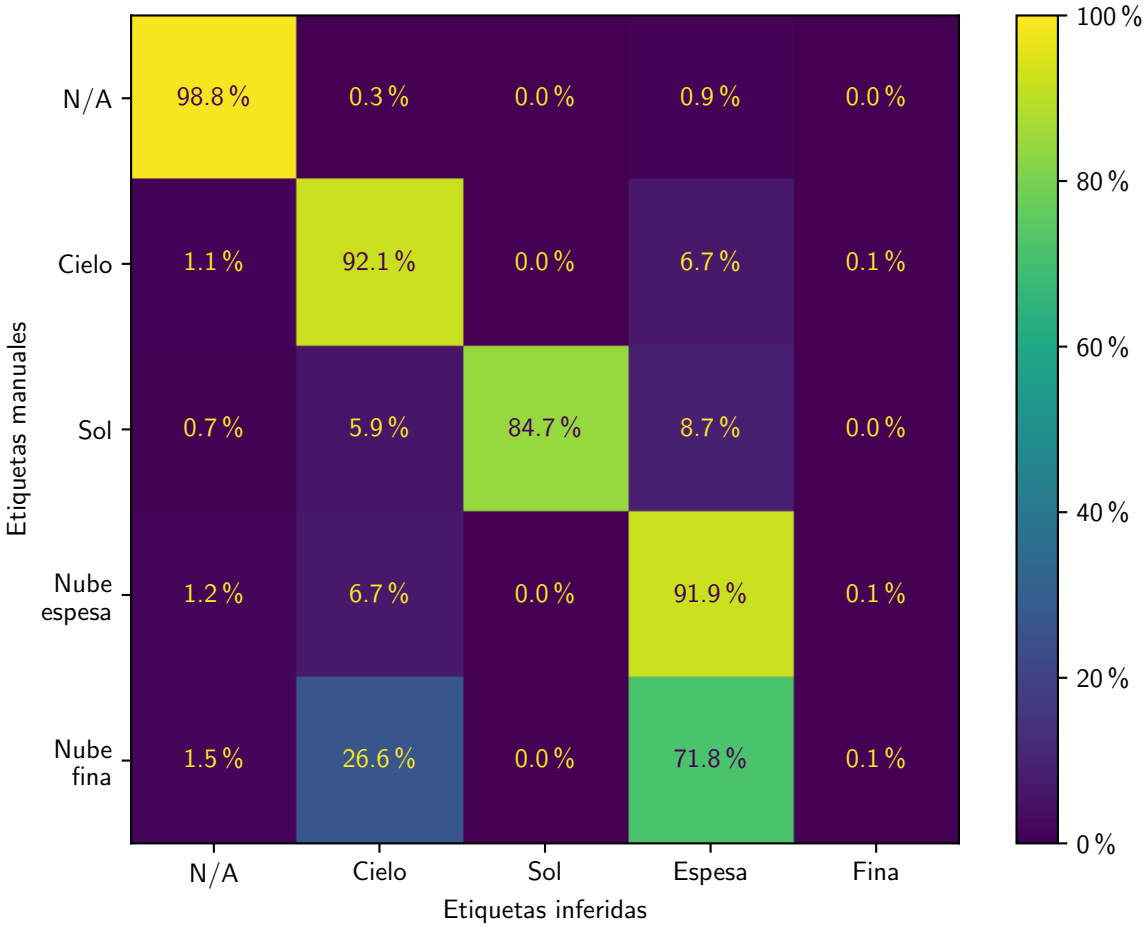


Figura 4.2: Matriz de confusión porcentual para las diferentes clases normalizada a las etiquetas verdaderas.

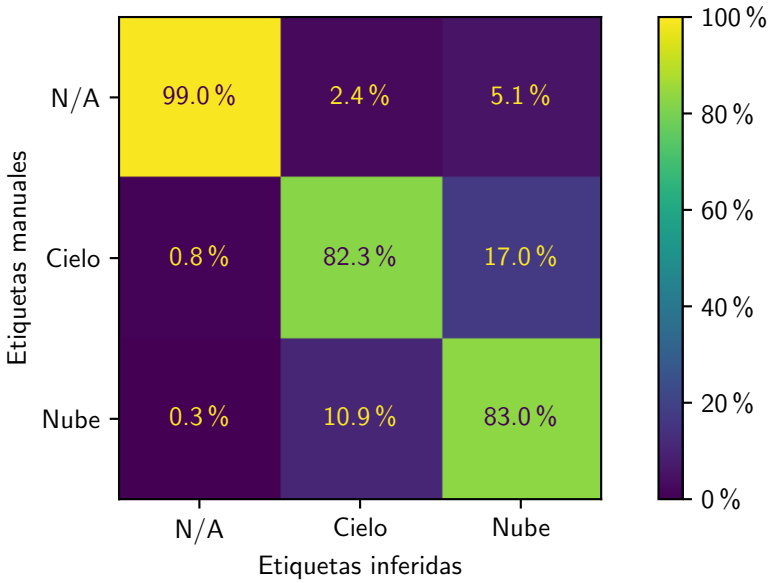


Figura 4.3: Matriz de confusión porcentual para el modelo U-NET (Ronneberger *et al.*, 2015) entrenado para la segmentación semántica de nubes por Sanz Huidobro (2023). Los valores están normalizados a las etiquetas verdaderas.

Como no todas las categorías tienen el mismo número de píxeles, los resultados se normalizan en función de las etiquetas manuales (por filas) y se muestran porcentualmente. Estos resultados han de tomarse con cautela porque solo se dispone de 46 imágenes en el conjunto de rendimiento.

Al igual que habíamos apreciado cualitativamente, la clasificación acierta en la mayoría de los píxeles, salvo para la nube fina. El modelo no predice nubes finas y son clasificadas como nubes espesas o como cielo despejado. Esto se debe probablemente a su infrarepresentación en el conjunto de datos (figura 2.7) y a su menor peso durante el entrenamiento (tabla 3.1). Las nubes espesas y el cielo despejado se clasifican adecuadamente. Las pequeñas desviaciones son que se prediga nube espesa para el cielo despejado o viceversa. Esto se debe principalmente a que el modelo no es preciso en los bordes. Lo mismo sucede en el caso del sol, que se confunde con cielo y nubes por sus bordes. La clasificación del sol frente al cielo o nubes es especialmente difícil por el halo luminoso que se produce a su alrededor.

