Exploración e implementación de estructuras avanzadas de aprendizaje automático para la segmentación y clasificación de imágenes aéreas

Santiago Adrián Yánez Martín

Supervisado por:
Adrián Peñate Sánchez
María Cristina Benlliure Jiménez

Julio 2023

Agradecimientos

A mi familia que me ha apoyado en todo momento, y a mis tutores Adrián y Cristina que me han guiado durante el desarrollo de este TFG.

Resumen

Este trabajo de fin de grado tiene como objetivo llevar a cabo el montaje de un sistema de segmentación semántica de imágenes aéreas.

Para ello, se llevará a cabo un análisis de los distintos modelos y arquitecturas para la segmentación semántica de imágenes y se usarán dos datasets para hacer los entrenamientos oportunos utilizando dos de las arquitecturas disponibles. Se compararán los resultados obtenidos y se decidirá que arquitectura elegir.

Se realizará un *transfer learning*, y se analizarán los resultados obtenidos, con el objetivo de determinar la efectividad del sistema de segmentación semántica. Finalmente, se utilizarán imágenes no etiquetadas, para valorar como segmenta el modelo en imágenes de las que no se tiene un etiquetado previo.

Abstract

This final degree project aims to carry out the assembly of a semantic segmentation system for aerial images.

For this, an analysis of the different models and architectures for the semantic segmentation of images will be carried out and two datasets will be used to perform the appropriate training using two of the available architectures. The results obtained will be compared and it will be decided which architecture to choose.

A *transfer learning* will be carried out, and the results obtained will be analyzed, in order to determine the effectiveness of the semantic segmentation system. Finally, unlabeled images will be used to assess how the model segments images that have not been previously labelled.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

Actualmente estamos viviendo una época en la que la inteligencia artificial se está abriendo camino y acentuando cada vez más en nuestras vidas, aunque a veces ésta puede incluso pasar totalmente desapercibida, bien porque están implementadas en aspectos básicos de nuestra vida como puede ser un filtrado de spam que puede utilizar nuestro cliente de correo electrónico para identificar de una manera automática correos electrónicos los cuales podemos considerar como spam, o bien porque al tenerlo tan asumido se olvida que hay detrás de ello, como puede ser un asistente de voz que tengamos en nuestras casas o incluso en nuestro dispositivo móvil.

Se podrían nombrar numerosos ejemplos en los que se haga uso de inteligencia artificial, pero al final todos convergen a un mismo fin, y es intentar de la mejor manera imitar al cerebro humano aprendiendo de la experiencia. Si vamos a una definición más formal de inteligencia artificial obtenemos la siguiente definición "Disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana, como el aprendizaje o el razonamiento lógico.", según la RAE [22].

Por ello, en este contexto de creciente presencia de la inteligencia artificial en nuestra vida cotidiana, resulta de gran importancia la exploración e implementación de estructuras avanzadas de aprendizaje automático para la segmentación y clasificación de imágenes aéreas, tema central de este trabajo.

La segmentación de imágenes corresponde en su definición más simple a asociar a cada píxel de la imagen la clase a la que corresponda, entiéndase como clase a tipo de objeto, individuo, animal u cosa, que se encuentra en la imagen. Se trata, por tanto, de una tarea no trivial que requiere muchísimas horas de análisis de imágenes por parte de personas físicas. Además, esta no trivialidad aparece cuando las imágenes son complejas en las que se encuentran multitud de elementos distintos en la misma, o cuando la imagen no tiene una calidad lo bastante buena como para identificar píxel a píxel a que clase corresponde.

Por ello, el aprendizaje automático es una de esas herramientas que hace que esta tarea la cual dispone de mucho tiempo físico pase a convertirse a una tarea totalmente automática de cara a la persona que haga uso de este.

La segmentación de imágenes se utiliza en una gran variedad de profesiones distintas, desde la construcción y arquitectos/as que hacen uso de segmentación de imágenes para analizar daños y/o defectos que puedan existir en un edificio, pasando por agricultores/as que necesitan identificar de alguna forma la existencia de anomalías en sus cultivos, y terminando en el área de medicina, en lo que la segmentación puede llegar incluso a salvar vidas detectando con las mismas tumores o lesiones que desde los ojos de un/a médico/a puede pasar desapercibido o tener dudas al respecto. Como se ha observado, la segmentación de imágenes no se trata de una utilidad que hace uso de inteligencia artificial y ya está, sino que se trata de un área en la que está presente en multitud de profesiones de forma directa o indirectamente.

En este trabajo de fin de grado, se llevará a cabo la exploración para una posterior implementación de estructuras avanzadas de aprendizaje automático para la segmentación y clasificación de imágenes aéreas.

1.1. Aportaciones

Tal y como se había nombrado anteriormente, la segmentación de imágenes es una utilidad que hace uso de la inteligencia artificial y que cada vez está en mayor crecimiento. Esto lo podemos corroborar observando la ilustración 1.1, en la que se compara los ingresos globales de la inteligencia artificial desde 2016 a 2025 en millones de euros. Se comprueba como en primer puesto de ingreso global se encuentra el reconocimiento estático de imagen, clasificación y etiquetado.

Todo esto al final también se ve reflejado en el PIB (Producto Interior Bruto) de cada país, ya sobre el año 2016 se realizaban estimaciones en las que se apuntaba que en el año 2036 en países desarrollados la mitad del PIB del país estaría relacionado con la inteligencia artificial tal y como se refleja en la ilustración 1.2. En dicha ilustración, se observa como existe un mayor crecimiento del PIB en todos los países en donde hacen uso de IA (Inteligencia Artificial). Entre los países más destacables de este crecimiento están, Alemania el cual el uso de IA duplica el PIB, y Japón que lo triplica. En España, aunque el crecimiento es algo menor, la IA todavía contribuye con un aumento de casi 0,75 puntos en el PIB en comparación con su ausencia.

En resumen, el uso de la inteligencia artificial en diferentes aplicaciones, como la segmentación de imágenes, está en constante crecimiento y se espera que tenga un impacto significativo en la economía de los países en los próximos años. La ilustración de ingresos globales de la IA muestra cómo el reconocimiento estático de imágenes, clasificación y etiquetado son las aplicaciones más rentables en este momento, justamente de lo que parte este trabajo de fin de grado. Además, la estimación del PIB en relación con la IA sugiere que su uso puede aumentar el crecimiento económico de los países, y que los países más avanzados en la adopción de la IA, como Alemania y Japón, tienen y tendrán mayores beneficios económicos.

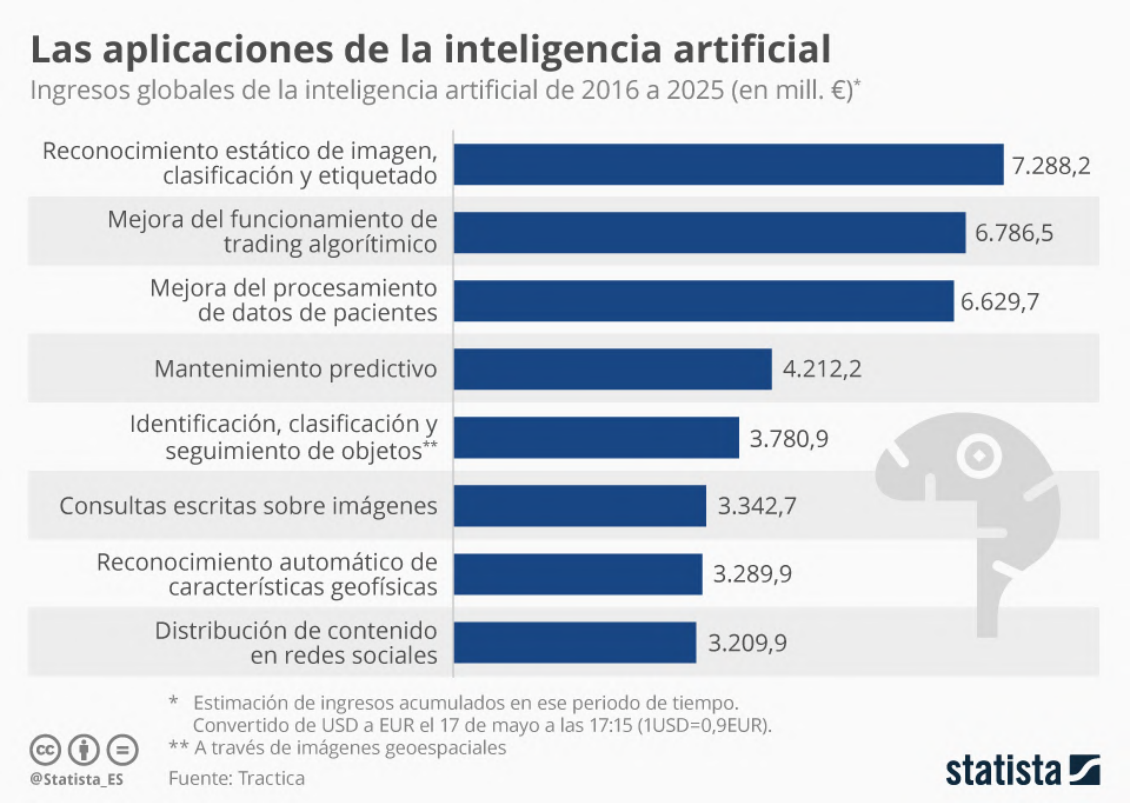

Ilustración 1.1: Gráfica de las aplicaciones de la inteligencia artificial según *Statista*. Fuente: [25]

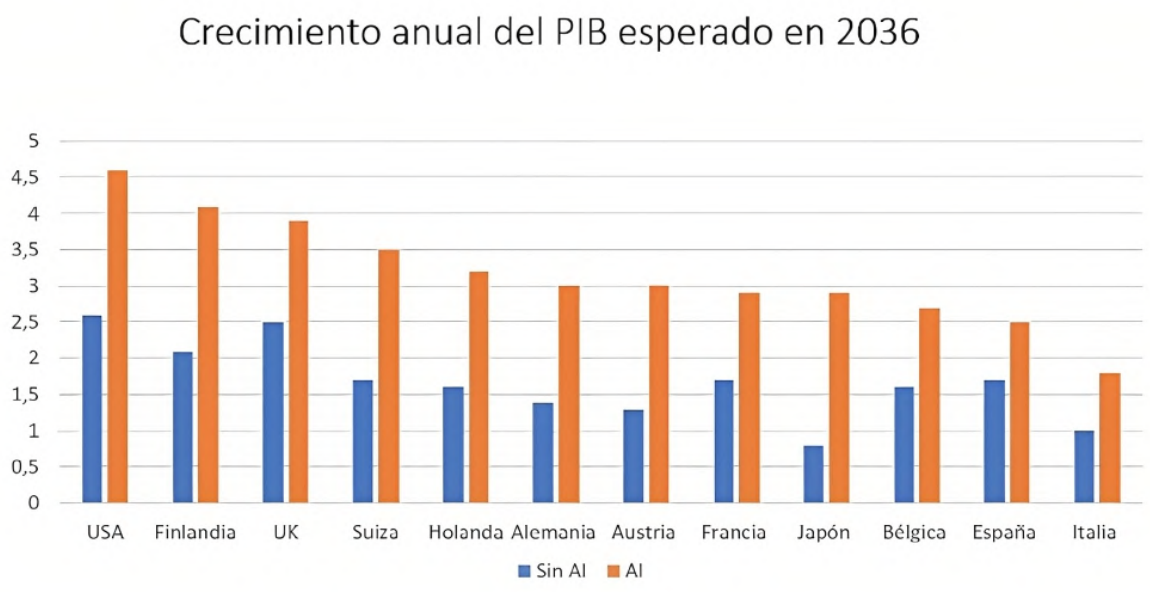

Ilustración 1.2: Estimación del PIB para el año 2036 con y sin IA. Fuente: [1]

1.2. Motivación inicial

La motivación de este presente trabajo de fin de grado partió a partir de ver en la bolsa de empleo de la Escuela de Ingeniería Informática de la ULPGC, un trabajo el cual me llamó bastante la atención, ya que se enfocaba en la utilización de estructuras de aprendizaje profundo para la segmentación de imágenes, temática que me gusta por la gran aplicación que tiene esto en la vida real.

Por ello, se contactó con el profesor responsable de dicha oferta, Dº Adrián Peñate Sánchez, y después de hablarlo se llegó a una ligera modificación del mismo, lo que hiciera que me entusiasmara más, puesto que ahora la segmentación de imágenes que se realizaría serían sobre imágenes aéreas.

Por otra parte, el aprendizaje y la utilización de herramientas y tecnologías con las que no he tenido experiencia previa en la carrera resulta especialmente motivador, ya que me permite ampliar mis habilidades y conocimientos para mi futura carrera profesional.

Por todo ello, este trabajo de fin de grado es una valiosa oportunidad para adquirir experiencia y prepararme para futuros desafíos laborales en el campo de la informática y más concretamente en el campo de la inteligencia artificial.

1.3. Redes neuronales

Las redes neuronales son una creación artificial puramente científica creada con la intención de que sea un tipo de modelo de aprendizaje automático que de alguna manera intente parecerse al propio funcionamiento del cerebro humano.

Se entiende como aprendizaje automático (machine learning) la capacidad que posee por ejemplo un equipo informático para que aprenda de manera automática sin ser programada manualmente para realizar la actividad propuesta, sino que con el conocimiento que este posee sea capaz de identificar el problema y solventarlo. En este sentido, el aprendizaje automático se puede dividir en distintos tipos:

- **Aprendizaje supervisado**. En este tipo de aprendizaje, se hace uso un conjunto de datos etiquetados para entrenar la red neuronal. De esta forma, los datos se van introduciendo en el modelo, y este va ajustando sus pesos para que el modelo tenga una mayor precisión. [12]
- **Aprendizaje no supervisado**. En este caso, lo que se trata es de analizar y agrupar conjuntos de datos sin etiquetar. Para ello, se van obteniendo patrones y características de forma que no haya una intervención humana directa. [11]
- **Aprendizaje semi-supervisado**. En este tipo de aprendizaje, se hace uso de un conjunto de datos de entrenamiento con etiquetado y sin etiquetado, de esta forma. se trata del tipo de aprendizaje intermedio entre aprendizaje supervisado y aprendizaje no supervisado. [5]

- **Aprendizaje por refuerzo**. Este tipo de aprendizaje se basa en prueba y error, por ello a diferencia de los tipos anteriores no es necesario una gran cantidad de datos. [15]

1.3.1. Neurona

Para entender bien el significado de una red neuronal, hay que tener claro el significado de neurona, para ello, hay que dirigirse a las definiciones que da la ciencia de la biología para ello.