El conjunto de rendimiento contaba con 9 imágenes nocturnas y 37 imágenes diurnas ($\sim 20\,\%$), lo cual es una proporción nocturna ligeramente superior que la que hay a nivel global en el conjunto de datos ($\sim 13\,\%$). Aunque hay muy pocas imágenes nocturnas, resultando en una estadística pobre, podemos preguntarnos como varía el rendimiento del modelo para ambos conjuntos de imágenes. La tabla 4.1 recoge los positivos verdaderos de las imágenes diurnas y nocturnas. El modelo mejora detectando el cielo despejado durante el día un $\sim 10\,\%$ pero no se aprecian diferencias significativas para la detección de nubes.

Tabla 4.1: Positivos verdaderos (diagonal de la matriz de confusión) segregando imágenes nocturnas y diurnas. Se normaliza a la categoría de referencia. Nótese que en las imágenes nocturnas no hay píxeles de sol. Se añade una columna adicional en la que se trata indistintamente las nubes espesas y finas.

	N/A	**Cielo desp.**	**Sol**	**Nube espesa**	**Nube fina**	**Nube**
Día	98.9 %	93.2 %	84.7 %	92.5 %	0.1 %	91.5 %
Noche	98.1 %	84.6 %	–	90.2 %	0.0 %	89.8 %
Todas	98.8 %	92.1 %	84.7 %	91.9 %	0.1 %	91.0 %

Estos resultados pueden compararse con otros TFGs de la Universidad de Valladolid. Existe un único TFG que aborde la segmentación semántica de nubes. Sanz Huidobro (2023) entrenó la arquitectura U-NET (Ronneberger *et al.*, 2015) para la segmentación semántica de imágenes del cielo en solo 3 categorías: N/A, cielo despejado y nube. La figura 4.3 muestra los resultados que obtuvo. Puede comprobarse como el modelo que proponemos mejora la detección de cielo y nubes en un 10 %. Aunque los resultados no son directamente comparables, pues Sanz Huidobro (2023) usó un conjunto de datos distinto y solo trató con imágenes diurnas.

Podemos ver cómo varía el rendimiento en función de la cobertura nubosa de la imagen sobre la que se aplica. La tabla 4.2 muestra el número de positivos verdaderos para imágenes con cielos totalmente despejados (0 octas), parcialmente cubiertos (1–7 octas) y totalmente cubiertos (8 octas). Para imágenes totalmente despejadas el modelo mejora, como era de esperar, pues el conjunto de datos contaba con más de estas imágenes (figura 2.6). El rendimiento se mantiene para imágenes totalmente cubiertas. Para imágenes parcialmente cubiertas, que tienen muchos bordes y había una menor cantidad en el conjunto de datos, la precisión en la detección de nubes y cielo despejado se reduce notablemente.

Tabla 4.2: Positivos verdaderos (diagonal de la matriz de confusión) segregando imágenes en función de la cobertura nubosa de referencia calculada según (4.1). Se normaliza a la categoría de referencia. Se considera cielo despejado con 0 octas, cielo parcialmente cubierto de 1 a 7 octas y cielo totalmente cubierto con 8 octas.

	N/A	**Cielo desp.**	**Sol**	**Nube espesa**	**Nube fina**	**Nube**
Depejado	99.0 %	97.4 %	82.7 %	–	–	–
Parcialmente	98.8 %	79.8 %	90.0 %	64.9 %	0.0 %	64.8 %
Totalmente	98.2 %	–	–	92.1 %	0.0 %	92.0 %

4.2 COBERTURA NUBOSA DEL CIELO

Una aplicación directa de la máscara de segmentación semántica es la medida de la cobertura nubosa del cielo. La cobertura nubosa CC (del inglés *cloud cover*) se mide habitualmente en octas, es decir, en el número de octavos de la bóveda celeste cubiertos. La cobertura nubosa en octas puede calcularse, para cada imagen, a partir del número de píxeles de cada categoría como:

$$\text{CC (octas)} = \text{redondeo}\left(8\,\frac{\text{NUBE}}{\text{CIELO DESP.} + \text{SOL} + \text{NUBE}}\right) \tag{4.1}$$

donde NUBE, SOL y CIELO DESP. corresponde al número de píxeles de cada una de esas categorías respectivamente. La supercategoría NUBE es la suma de los píxeles asociados a nube espesa y nube fina.

Aplicando (4.1) a las 46 imágenes del conjunto de rendimiento para las máscaras manuales y las máscaras inferidas, podemos calcular la cobertura nubosa de referencia e inferida por la red y el error cometido por esta. La figura 4.4 muestra un gráfico de barras de este error segmentando las imágenes diurnas y nocturnas y la tabla 4.3 algunos estimadores estadísticos asociados.

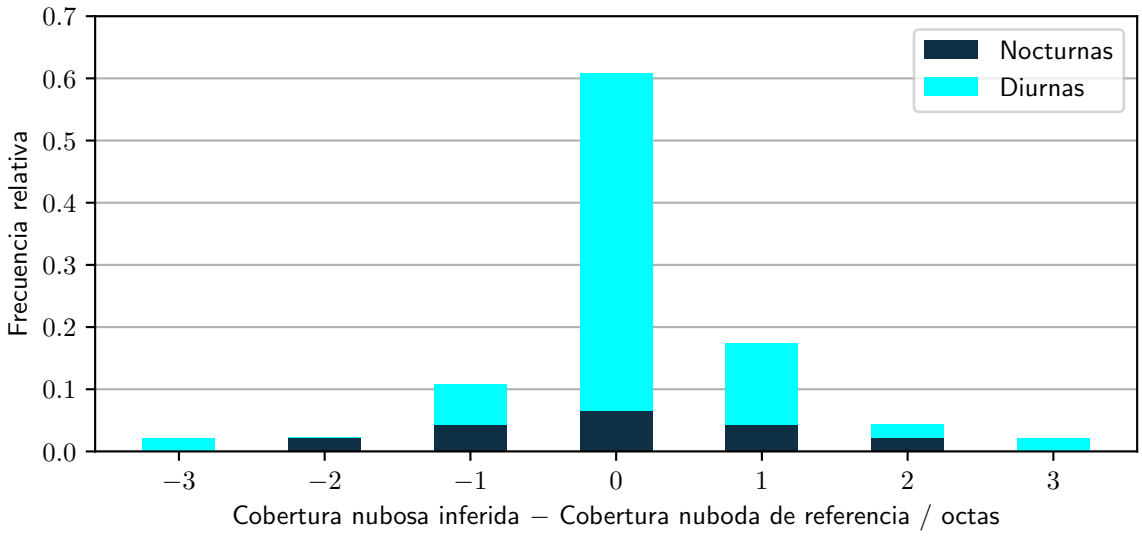


Figura 4.4: Frecuencia relativa del error cometido por la red en la inferencia de la cobertura nubosa en octas. Los datos se segmentan en imágenes nocturnas y diurnas acumuladas.