1.3.1.1. Neurona biológica

Una neurona se trata de una célula, que en conjunto con aproximadamente otras 100.000 millones de neuronas son capaces de hacer controlar todas la funciones tanto voluntarias como involuntarias de los seres vivos. Por ello, las neuronas poseen tres funciones principales, que son: recibir, procesar y transmitir la información que les llega. Esto es posible realizarse a la propia anatomía y funcionamiento de la neurona [16], en la ilustración 1.3 se puede observar como se encuentra constituida la misma:

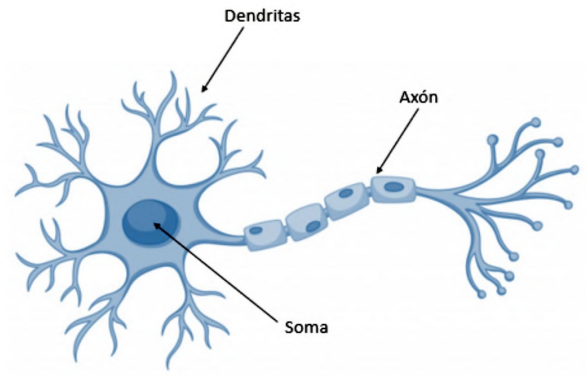

Ilustración 1.3: Partes básicas de una neurona biológica. Fuente: [27]

La neurona posee muchas partes, pero para entender como funciona una red neuronal artificial, es suficiente con las partes que se muestran en la ilustración 1.3. Una neurona está formada por la soma, que se trata del cuerpo de la neurona y es donde además se aloja el núcleo de la misma tal y como se está indicando en la ilustración, las dendritas son las encargadas de captar la información, mientras que el axón es el encargado de transmitir la información que ha sido obtenida por la dendritas [16]. Esta es, de una manera muy básica pero suficiente, las partes por las que está constituida una neurona.

Luego, una red neuronal, no es más que la unión de millones de neuronas entrelazadas unas con otras en las que cada una recibe una información de una neurona, la procesa y la transmite a la siguiente neurona.

1.3.1.2. Neurona artificial

Teniendo esta idea de como funciona una neurona, ahora se va a proceder a explicar el concepto de neurona artificial.

El funcionamiento de una neurona artificial se basa en gran medida al funcionamiento de una neurona biológica tal y como se comentó anteriormente, en esta ocasión una neurona artificial se puede representar tal y como aparece en la ilustración 1.4 en la que posee al igual que la neurona biológica distintas partes.

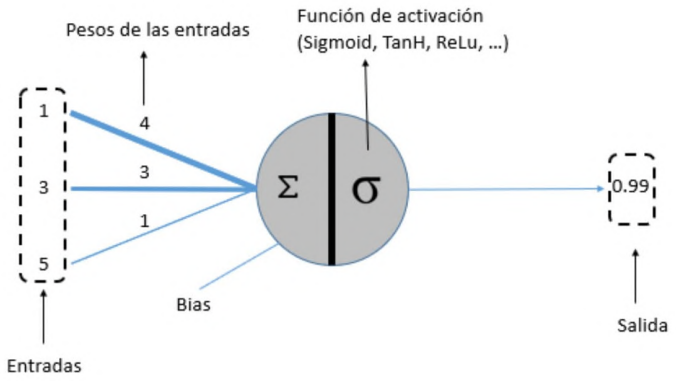

Ilustración 1.4: Partes de una neurona artificial. Fuente: [26]

Las entradas de la neurona artificial, corresponden a las salidas proporcionadas por otras neuronas, ya que las neuronas artificiales también estarán conectadas entre sí, esto extrapolado con la neurona biológica que se presentó anteriormente, correspondería con las dendritas. Luego, como se observa, cada entrada posee un peso, este peso es modificado automáticamente en el entrenamiento y son los responsables de que una neurona tenga una determinada función. El parámetro bias, corresponde a un término constante que se suma a la entrada ponderada, es decir, a la suma de las entradas por su peso, de la neurona antes de aplicar la función de activación. Seguidamente se encuentra la función de activación que se nombró anteriormente, la función de activación es la encargada de devolver un valor a partir de los datos de entrada.

Las funciones de activación se pueden dividir en dos grandes bloques: función lineal, y funciones no lineales. La función lineal se refiere a la función identidad, que se basa en la regresión lineal, por lo que de esta manera, el valor de salida será único, en la ilustración 1.5 se observa como es la gráfica de una función lineal. Mientras que las funciones no lineales permiten a las redes neuronales aprender relaciones no lineales entre las entradas y las salidas, de esta manera puede modelar relaciones más complejas y sofisticadas que no se pueden representar mediante una función de activación lineal.

En este sentido, existen distintas funciones no lineales, a continuación se nombrarán algunas de ellas [10]:

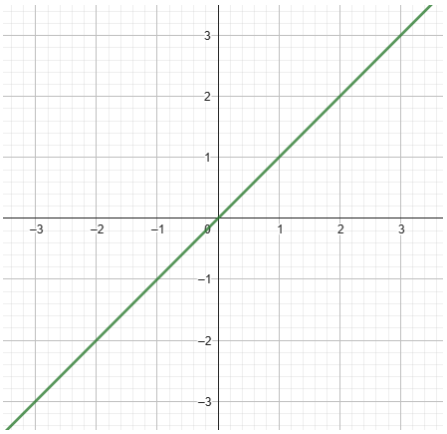

Ilustración 1.5: Función lineal.

- **Función sigmoide**. Se trata de una función matemática que toma como entrada cualquier valor real y como salida produce o un 0 o un 1. Esta función se puede definir por tanto como $f(x) = \dfrac{1}{1+e^{-x}}$, donde x es el valor de entrada a la función. En la ilustración 1.6 se observa una representación de esta función.

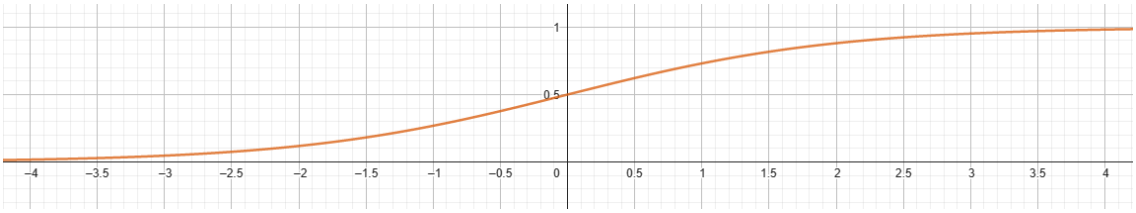

Ilustración 1.6: Función Sigmoide.