El modelo propuesto sobrestima la cobertura nubosa muy ligeramente por lo que es une estimador prácticamente insesgado. En este caso los resultados para el día son mucho

Tabla 4.3: Estimadores estadísticos del error cometido por la red en la inferencia de la cobertura nubosa: media y desviación estándar (SD) e intervalos de confianza asociados a una medida. También se incluyen los resultados afirmados por otros Trabajos de Fin de Grado de la Universidad de Valladolid para permitir la comparación. Téngase en cuenta que Martínez Celda (2021) y Alegre Fernández (2022) solo trabajaron con imágenes diurnas mientras que Calvo Herrero (2023) trabajó con imágenes nocturnas y diurnas.

	Media (octas)	**SD (octas)**	**Intervalo confianza (octas)**		
			0	**±1**	**±2**
Día	+0.14	0.91	68 %	92 %	95 %
Noche	+0.00	1.16	33 %	78 %	100 %
Todas	+0.11	0.96	61 %	89 %	97 %
Martínez Celda (2021)	−0.01	1.05	–	–	–
Alegre Fernández (2022)	+0.05	0.69	75 %	96 %	–
Calvo Herrero (2023)	−0.04	0.84	67 %	97 %	–

mejores (∼ 15 % más confianza en el intervalo ±1 octa). Aunque, recordemos que los datos para la noche debemos tomarlos con especial cautela debido al minúsculo número de imágenes disponibles. Desgraciadamente, con un número tan pequeño no podemos estudiar la matriz de confusión para conocer en qué casos (cielo totalmente despejado, parcialmente cubierto y totalmente cubierto) el modelo comete más errores.

Estos resultados pueden compararse con los obtenidos por otros TFGs de la Universidad de Valladolid. Hasta ahora se han publicado tres trabajos que traten la medida de la cobertura nubosa con CNN: Martínez Celda (2021), Alegre Fernández (2022) y Calvo Herrero (2023). Estos trabajos abordan la cobertura nubosa como un problema de *clasificación* en lugar de *segmentación*, entrenando modelos cuya entrada es una imagen todo cielo y la salida el número de octas. Martínez Celda (2021) y Alegre Fernández (2022) trabajaron con imágenes todo cielo diurnas, mientras que Calvo Herrero (2023) incluyo imágenes nocturnas. Para el entrenamiento dispusieron de una cantidad mucho mayor de imágenes: ∼ 26 000, 34 826 y 3 361 respectivamente. Todos ellos presentan entorno a una docena de modelos con distintas configuraciones, la tabla 4.3 contiene algunos estimadores para el mejor de los modelos que consiguió cada uno.

Los resultados de Alegre Fernández (2022) y Calvo Herrero (2023) fueron ligeramente mejores que nuestros resultados. Esto seguramente este ligado a que ellos dispusieron de 100 y 10 veces más imágenes respectivamente que nosotros para el entrenamiento del modelo. Pese a disponer de menos imágenes, Calvo Herrero (2023) mejoró ligeramente los resultados de Alegre Fernández (2022).

La comparación entre estos trabajos y el presente debe hacerse con cautela, pues las 46 imágenes del conjunto de rendimiento pueden no ser suficientes para una estadística fiable. La conclusión fundamental es que nuestro modelo, pese a no estar entrenado para la clasificación de imágenes, es capaz de replicar de cerca modelos hechos *ad hoc* para esta tarea, dando más valor y validación a la tarea de segmentación semántica.

Capítulo 5

CONCLUSIONES

Para generar máscaras de segmentación semantica del cielo, los algoritmos basados en la aplicación de umbrales han resultado ser insuficientes. Para remplazar estos algoritmos, surgen los algoritmos basados en redes neuronales convolucionales. Estos son familias de funciones obtenidas mediante la concatenación sucesiva de funciones sencillas siguiendo patrones determinados y que, mediante un proceso de optimización con un conjunto de datos, se pueden emplear para la segmentación semántica de imágenes de cielo.

Nuestro conjunto de datos consta de 346 imágenes provenientes de la red de cámaras todo cielo del GOA-UVa que han sido segmentadas manualmente en cinco categorías: N/A, cielo despejado, sol, nube espesa y nube fina. Cuenta con imágenes diurnas y nocturnas ($\sim 10\,\%$) y con variedad de estados del cielo, principalmente: totalmente despejado y totalmente cubierto (figura 2.6). El conjunto de datos se divide de forma aleatoria en tres subconjuntos más pequeños: entrenamiento, validación y rendimiento.

La arquitectura propuesta se basa en un modelo *encoder–decoder*. Como *encoder*, se utiliza la red VGG16, compuesta de cinco bloques con dos o tres capas convolucionales y una capa de Maxpooling. Como *decoder* se emplean cuatro bloques formados por una convolución transpuesta, una concatenación y dos capas convolucionales (figura 3.7). Al final de la red se sitúa la función softmax para generar un mapa de probabilidad para cada categoría, que se convierten en la predicción final (figura 3.8).

Se trata de una arquitectura muy ligera cuyo entrenamiento se puede abordar con un ordenador portátil en una cantidad de tiempo razonable. Además, el uso de un *encoder* preestablecido como VGG16 permite usar pesos preentrenados sobre el conjunto ImageNet, reduciendo drásticamente el coste computacional.

El modelo se entrena usando la entropía cruzada multicategoría con pesos durante 50 épocas, empleando una tasa de aprendizaje variable. Consigue detectar muy bien la forma general de las nubes, teniendo problemas para detectar los bordes, en este sentido. El $93\,\%$ de los píxeles son clasificados adecuadamente. Se cumple asi el primer objetivo de este trabajo.