- **Función softmax**. En este caso, toma un vector de entrada con números reales y lo convierte en una distribución de probabilidad, donde cada elemento del vector se interpreta como la probabilidad de que la entrada pertenezca a una de las clases. Su función por tanto es $\sigma(z_i) = \dfrac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K$, en donde z_i es el valor de entrada de la i-ésima neurona de la capa de salida, y siendo K el número de neuronas que se existen en la capa de salida. En la ilustración 1.7 se observa la representación de la función.

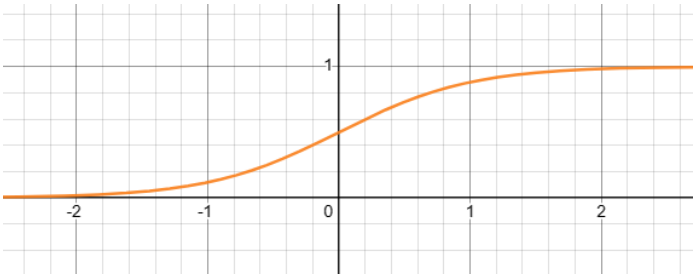

Ilustración 1.7: Función Softmax.

- **Función Relu**. Se trata de una función muy sencilla que lo que realiza es devolver un 0 si el valor de entrada es negativo, y en caso de que el valor de entrada sea positivo devolverá dicho valor. Por ello, la expresión de dicha función es $Relu(z) = max(0, z)$ donde z es el valor de entrada. En la ilustración 1.8 se observa la representación de la función Relu.

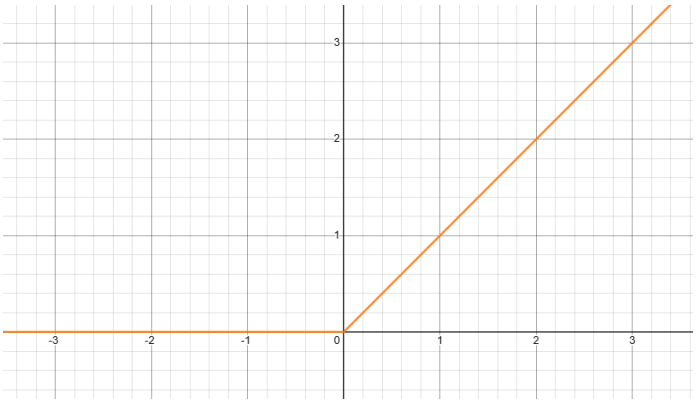

Ilustración 1.8: Función Relu.

- **Función Tanh**. Se trata de una función que devuelve valores entre -1 y 1, devolverá un valor negativo si el valor de entrada lo es también, 0 en caso de que el valor de entrada se encuentre muy cerca de él, o un valor positivo si el valor de entrada también lo es. La expresión de esta función es $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$, en donde x es el valor de entrada. En la ilustración 1.9 se observa la representación de dicha expresión.

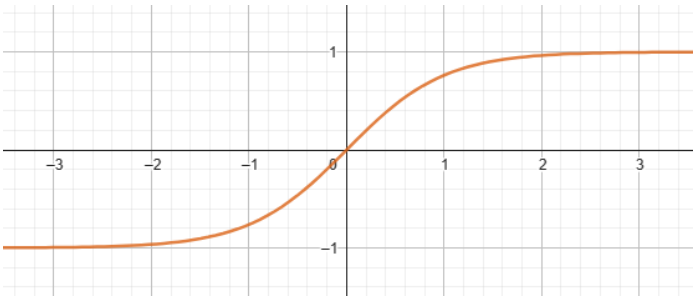

Ilustración 1.9: Función Tanh.

1.3.2. Capas de redes neuronales

Tal y como se ha explicado en la sección anterior, las neuronas artificiales se encuentran conectadas entre sí dando lugar a una red neuronal, y estas capas se distribuyen de manera que se da lugar a un nuevo término denominado capas. Estas capas, reciben diversos nombres según en donde se encuentre en la red neuronal, en la ilustración 1.10 se observa un esquema básico en el que se puede observar como se encuentra por lo general distribuidas las neuronas en capas.

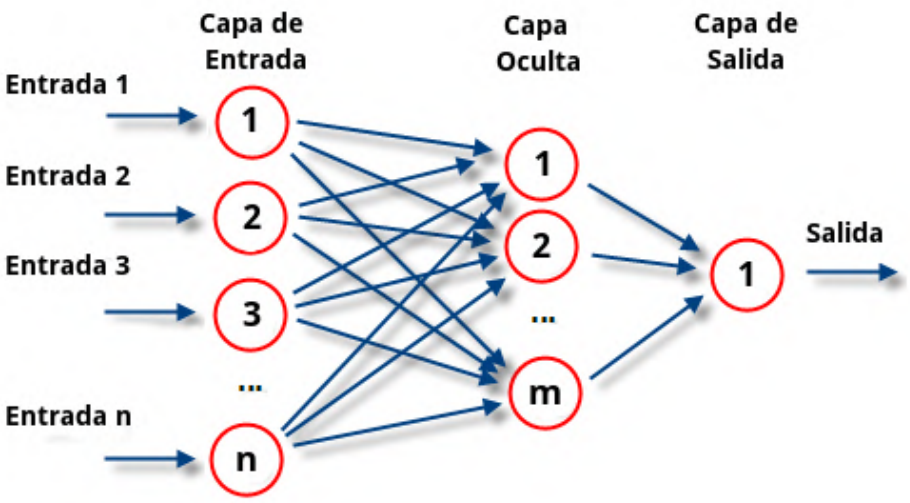

Ilustración 1.10: Partes de una red neuronal artificial. Fuente: [4]

Capítulo 2

Estado del arte

En el presente capítulo se realizará un análisis sobre los distintos estudios que se han realizado acerca de la inteligencia artificial, y más concretamente con la segmentación de imágenes, que se trata del tema principal de este trabajo.

Por ello, lo primero que hay que tener en mente es que existen multitud de tipos de segmentación, algunos de ellos se muestran en el cuadro 2.1.

Cuadro 2.1: Tipos de segmentación.

Tipo	Uso
Segmentación de imagen	Consiste en dividir una imagen en regiones o segmentos con características similares, con el fin de facilitar su análisis y procesamiento posterior.
Segmentación semántica	Consiste en asignar a cada píxel de una imagen una etiqueta/máscara que identifica el objeto o la clase a la que pertenece.
Segmentación de instancias	En este caso, se segmentan los objetos de una imagen y se les asigna un número de instancia único para diferenciarlos entre sí, permitiendo diferenciar individualmente cada objeto aún perteneciendo a la misma clase.
Segmentación de vídeo	Partiendo de un vídeo, se obtienen sus fotogramas y se segmentan para identificar y analizar los distintos objetos que aparezcan, así como el plano en el que se encuentran.
Segmentación de texto	Consiste en dividir un texto en párrafos, frases o palabras, facilitando así la obtención de un análisis, la idea principal del texto o incluso el análisis morfosintáctico.
Segmentación de audio	Una señal de audio se divide en fragmentos y se obtienen diferentes sonidos, como música o ruido de fondo, entre otros.