El borde de la imagen, el cielo despejado y la nube espesa son detectadas con un número de positivos verdaderos superior al $90\,\%$. Sin embargo, las nubes finas no son detectadas y, casi en la totalidad de las ocasiones, son marcadas como nube espesa o como cielo despejado

(figura 4.2). Recordemos que esta categoría estaba infrarepresentada en el conjunto de datos (figura 2.7) y que se usó un peso menor durante el entrenamiento (tabla 3.1). Un trabajo futuro podría tratar de entrenar el modelo dando mayor peso a la nube fina. El sol, que aparece mucho menos en las imágenes, también es detectado con un número de positivos verdaderos del 84 %. Estos resultados son capaces de mejorar el TFG anterior de Sanz Huidobro (2023) en aproximadamente un 10 %. Con esto, cumplimos el segundo objetivo de este trabajo.

Cuando segregamos los datos en función del día y la noche (tabla 4.1), vemos como la precisión del modelo aumenta durante el día y se reduce por la noche. La categoría a la que más afecta es el cielo despejado, bajando los positivos verdaderos del 92 % al 85 % durante la noche. El conjunto de datos apenas contiene un 10 % de imágenes nocturnas.

Si segregamos los datos en función de la cobertura nubosa de la imagen (tabla 4.2) vemos como el modelo es especialmente preciso en las imágenes de cielo despejadas y mantiene una alta precisión en las imágenes totalmente cubiertas. El problema viene en las imágenes parcialmente cubiertas, que tienden a tener un número grande de bordes de nubes dispersos. Estos bordes son los que producen una caída drástica del número de positivos verdaderos del cielo despejado (80 %) y la nube espesa (65 %).

Podemos concluir que los modelos de tipo *encoder-decoder* son aptos para la generación de las máscaras de segmentación semántica. Estas máscaras se pueden usar para obtener otras medidas cómo saber si hay ocultación o no del sol o medir la cobertura nubosa del cielo. El 89 % de las medidas tienen un error igual o inferior a ± 1 octas (tabla 4.3), mejorando este dato hasta el 92 % en las imágenes diurnas. Además, la estimación de la cobertura nubosa es insesgada.

La estimación de la cobertura nubosa realizada por otros TFGs mediante redes neuronales es fundamentalmente distinta de la aquí presentada. Estos trabajos emplearon un modelo que clasifica las imágenes directamente en función de su cobertura nubosa. Sus resultados consiguen mejorar ligeramente los aquí presentados (tabla 4.3). Esos trabajos emplean conjuntos de imágenes con un tamaño de 10 a 100 veces superior al nuestro y sus coberturas nubosas de referencia están sujetas a la subjetividad del observador humano que clasifica la imagen.

Pese a que nuestros resultados no son tan buenos como los de otros trabajos, no debemos descartar la segmentación semántica para la medida de la cobertura nubosa, sino que debemos mejorar la estimación de la máscara. El uso de la máscara elimina la subjetividad de las mediciones y la conclusión fundamental que debemos extraer es que el modelo es capaz de obtener, con alta precisión, medidas derivadas que refuerzan su validación. Se cumple, por tanto, el tercer objetivo de este trabajo.

Una investigación posterior podría usar las miles de imágenes clasificadas en otros TFGs para validar la inferencia de la cobertura nubosa. Si una imagen ha sido clasificada solo con su cobertura nubosa, no es necesaria ninguna máscara de referencia para aplicar el modelo e inferir su cobertura nubosa. De esta forma, se eliminan las limitaciones asociadas a solo disponer de 46 imágenes para validar la inferencia de la cobertura nubosa.

La mejora de este modelo pasa, sin lugar a dudas, por el aumento del tamaño del conjunto de datos y de la variedad de sus imágenes. Son necesarias más imágenes nocturnas y más imágenes parcialmente cubiertas. En este sentido, se podría explorar la posibilidad de

realizar más aumentación de datos de aquellas imágenes infrarepresentadas en el conjunto de datos original y menos de aquellas sobrerepresentadas. Además, este modelo se entrenó reduciendo el tamaño de las máscaras con la interpolación lineal, lo que introdujo artefactos en estas (figura 2.5). Entrenar de nuevo el modelo con las máscaras correctas también podría mejorarlo.

Futuros trabajos pueden enfocarse en el aumento de la resolución de las imágenes de entrada, mejorando los 224×224 píxeles actuales. Esto permitiría una mejor clasificación de los bordes de las nubes, pues la red tendría acceso a texturas mucho más detalladas. Esto es especialmente importante si queremos clasificar adecuadamente las imágenes parcialmente cubiertas.

También se puede trabajar en implementar otros modelos que se han usado en otras tareas de segmentación de nubes similares, como el modelo DeeplabV3+. Este modelo además se puede conseguir preentrenado sobre el conjunto de datos ImageNet. Se realizaron algunas pruebas al inicio del discurso de este trabajo, pero el coste computacional era demasiado alto para ejecutar adecuadamente el modelo. En este sentido, cabe destacar que existen plataformas online como Kaggle o Google Colab que, aunque muy limitadas, permiten el acceso a recursos computacionales de manera gratuita.

Un objetivo mucho más ambicioso que la obtención de máscaras de segmentación sería conseguir un modelo capaz de inferir el espesor óptico de las nubes directamente. Esto eliminaría la subjetividad en la clasificación de nubes como fina o espesa y del espesor óptico podrían obtenerse también las máscaras de segmentación.

BIBLIOGRAFÍA

ABADI, Martin; BARHAM, Paul; CHEN, Jianmin; CHEN, Zhifeng; DAVIS, Andy; DEAN, Jeffrey; DEVIN, Matthieu; GHEMAWAT, Sanjay; IRVING, Geoffrey; ISARD, Michael; KUDLUR, Manjunath; LEVENBERG, Josh; MONGA, Rajat; MOORE, Sherry; MURRAY, Derek G.; STEINER, Benoit; TUCKER, Paul; VASUDEVAN, Vijay; WARDEN, Pete; WICKE, Martin; YU, Yuan y ZHENG, Xiaoqiang, 2016. TensorFlow: A system for large-scale machine learning. En: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, págs. 265-283. También disponible en: https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf.

ALEGRE FERNÁNDEZ, Sergio, 2022. Clasificación automática de imágenes del cielo mediante inteligencia artificial. También disponible en: https://uvadoc.uva.es/handle/10324/58262.

ANTUÑA-SÁNCHEZ, J.C., 2021. Configuración y metodología para el uso de cámaras de todo cielo en la obtención de parámetros atmosféricos. Disp. desde DOI: 10.35376/10324/60022.

ANTUÑA-SÁNCHEZ, J.C.; ROMÁN, R.; CACHORRO, Victoria E.; TOLEDANO, Carlos; LÓPEZ, César; GONZÁLEZ, Ramiro; MATEOS, David; CALLE, Abel y FRUTOS, Ángel M. de, 2021. Relative sky radiance from multi-exposure all-sky camera images. *Atmospheric Measurement Techniques*. Vol. 14, págs. 2201-2217. ISSN 1867-8548. Disp. desde DOI: 10.5194/amt-14-2201-2021.

BOUCHER, O; RANDALL, D; ARTAXO, P; BRETHERTON, C; FEINGOLD, G; FORSTER, P; KERMINEN, V-m; KONDO, Y; LIAO, H; LOHMANN, U; RASCH, P; SATHEESH, SK; SHERWOOD, S; STEVENS, B; ZHANG, XY; QIN, D; PLATTNER, G-k; TIGNOR, M; ALLEN, SK; BOSCHUNG, J; NAUELS, A; XIA, Y; BEX, V; MIDGLEY, PM; BOUCHER, Olivier y RANDALL, David, 2013. Clouds and Aerosols. En: *Climate Change 2013 – The Physical Science Basis: Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press, cap. 7, págs. 571-658.

BRADSKI, G., 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

BUSLAEV, Alexander; IGLOVIKOV, Vladimir I.; KHVEDCHENYA, Eugene; PARINOV, Alex; DRUZHININ, Mikhail y KALININ, Alexandr A., 2020. Albumentations: Fast and Flexible Image Augmentations. *Information*. Vol. 11, n.º 2. ISSN 2078-2489. Disp. desde DOI: 10.3390/info11020125.

CALBÓ, Josep; GONZÁLEZ, Josep-Abel; JAHANI, Babak; SOLA, Yolanda y MORALES, Jaume Ruiz de, 2024. How important is the transition zone between clouds and aerosol? En: disp. desde DOI: 10.1063/5.0182769.

CALBÓ, Josep y SABBURG, Jeff, 2008. Feature extraction from Whole-sky ground-based images for cloud-type recognition. *Journal of Atmospheric and Oceanic Technology*. Vol. 25, págs. 3-14. ISSN 07390572. Disp. desde DOI: 10.1175/2007JTECHA959.1.

CALVO HERRERO, Carolina, 2023. Clasificación automática de imágenes de cielo mediante Inteligencia Artificial. También disponible en: https://uvadoc.uva.es/handle/10324/63294.

CAZORLA, A.; HUSILLOS, C.; ANTÓN, M. y ALADOS-ARBOLEDAS, L., 2015. Multi-exposure adaptive threshold technique for cloud detection with sky imagers. *Solar Energy*. Vol. 114, págs. 268-277. ISSN 0038092X. Disp. desde DOI: 10.1016/j.solener.2015.02.006.

CHEN, Liang-Chieh; ZHU, Yukun; PAPANDREOU, George; SCHROFF, Florian y ADAM, Hartwig, 2018. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. También disponible en: http://arxiv.org/abs/1802.02611.

CHOW, Chi Wai; URQUHART, Bryan; LAVE, Matthew; DOMINGUEZ, Anthony; KLEISSL, Jan; SHIELDS, Janet y WASHOM, Byron, 2011. Intra-hour forecasting with a total sky imager at the UC San Diego solar energy testbed. *Solar Energy*. Vol. 85, págs. 2881-2893. ISSN 0038092X. Disp. desde DOI: 10.1016/j.solener.2011.08.025.

DEV, Soumyabrata; LEE, Yee Hui y WINKLER, Stefan, 2017. Color-Based Segmentation of Sky/Cloud Images from Ground-Based Cameras. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*. Vol. 10, págs. 231-242. ISSN 21511535. Disp. desde DOI: 10.1109/JSTARS.2016.2558474. SWIMSEG Database.

DEV, Soumyabrata; NAUTIYAL, Atul; LEE, Yee Hui y WINKLER, Stefan, 2019. CloudSegNet: A Deep Network for Nychthemeron Cloud Image Segmentation. *IEEE Geoscience and Remote Sensing Letters*. Vol. 16, págs. 1814-1818. ISSN 15580571. Disp. desde DOI: 10.1109/LGRS.2019.2912140.

DUMOULIN, Vincent y VISIN, Francesco, 2016. A guide to convolution arithmetic for deep learning. También disponible en: http://arxiv.org/abs/1603.07285.

FA, Tao; XIE, Wanyi; WANG, Yiren y XIA, Yingwei, 2019. Development of an all-sky imaging system for cloud cover assessment. *Applied Optics*. Vol. 58, pág. 5516. ISSN 1559-128X. Disp. desde DOI: 10.1364/ao.58.005516. U-NETWSISEG.

FITZGERALD, Richard J., 2017. International Cloud Atlas. *Physics Today*. Vol. 70, págs. 76-76. ISSN 0031-9228. Disp. desde DOI: 10.1063/PT.3.3567.

FORSTER, P.; STORELVMO, T.; ARMOUR, k.; COLLINS, W.; DUFRESNE, J. L.; FRAME, D.; LUNT, D. J.; MAURITSEN, T.; PALMER, M. D.; WATANABE, M.; WILD, M. y ZHANG, H., 2023. The Earth's Energy Budget, Climate Feedbacks and Climate Sensitivity. En: Cambridge University Press, págs. 923-1054. Disp. desde DOI: 10.1017/9781009157896.009.

FRESNEDA, Ramiro Romero; GARCÍA, José Vicente Moreno; NÚÑEZ, Lourdes Martínez; ITULAIN, María Teresa Huarte; BALLESTERO, César Rodríguez y FULLAT, Roser Botey, 2020. *Comportamiento de las precipitaciones en España y periodos de sequía (periodo 1961-2018)*. Agencia Estatal de Meteorología. Disp. desde DOI: 10.31978/666-20-006-0.

GHONIMA, M. S.; URQUHART, B.; CHOW, C. W.; SHIELDS, J. E.; CAZORLA, A. y KLEISSL, J., 2012. A method for cloud detection and opacity classification based on ground based sky imagery. *Atmospheric Measurement Techniques*. Vol. 5, págs. 2881-2892. ISSN 18671381. Disp. desde DOI: 10.5194/amt-5-2881-2012.

GOODFELLOW, Ian; BENGIO, Yoshua y COURVILLE, Aaron, 2016. *Deep Learning*. MIT Press. ISBN 9780262337373.