En el caso que nos concierne, en este TFG la segmentación de la que se llevará estudio es de la segmentación semántica, tal y como se había adelantado en la introducción.

Dentro de la segmentación semántica también podremos encontrar distintas arquitecturas que podemos hacer uso. En el cuadro 2.2 se observan los tipos de arquitecturas más conocidas y que más se usan, además, se recoge una breve descripción y el año de publicación del *paper* original.

Cuadro 2.2: Tipos de arquitecturas.

Arquitectura	Año	Descripción
Fully Convolutional Networks (FCN)	2014	Se trata de las primeras arquitecturas que se crearon y consiste en usar capas conectadas localmente.
DeepLabV1	2014	Esta arquitectura utiliza una combinación de capas convolucionales y de agrupamiento por píxel para mejorar la precisión de la segmentación.
SegNet	2015	Es una arquitectura de codificador-decodificador para la segmentación semántica. Utiliza una red convolucional profunda para codificar la imagen de entrada y una red convolucional inversa para reconstruir la segmentación de la imagen.
U-Net	2015	Se trata una arquitectura de redes neuronales profundas para la segmentación semántica que hace uso de una arquitectura de codificador-decodificador con conexiones residuales para mejorar de esta manera la precisión de la segmentación.
DeepLabV2	2016	Es una mejora de DeepLabV1, en el que sigue utilizando redes convolucionales profundas, agregando en esta nueva versión convoluciones dilatadas y empleando técnicas de agrupación de píxel, reduciendo así el tamaño del kernel.
RefineNet	2016	Se trata de una arquitectura que utiliza una arquitectura que hace uso de múltiples vías de procesamiento y fusiona características en diferentes niveles de resolución para obtener predicciones más precisas. De esta forma, las capas más profundas pueden refinarse directamente utilizando características de convoluciones anteriores.
PSPNet	2016	Esta arquitectura hace uso de una pirámide de agrupación por píxel para capturar características de diferentes escalas y mejorar la precisión con un codificador. Para ello, utiliza una combinación de convoluciones con diferentes tamaños de kernel y agrupaciones de píxeles con diferentes escalas.

Cuadro 2.2: Tipos de arquitecturas.

Arquitectura	Año	Descripción
DeepLabV3	2017	Es la versión siguiente a DeepLabV2, y en este caso también hace uso de convoluciones dilatadas, además de *atrous spatial pyramid pooling* para combinar características a múltiples escalas.
PANet	2018	Se trata de una arquitectura que utiliza una red convolucional con un extractor de características a diferentes resoluciones. Esta arquitectura captura detalles con múltiples escalas de imagen.
HRNet	2019	Es una arquitectura la cual se distingue por mantener la resolución de las características a través de la red, lo que le permite capturar detalles finos y mejorar el rendimiento en tareas de alta resolución.

En cuanto a las arquitecturas nombradas en el cuadro 2.2 son algunas de las arquitecturas que existen, unas son mejores que otras pero esto dependerá mucho del tipo de segmentación que se está realizando al igual que la naturaleza de los datos con los que se están trabajando. La segmentación semántica no es más que un conjunto de técnicas las cuales lo que tratan de realizar es dividir en zonas las imágenes de entrada con la finalidad de atribuirle a cada zona un significado semántico, en este caso, se estará atribuyendo a cada píxel de la imagen un significado semántico, que será el identificador de la clase a la que pertenecerá.

En estos últimos años, se ha incrementado el desarrollo de distintos métodos para abordar el problema de la segmentación semántica, desde enfoques basados en redes neuronales convolucionales tal y como se hará uso para el desarrollo de este TFG, hasta técnicas de procesamiento de grafos.

Actualmente, observando desde una perspectiva general lo que se está tratando de realizar es por una parte mejorar la precisión, aunque esto siempre se ha tratado de hacer. Mientras que por otra parte, se busca mejorar la eficiencia de los algoritmos ya presentes, un buen ejemplo de esto es la arquitectura DeepLabV3 de la cual se hará uso en este trabajo y consiste en una versión mejorada de la arquitectura anterior con el mismo nombre. También la adaptación a diferentes tipos de imágenes, y la integración de la segmentación semántica en aplicaciones prácticas como la conducción autónoma o la realidad aumentada. Estos son algunos de los campos de manera generalizada en los que se encuentra actualmente la segmentación semántica. Por ello, la segmentación semántica sigue siendo un área de investigación activa y prometedora en el campo de la visión por computadora.

Como se mencionó anteriormente, las arquitecturas y redes actuales están siendo mejoradas y entrenadas utilizando una mayor cantidad de imágenes a la vez que ampliando la variedad de escenarios. Por lo general, los conjuntos de datos iniciales para el entrenamiento de una red neuronal son los siguientes:

- **COCO**. Se trata de uno de los datasets más usados en la actualidad, y es que se trata de un dataset de más de 300.000 imágenes y la cantidad de imágenes etiquetadas

asciende a más de 200.000, con un total de 80 clases distintas, es por esta última razón por la que este dataset lo hace tan práctico su uso como red pre-entrenada. Fuente: [7]

- **Pascal VOC**. Se trata de un dataset más pequeño que COCO, aún así, se trata de un dataset que es muy utilizado, en esta ocasión, este dataset posee en su versión de 2012 más de 11.300 imágenes de entrenamiento, siendo capaz de identificar hasta 20 clases distintas. Fuente: [21]
- **Cityscapes**. Se trata de un dataset que cuenta con 25.000 imágenes etiquetadas, siendo capaz de identificar hasta 30 clases distintas. Fuente: [6]
- **ADE20K**. Se trata de un dataset que posee más de 25.000 imágenes tanto de exteriores como de interiores, esta es una de las peculiaridades de este dataset. Además, este dataset posee etiquetado de 150 clases distintas. Fuente: [2]
- **ImageNet**.Este dataset cuenta con más de 1.000.000 de imágenes y con etiquetado de 1000 clases distintas. Por esta razón, este dataset es frecuentemente utilizado como dataset del que se parte para realizar un *transfer learning*, debido a que contiene mucha información que se puede escalar al dataset al que se le va a realizar el *transfer learning*. Fuente: [13]

Uno de los proyectos más recientes, ha sido publicado en el mes de abril, y es una de las investigaciones que ha estado llevando a cabo Meta. Se trata de un modelo de segmentación de imagen llamado SAM (*Segment Anything Model*) cuyo objetivo según afirma la propia compañía es democratizar la segmentación. La funcionalidad de este proyecto es cortar cualquier tipo de objeto en una imagen con un solo clic en la misma. Para ello, tan solo basta con pasarle la imagen con la que se desea trabajar, y una vez realizado el proceso que ellos mismos denominan extracción de la incrustación para la imagen, que no es más que obtener una representación numérica de la imagen, para que finalmente ya se puede obtener cualquier objeto que aparezca en la imagen y discretizarlo de manera que solo se pueda obtener el mismo.