HÄNEL, Andreas; POSCH, Thomas; RIBAS, Salvador J.; AUBÉ, Martin; DURISCOE, Dan; JECHOW, Andreas; KOLLATH, Zoltán; LOLKEMA, Dorien E.; MOORE, Chadwick; SCHMIDT, Norbert; SPOELSTRA, Henk; WUCHTERL, Günther y KYBA, Christopher C.M., 2018. *Measuring night sky brightness: methods and challenges*. Vol. 205. Elsevier Ltd. ISSN 00224073. Disp. desde DOI: 10.1016/j.jqsrt.2017.09.008.

HENDRIKX, N. Y.; BARHMI, K.; VISSER, L. R.; BRUIN, T. A. de; PÓ, M.; SALAH, A. A. y SARK, W. G.J.H.M. van, 2024. All sky imaging-based short-term solar irradiance forecasting with Long Short-Term Memory networks. *Solar Energy*. Vol. 272. ISSN 0038092X. Disp. desde DOI: 10.1016/j.solener.2024.112463.

HUO, Juan y LU, Daren, 2009. Cloud determination of All-Sky images under low-visibility conditions. *Journal of Atmospheric and Oceanic Technology*. Vol. 26, págs. 2172-2181. ISSN 07390572. Disp. desde DOI: 10.1175/2009JTECHA1324.1.

JAHANI, Babak; CALBÓ, Josep y GONZÁLEZ, Josep-Abel, 2019. Transition Zone Radiative Effects in Shortwave Radiation Parameterizations: Case of Weather Research and Forecasting Model. *Journal of Geophysical Research: Atmospheres*. Vol. 124, págs. 13091-13104. ISSN 2169-897X. Disp. desde DOI: 10.1029/2019JD031064.

KEGELMEYER, Jr W., 1994. *Extraction of cloud statistics from whole sky imaging cameras*. Disp. desde DOI: 10.2172/10141846.

KHAN, Salman; NASEER, Muzammal; HAYAT, Munawar; ZAMIR, Syed Waqas; KHAN, Fahad Shahbaz y SHAH, Mubarak, 2022. Transformers in Vision: A Survey. *ACM Computing Surveys*. Vol. 54. ISSN 15577341. Disp. desde DOI: 10.1145/3505244.

KINGMA, Diederik P. y BA, Jimmy, 2014. Adam: A Method for Stochastic Optimization. También disponible en: http://arxiv.org/abs/1412.6980.

KREUTER, Axel; ZANGERL, Matthias; SCHWARZMANN, Michael y BLUMTHALER, Mario, 2009. All-sky imaging: a simple, versatile system for atmospheric research. *Applied Optics*. Vol. 48, pág. 1091. ISSN 0003-6935. Disp. desde DOI: 10.1364/AO.48.001091.

LEVIN, Noam; KYBA, Christopher C.M.; ZHANG, Qingling; MIGUEL, Alejandro Sánchez de; ROMÁN, Miguel O.; LI, Xi; PORTNOV, Boris A.; MOLTHAN, Andrew L.; JECHOW, Andreas; MILLER, Steven D.; WANG, Zhuosen; SHRESTHA, Ranjay M. y ELVIDGE, Christopher D., 2020. Remote sensing of night lights: A review and an outlook for the future. *Remote Sensing of Environment*. Vol. 237. ISSN 00344257. Disp. desde DOI: 10.1016/j.rse.2019.111443.

LI, Qingyong; LU, Weitao y YANG, Jun, 2011. A hybrid thresholding algorithm for cloud detection on ground-based color images. *Journal of Atmospheric and Oceanic Technology*. Vol. 28, págs. 1286-1296. ISSN 07390572. Disp. desde DOI: 10.1175/JTECH-D-11-00009.1. HYTA.

LIU, Shuang; ZHANG, Jiafeng; ZHANG, Zhong; CAO, Xiaozhong y DURRANI, Tariq S., 2022. TransCloudSeg: Ground-Based Cloud Image Segmentation With Transformer. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*. Vol. 15, págs. 6121-6132. ISSN 21511535. Disp. desde DOI: 10.1109/JSTARS.2022.3194316.

MARTÍNEZ CELDA, Bernardo, 2021. Clasificación automática de imágenes de cielo mediante inteligencia artificial. También disponible en: https://uvadoc.uva.es/handle/10324/50628.

MIÑANO, Rosario, 2023. Cortes de agua por la sequía en España: estas son las comunidades con restricciones [en línea] [visitado 2024-05-03]. Disp. desde: https://www.antena3.com/noticias/el-tiempo/actualidad/cortes-agua-sequia-espana-estas-son-comunidades-restricciones_2023080464ccedf6502f1e0001cc4896.html.

NIE, Yuhao; LI, Xiatong; PALETTA, Quentin; ARAGON, Max; SCOTT, Andea y BRANDT, Adam, 2024. *Open-source sky image datasets for solar forecasting with deep learning: A comprehensive survey*. Vol. 189. Elsevier Ltd. ISSN 18790690. Disp. desde DOI: 10.1016/j.rser.2023.113977.

PERIS-FERRÚS, Caterina, 2021. Optical properties of cloudy atmospheres through Radiative Transfer and Remote Sensing: From 1D to 3D approach.

REDMON, Joseph; DIVVALA, Santosh; GIRSHICK, Ross y FARHADI, Ali, 2016. You Only Look Once: Unified, Real-Time Object Detection. En: IEEE, págs. 779-788. ISBN 978-1-4673-8851-1. Disp. desde DOI: 10.1109/CVPR.2016.91.

ROMÁN, R.; ANTÓN, M.; CAZORLA, A.; MIGUEL, A. De; OLMO, F. J.; BILBAO, J. y ALADOS-ARBOLEDAS, L., 2012. Calibration of an all-sky camera for obtaining sky radiance at three wavelengths. *Atmospheric Measurement Techniques*. Vol. 5, págs. 2013-2024. ISSN 18671381. Disp. desde DOI: 10.5194/amt-5-2013-2012.

ROMÁN, R.; ANTUÑA-SÁNCHEZ, J.C.; CACHORRO, Victoria E.; TOLEDANO, Carlos; TORRES, Benjamín; MATEOS, David; FUERTES, David; LÓPEZ, César; GONZÁLEZ, Ramiro; LAPIONOK, Tatyana; HERRERAS GIRALDA, Marcos; DUBOVIK, Oleg y FRUTOS, Ángel M. de, 2022. Retrieval of aerosol properties using relative radiance measurements from an all-sky camera. *Atmospheric Measurement Techniques*. Vol. 15, págs. 407-433. ISSN 1867-8548. Disp. desde DOI: 10.5194/amt-15-407-2022.

ROMÁN, R.; CAZORLA, A.; TOLEDANO, C.; OLMO, F. J.; CACHORRO, V. E.; FRUTOS, A. de y ALADOS-ARBOLEDAS, L., 2017. Cloud cover detection combining high dynamic range sky images and ceilometer measurements. *Atmospheric Research*. Vol. 196, págs. 224-236. ISSN 01698095. Disp. desde DOI: 10.1016/j.atmosres.2017.06.006.

ROMÁN, R.; TORRES, B.; FUERTES, D.; CACHORRO, V. E.; DUBOVIK, O.; TOLEDANO, C.; CAZORLA, A.; BARRETO, A.; BOSCH, J. L.; LAPYONOK, T.; GONZÁLEZ, R.; GOLOUB, P.; PERRONE, M. R.; OLMO, F. J.; FRUTOS, A. de y ALADOS-ARBOLEDAS, L., 2017. Remote sensing of lunar aureole with a sky camera: Adding information in the nocturnal retrieval of aerosol properties with GRASP code. *Remote Sensing of Environment*. Vol. 196, págs. 238-252. ISSN 00344257. Disp. desde DOI: 10.1016/j.rse.2017.05.013.

RONNEBERGER, Olaf; FISCHER, Philipp y BROX, Thomas, 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. En: Springer Verlag, vol. 9351, págs. 234-241. ISBN 9783319245737. ISSN 16113349. Disp. desde DOI: 10.1007/978-3-319-24574-4_28.

RUSSAKOVSKY, Olga; DENG, Jia; SU, Hao; KRAUSE, Jonathan; SATHEESH, Sanjeev; MA, Sean; HUANG, Zhiheng; KARPATHY, Andrej; KHOSLA, Aditya; BERNSTEIN, Michael; BERG, Alexander C. y FEI-FEI, Li, 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*. Vol. 115, págs. 211-252. ISSN 0920-5691. Disp. desde DOI: 10.1007/s11263-015-0816-y.

SANZ HUIDOBRO, Sergio, 2023. Análisis de imágenes de cielo con Inteligencia Artificial.

SHALEV-SHWARTZ, Shai y BEN-DAVID, Shai, 2014. *Understanding Machine Learning*. Cambridge University Press. ISBN 9781107057135. Disp. desde DOI: 10.1017/CBO9781107298019.

SHI, Mengyun; XIE, Fengying; ZI, Yue y YIN, Jihao, 2016. Cloud detection of remote sensing images by deep learning. En: IEEE, págs. 701-704. ISBN 978-1-5090-3332-4. Disp. desde DOI: 10.1109/IGARSS.2016.7729176.

SIMONYAN, Karen y ZISSERMAN, Andrew, 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. También disponible en: http://arxiv.org/abs/1409.1556.

TAPAKIS, R. y CHARALAMBIDES, A.G., 2013. Equipment and methodologies for cloud detection and classification: A review. *Solar Energy*. Vol. 95, págs. 392-430. ISSN 0038092X. Disp. desde DOI: 10.1016/j.solener.2012.11.015.

THEVENOT, Alex, 2020. *Conv2d: Finally Understand What Happens in the Forward Pass* [en línea]. [visitado 2024-04-18]. Disp. desde: https://towardsdatascience.com/conv2d-to-finally-understand-what-happens-in-the-forward-pass-1bbaafb0b148.

WMO y JWGFVR, 2012. *WWRP, 2012-1. Recommended Methods for Evaluating Cloud and Related Parameters*. También disponible en: https://library.wmo.int/idurl/4/40114.

XIE, Wanyi; LIU, Dong; YANG, Ming; CHEN, Shaoqing; WANG, Benge; WANG, Zhenzhu; XIA, Yingwei; LIU, Yong; WANG, Yiren y ZHANG, Chaofang, 2020. SegCloud: A novel cloud image segmentation model using a deep convolutional neural network for ground-based all-sky-view camera observation. *Atmospheric Measurement Techniques*. Vol. 13, págs. 1953-1961. ISSN 18678548. Disp. desde DOI: 10.5194/amt-13-1953-2020. Propuesto por Roberto.

YUAN, Kun; MENG, Gaofeng; CHENG, Dongcai; BAI, Jun; XIANG, Shiming y PAN, Chunhong, 2017. Efficient cloud detection in remote sensing images using edge-aware segmentation network and easy-to-hard training strategy. En: IEEE, págs. 61-65. ISBN 978-1-5090-2175-8. Disp. desde DOI: 10.1109/ICIP.2017.8296243.

ZHOU, Zecheng; ZHANG, Feng; XIAO, Haixia; WANG, Fuchang; HONG, Xin; WU, Kun y ZHANG, Jinglin, 2022. A Novel Ground-Based Cloud Image Segmentation Method by Using Deep Transfer Learning. *IEEE Geoscience and Remote Sensing Letters*. Vol. 19. ISSN 15580571. Disp. desde DOI: 10.1109/LGRS.2021.3072618. Prueban varios modelos diferentes entrenandolos con su DataSet y después seleccionan DeepLabV3+ como el mejor de ellos. No modifican modelos. Desarrollado en TensorFlow.Dataset: Referencias Imágenes Cielo completo fisheye.

Apéndice

MATERIAL SUPLEMENTARIO

A.1 ACCESO AL CÓDIGO FUENTE

Todo el código empleado en la realización de este trabajo, en lugar de recopilarse en las sucesivas páginas de este trabajo, puede consultarse en línea bajo el enlace permanente https://github.com/sergiogpajares/TFG-fis. `GitHub` es un repositorio de código en línea donde se encuentran disponible la mayoría del código abierto. El repositorio contiene un archivo `README.md` con información detallada sobre la instalación, estructura y ejecución del código y un archivo `requirements.txt` para la resolución automática de las dependencias del código.