Para ello, Meta ha entrenado este modelo con un dataset llamado SA-1B Dataset, que consta con 11 millones de imágenes y con un número de máscaras que asciende a más de mil millones. Esto lo hace uno de los conjuntos más grandes con imágenes etiquetadas. En la ilustración 2.1 se observa una gráfica desarrollada por los mismos creadores del dataset SA-1B en el que comparan la cantidad de imágenes existentes en ese dataset con respecto a otros datasets. En el que se aprecia como tiene 6 veces más de imágenes que el dataset OpenImages V5, este último se trata de un dataset desarrollado por Google, y se trata de una versión anterior a OpenImages V7, que es más grande que la versión 5, pero aún así, SA-1B posee una cantidad mayor de imágenes. Fuente: [24].

A su vez, lo mismo ocurre con las imágenes etiquetadas, es decir, las máscaras, el dataset SA-1B posee 400 veces más de máscaras que el dataset OpenImages V5, como se observa en la ilustración 2.2 también desarrollada por los mismos autores del dataset. De esta manera, se posiciona en uno de los mayores datasets que se puede usar a día de hoy. Fuente: [24].

La estructura que utiliza SAM, es un codificador de imágenes ViT-H y fue presentado por

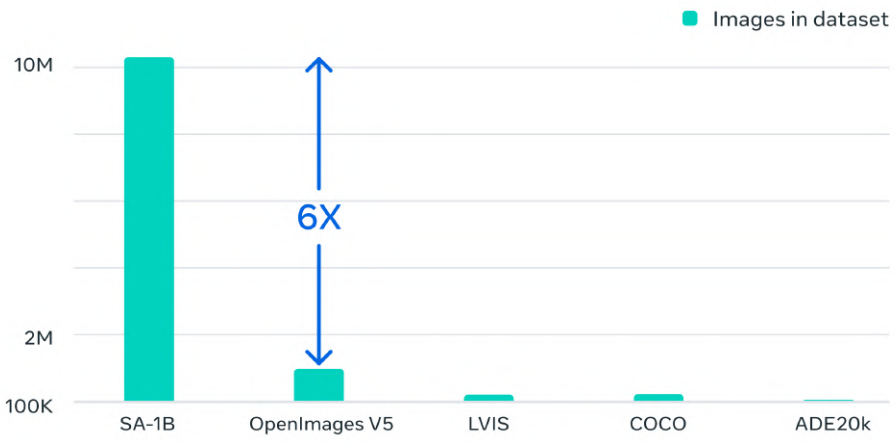

Ilustración 2.1: Comparativa de cantidad de imágenes del dataset SA-1B en comparación con otros datasets. Fuente: [24]

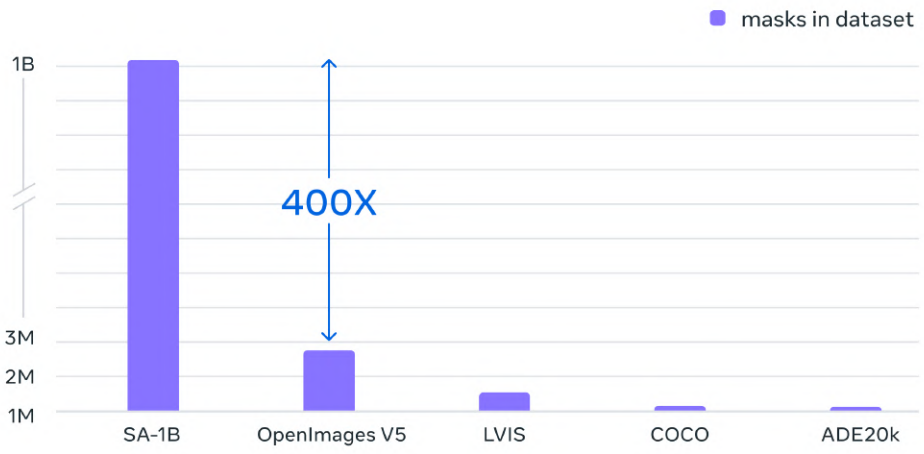

Ilustración 2.2: Comparativa de cantidad de máscaras del dataset SA-1B en comparación con otros datasets. Fuente: [24]

el equipo de Google Brain. El modelo ViT-H utiliza una arquitectura de Transformer para procesar imágenes, en este caso no utiliza convoluciones como lo hace normalmente las redes neuronales, sino que el modelo ViT divide la imagen en distintos parches y los convierte en secuencias de tokens que se procesan utilizando capas de Transformer. Fuente: [20]

Capítulo 3

Objetivos

El objetivo principal de este presente trabajo de fin de grado es utilizar un modelo de segmentación semántica, el cual dada una imagen de entrada, sea capaz de devolver la imagen segmentada identificando las distintas clases existentes.

Este trabajo posee varios objetivos, para alcanzarlos, será necesario llevar a cabo varias etapas para culminar con un modelo entrenado y que sea capaz de segmentar imágenes aéreas. Así pues, los pasos que se han realizado a lo largo de este TFG son los siguientes:

- **Análisis de modelos existentes**. En esta primera etapa del desarrollo, lo que se realizará será un análisis exhaustivo de los distintos modelos existentes y más concretamente con las arquitecturas que se usarán para entrenar el modelo. Este análisis ha permitido identificar las fortalezas y debilidades de distintos modelos y que ya se encuentran desarrollados por terceras personas.

  La idea principal ha sido obtener datasets y observar cómo se encuentran organizados, se ha probado código existente y se han analizado los resultados obtenidos para comprobar la precisión y la intersección de los píxeles. Además se ha familiarizado con el entorno y con la herramienta Pytorch, que será la herramienta que se utilizará para el desarrollo de este TFG.

  Esta primera parte ha sido de gran utilidad para seleccionar el modelo más adecuado acorde al propósito específico de este trabajo, como a su vez la obtención y el análisis de los datasets que serán usados.

- **Entrenamiento y testeo de las redes**. Teniendo ya seleccionado el modelo más adecuado a las características de segmentación que se está realizando y los datasets que se usarán para ello, se procederá a realizar la fase de entrenamiento y testeo de diversas arquitecturas de redes neuronales. Se utilizarán distintos datasets que serán con los que se entrene la red, en este desarrollo del TFG se elegirán dos arquitecturas que serán entrenadas. Posteriormente, se testeará los resultados de las redes para cada dataset con el objetivo de saber cuál es la mejor configuración para nuestra red.

- **Validación de los resultados obtenidos**. Una vez que se hayan obtenido los modelos

resultados del entrenamiento de las dos arquitecturas elegidas con los dos datasets, se procederá a llevar a cabo un exhaustivo análisis sobre los resultados obtenidos. En esta etapa se pretende elegir una de las dos arquitecturas entrenadas, valorando los resultados obtenidos.