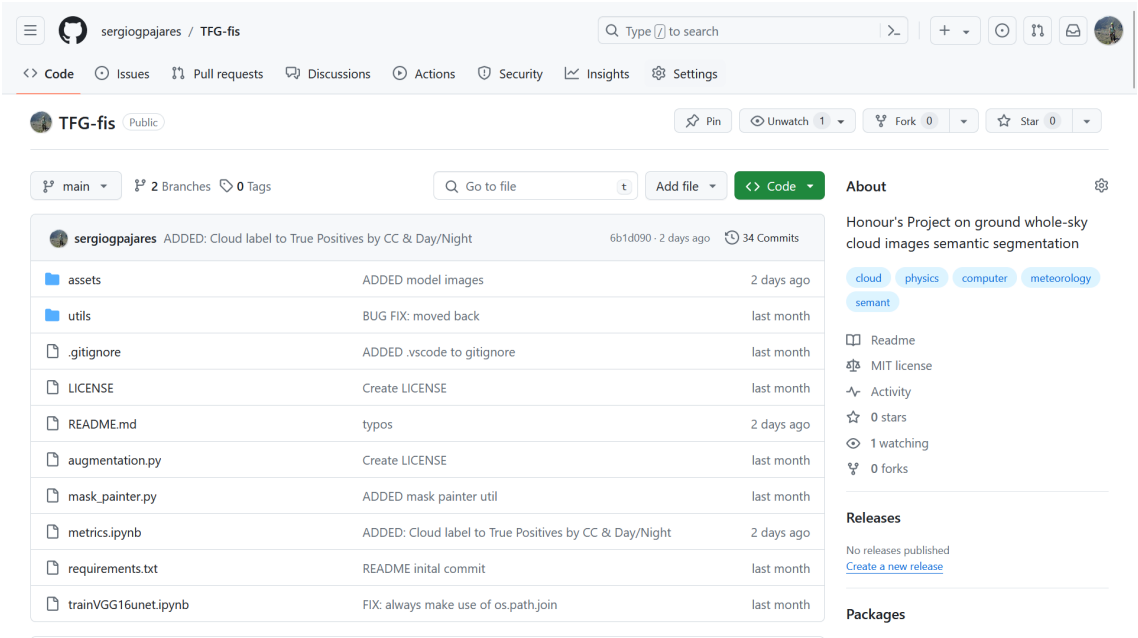


Figura A.1: Página de github.com de este trabajo.

Por otro lado, el código puede descargarse desde la propia interfaz gráfica de GitHub o mediante el software de control de versiones `git`.

La mayor parte del código viene en la forma de cuadernillos de Jupyter que están pre-ejecutados. De esta forma, se pueden ver todas las salidas del código, como muestra la figura A.2.

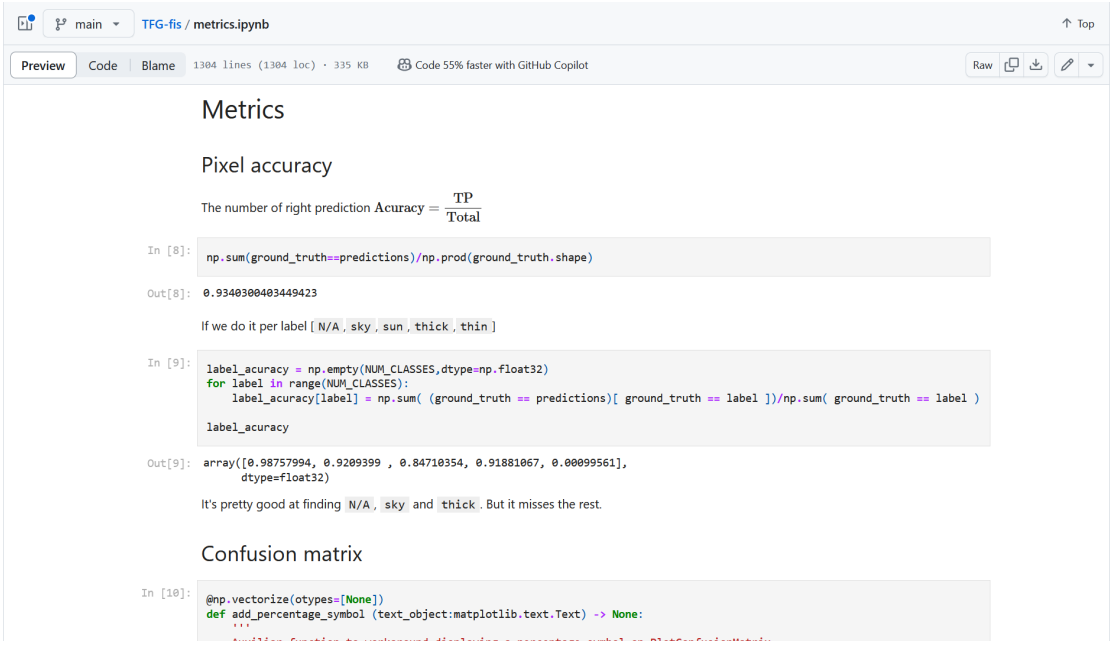


Figura A.2: Ejemplo del código en los cuadernillos de Jupyter

A.2 MODELO ENTRENADO

A diferencia de la mayoría de artículos, que no publican sus modelos finales, el modelo con su estructura de capas, función de perdida y pesos entrenados también se puede descargar desde: www.kaggle.com/models/sergiogarciapajares/vggcloudunet.

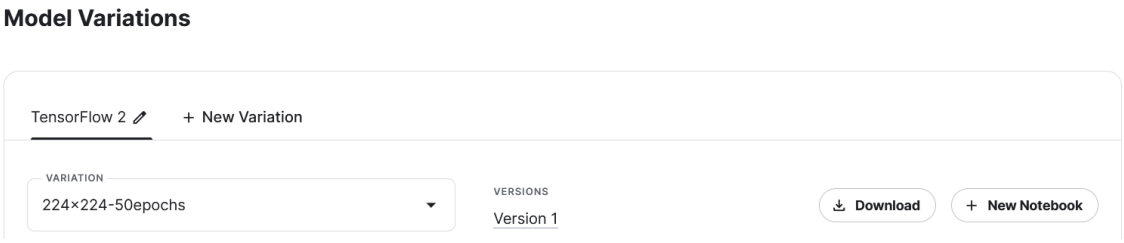


Figura A.3: Página de descarga del modelo.

El modelo está totalmente contenido en el archivo `VGG16unet-224x224-50epochs.model.keras` y tiene un peso de 106 Mb (demasiado para su inclusión en el repositorio de GitHub). Este modelo debe ser leído con la versión `TensorFlow 2.16.1`.