- **Realización del *transfer learning***. Teniendo ya los modelos entrenados con la arquitectura seleccionada, se procederá a la realización del *transfer learning*, con esto lo que se pretender realizar es que el aprendizaje parta de uno de los modelos ya entrenados con uno de los datasets, con ello se intentará mejorar la precisión de cada una de las clases dando lugar a un modelo más preciso y robusto.
- **Valoración y conclusión de los resultados**. Una vez que se hayan obtenido los resultados con las imágenes no etiquetadas se realizará una valoración exhaustiva de los resultados obtenidos. A su vez, se compararán los resultados con los valores obtenidos del conjunto de testeo de los datasets, para de esta manera, teniendo en cuenta que son imágenes no etiquetas tener un punto de referencia con el que comparar lo tan buena o no que es la segmentación semántica que ha realizado el modelo. Se concluirá con un análisis sobre el modelo de segmentación semántica en el que se ha usado *transfer learning*, usando para ello los resultados que se han obtenido con los modelos entrenados por separados.

Capítulo 4

Herramientas usadas

En esta sección, se nombrarán las distintas tecnologías que se han usado en este TFG, además se comentarán los distintos paquetes y librerías que se han usado para el desarrollo del mismo.

4.1. Tecnologías usadas

Para la realización de este trabajo de fin de grado se han usado distintas tecnologías, utilizando como lenguaje de programación Python. Para ello, primero se ha comenzado utilizando el cuaderno de Google Colaboratory el cual ofrece una serie de GPUs gratuitas, esta tecnología se ha utilizado en las primeras semanas de trabajo con el TFG cuando se encontraba buscando distintas arquitecturas y se hallaban algunas arquitecturas con ejemplos, de esta manera se podía probar el código proporcionado de una manera rápida y eficaz.

Ya habiendo decidido las dos arquitecturas que se querían comparar para llevar a cabo la segmentación semántica de las imágenes aéreas, se ha procedido a utilizar un entorno local, puesto que aunque Google Colab proporciona un entorno en la nube, el uso de GPU es bastante limitado. Por ello, se ha obtenido un dispositivo con una GPU propia, para de esta manera llevar a cabo los entrenamientos pertinentes. En esta parte, se ha usado Jupyter Notebook, esta herramienta se trata de un software libre el cual permite la creación y compartición de documentos interactivos en los que se puede combinar código, texto y elementos multimedia como imágenes. Esto último es lo que lo hace una herramienta sumamente interesante para la realización de este presente trabajo, y es que se pueden observar en el mismo entorno tanto gráficos como imágenes, pudiendo así, apreciar las predicciones del propio modelo entrenado, tal y como se irá observando a lo largo de este trabajo.

La instalación de Jupyter Notebook se ha realizado dentro de Anaconda, puesto que Anaconda entre otras funcionalidades posee el uso de *environments*. De esta manera, se pueden crear tantos entornos funcionales de Python como sea necesario.

4.2. Paquetes y librerías usadas

Tal y como se había adelantado anteriormente, el desarrollo de este TFG se ha realizado con Pytorch. Pytorch está basado principalmente en la biblioteca de Python llamada Torch que está dirigida a visión por computador. Se ha decidió utilizar dicha biblioteca puesto que el resto de compañeros que estaban realizando el TFG con los mismos tutores que yo, usaban dicha biblioteca, de esta manera, en mi caso, al usarla, se creó una pequeña comunidad de compañeros en la que la idea principal era intercambiar conocimiento acerca de la biblioteca, así podíamos ayudarnos los unos a los otros y por tanto existía una unificación de herramientas.

Por esta razón, se ha utilizado diversos paquetes de Pytorch como lo son torch.utils.data del que se importó la función Dataset, esta función permite acceder a los datos de una manera eficiente. Del paquete llamado torchvision, se importó models, para poder obtener los modelos preentrenados para usarlos en el desarrollo del TFG. Además, se ha importado el paquete torch.nn, para poder modificar las últimas de las capas de los modelos para que dispongan de las salidas necesarias. Finalmente, de los paquetes, se ha usado torch.optim, de este paquete se ha importado Adam, que será el optimizador que se usará para el entrenamiento de las arquitecturas con los distintos datasets.

Además de los paquetes de Pytorch, han sido necesarios otros tipos de paquetes, para la visualización de las distintas imágenes en el propio Jupyter Notebook, en esta ocasión los paquetes que se han importado han sido numpy y PIL. El primero se trata de uno de los paquetes más comunes y necesarios, en este caso se ha usado para pasar de las imágenes predichas por el modelo en formato tensor de Pytorch a formato numpy. Por otra parte, el segundo de ellos, es decir, PIL, se ha usado principalmente para el módulo Image, el cual nos permite abrir imágenes que se encuentren almacenadas de forma local. Además, para poder visualizar varias imágenes de forma simultánea es necesario importar matplotlib.pyplot.

Finalmente se han usado otras librerías, como lo es la librería OS, esta librería se ha usado para que cuando se creen los distintos datasets para el entrenamiento, las imágenes y sus correspondientes máscaras se listen de manera ordenada alfabéticamente. De esta manera, se asegura que a cada imagen le corresponde su máscara y no hay lugar a errores. Con la librería tqdm, lo que ha proporcionado es la barra de progreso de entrenamiento, con esto se tiene una idea de la demora que puede llegar a tener el actual entrenamiento. Para llevar a cabo la comparación de los modelos entrenados y poder escoger el mejor modelo, se ha usado distintas tablas, dichas tablas han sido creadas con la funciones prettytable y pandas.

Capítulo 5

Detalles de implementación

En este capítulo, se comentará brevemente algunos de los detalles de implementación que se han realizado en el código creado para el desarrollo de este TFG.

Este código se ha realizado íntegramente en Python, utilizando Pytorch como bien se indicó en la sección de herramientas utilizadas. Pytorch proporciona distintas utilidades que pueden ser usadas como herramientas para realizar un aprendizaje profundo, en el caso que se está trabajando, se hará uso de dos de sus clases, la clase Dataset, y la clase Dataloader. Haciendo uso de la primera de las clases nombradas, se ha creado una clase que herede de Dataset de manera que se inicializa con las transformaciones y con las imágenes y máscaras que se le pasan por parámetro. Esta clase además, debe de tener implementados dos métodos que posee la clase desde la que se ha heredado, el primer método lo que devuelve es la cantidad de imágenes que posee el dataset, mientras que el segundo de los métodos devuelve la imagen y su correspondiente máscara aplicadas las transformaciones necesarias en ambas.

Los distintos *batches* de entrenamiento, validación y testeo, se han obtenido al llamar a una función que crea el dataset haciendo uso de la clase nombrada anteriormente, y luego se obtiene de forma aleatoria el 80 % para imágenes de validación, y el 20 % sobrante se reparte de forma equitativa para imágenes de validación y testeo, de manera que se obtienen tres conjuntos con las imágenes de cada tipo. Finalmente, se hace uso de la clase Dataloader de Pytorch, esta clase permite cargar los datos para poder iterar fácilmente entre ellos, además de pasarle la cantidad de imágenes por *batch* que se desea que tenga.

Teniendo los tres dataloaders correspondientes para los conjuntos de entrenamiento, validación y testeo, se procede a crear el modelo, en el caso de la arquitectura que se ha elegido finalmente, DeepLabV3, el modelo se puede llamar desde la librería torchvision.models.segmentation en la que se encuentra dicha implementación. Y una vez cargado el modelo se inicializan los hiperparámetros necesarios, en este caso se ha usado como *learning rate* un valor de 1×10^{-4} y como optimizador el Adam.

Luego se definen métodos que se usarán durante el entrenamiento, entre ellas, hay que destacar una función de creación propia en la que dada una máscara ya sea del propio dataset o predicha, devuelve la máscara en colores RGB. Además, se han desarrollado dos funciones

principales que serán usadas para comprobar como está entrenando el modelo en la etapa de aprendizaje, estas funciones son las que calcula el valor de IOU (*Intersection Over Union*) y el coeficiente DICE. Finalmente, se ha creado otro método que es el encargado al finalizar cada época de evaluar el modelo utilizando el dataloader de validación, y retornando las métricas de coste de validación, la precisión de validación, el coeficiente DICE y el valor de IOU de media que posee el conjunto de validación.

Para el entrenamiento se ha establecido una parada automática que se llevará a cabo si el coste de validación no mejora en tres épocas consecutivas. En caso de que esto no ocurra, el modelo tendrá como máximo 50 épocas. En cada época se entrenarán todos los *batches* que tenga el conjunto de entrenamiento, y una vez que se entrene cada *batch*. Al finalizar cada época, se obtendrá además de los valores de coste de validación, precisión de la validación y la media de los valores de IOU, DICE y coste del conjunto de entrenamiento, la media de IOU y DICE del conjunto de entrenamiento.

Una vez que se ha finalizado el entrenamiento, se pueden ejecutar dos métodos, estos son, un método que es el encargado de dado un modelo y un conjunto de entrada, como lo es por ejemplo el conjunto de testeo, calcula la cantidad de TP, TN, FP y FN para cada imagen, para mostrar con estos resultados en forma de tabla los valores de las métricas nombradas en este TFG. Finalmente, con el último método se obtienen tres imágenes, siendo estas una imagen aleatoria del conjunto de datos de testeo, su correspondiente máscara, y la máscara que ha predicho el modelo entrenado. De esta manera, se puede tener una referencia acerca de como está segmentando el modelo en una imagen aleatoria.

Capítulo 6

Competencias específicas cubiertas

Con la realización de este TFG se han adquirido dos competencias distintas, por un lado la propia competencia del TFG, y por otra parte la competencia específica CI15 [9]. A continuación, se muestran en el cuadro las definiciones de ambas competencias y posteriormente se comenta su relación con lo que se ha realizado en este presente trabajo.

Cuadro 6.1: Competencia general y específica cubiertas.

TFG	Ejercicio original a realizar individualmente y presentar y defender ante un tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas de la Ingeniería en Informática de naturaleza profesional en el que se sinteticen e integren las competencias adquiridas en las enseñanzas.
CI15	Conocimiento y aplicación de los principios fundamentales y técnicas básicas de los sistemas inteligentes y su aplicación práctica.

Tal y como se ha observado en el cuadro 6.1, la competencia TFG se ha adquirido al realizar, presentar y defender este propio trabajo de fin de grado que se está realizando el cual está relacionado con la Ingeniería Informática, carrera a la que pertenece este trabajo de fin de grado. Mientras que con la competencia CI15, se ha cubierto la susodicha ya que este trabajo está relacionado directamente con los sistemas inteligente. Debido que la segmentación semántica de imágenes aéreas es una tarea que involucra el uso de sistemas inteligentes y la aplicación de técnicas de aprendizaje profundo tal y como se irá observando a lo largo de este trabajo.

Capítulo 7

Desarrollo

7.1. Descripción de los datasets utilizados

Para poder realizar la segmentación de imágenes aéreas con distintas arquitecturas, ha sido necesario el uso de dos datasets distintos. La idea principal que se desea llevar a cabo es partiendo de dos arquitecturas distintas para la segmentación de imágenes, entrenar la red utilizando una arquitectura y uno de los datasets. Se observan los datos que se obtienen y se vuelve a entrenar la red pero esta vez utilizando el segundo de los datasets. Una vez que obtengan estos dos resultados, se realizará una comparación acerca de las distintas clases que segmenta la red y realizado esto, se procede a rehacer nuevamente el proceso pero esta vez utilizando la segunda de las arquitecturas que se ha elegido. Se obtendrán dos modelos resultantes del entrenamiento y se llevará a cabo un estudio de esta segunda arquitectura basándonos en los resultados de ambos modelos.

Teniendo el resultado de ambas arquitecturas con ambos datasets, se llevará a cabo una exhaustiva comparación sobre cual modelo es mejor, para poder realizar un *transfer learning* y obtener así mejores resultados.

Asi pues, a continuación se explicará como se encuentran organizados ambos datasets obtenidos y que clases posee cada uno de ellos.

7.1.1. Dataset rural

Este dataset se trata de un dataset de imágenes aéreas de paisajes rurales que se presentó como un paper en la presentación oral en la 15ª conferencia asiática sobre visión artificial 2020 [19]. Este dataset está formado por un total de 20 vídeos en calidad 4K, y por un total de 1127 imágenes etiquetadas manualmente. El total de minutos de vídeos que se poseen es de 17 minutos, y las imágenes etiquetadas manualmente corresponden con cada 50 frames de cada vídeo.

Por lo que sabiendo esta información, se ha procedido a crear un pequeño script en Python

que vaya obteniendo imágenes de cada vídeo cada 50 frames, a excepción del último vídeo, que se ha observado que el etiquetado de imágenes corresponde cada 25 frames, es decir, que se ha obtenido el doble de imágenes en ese vídeo que en el resto. Se han obtenido las imágenes correspondientes a cada imagen etiquetada debido a que es necesario para la red que se va a crear tener dos imágenes, por un lado la imagen real, que será una imagen en color, que en este caso corresponde con la imagen que se está obteniendo de los vídeos, y por otra parte es necesario las imágenes etiquetadas, que estas si ya nos la dan como imagen de tipo PNG.

7.1.1.1. Estructura del dataset rural

Este dataset consta de un total de 12 clases distintas, en el cuadro 7.1 se pueden observar las distintas clases que tiene dicho dataset, al igual que el color con el que se está indicando en las máscaras (etiquetas de la imagen).

Cuadro 7.1: Representación de las clases del dataset rural.

Color	Clase
	Forest
	Residential
	Land
	Sky
	Hill
	Road
	Church
	Fence
	Water
	Car
	Person
	Haystack

Ahora se van a visualizar unas imágenes y sus correspondientes máscaras para tener una referencia acerca de como son las imágenes de este dataset. Por ello, en la ilustración 7.1 se pueden observar en el margen izquierdo cuatro de las imágenes de dicho dataset, en la parte central se muestran sus correspondientes máscaras, y finalmente en el margen derecho se muestra la superposición de la máscara sobre la imagen, de esta manera se observa como corresponden correctamente las máscaras a dichas imágenes.

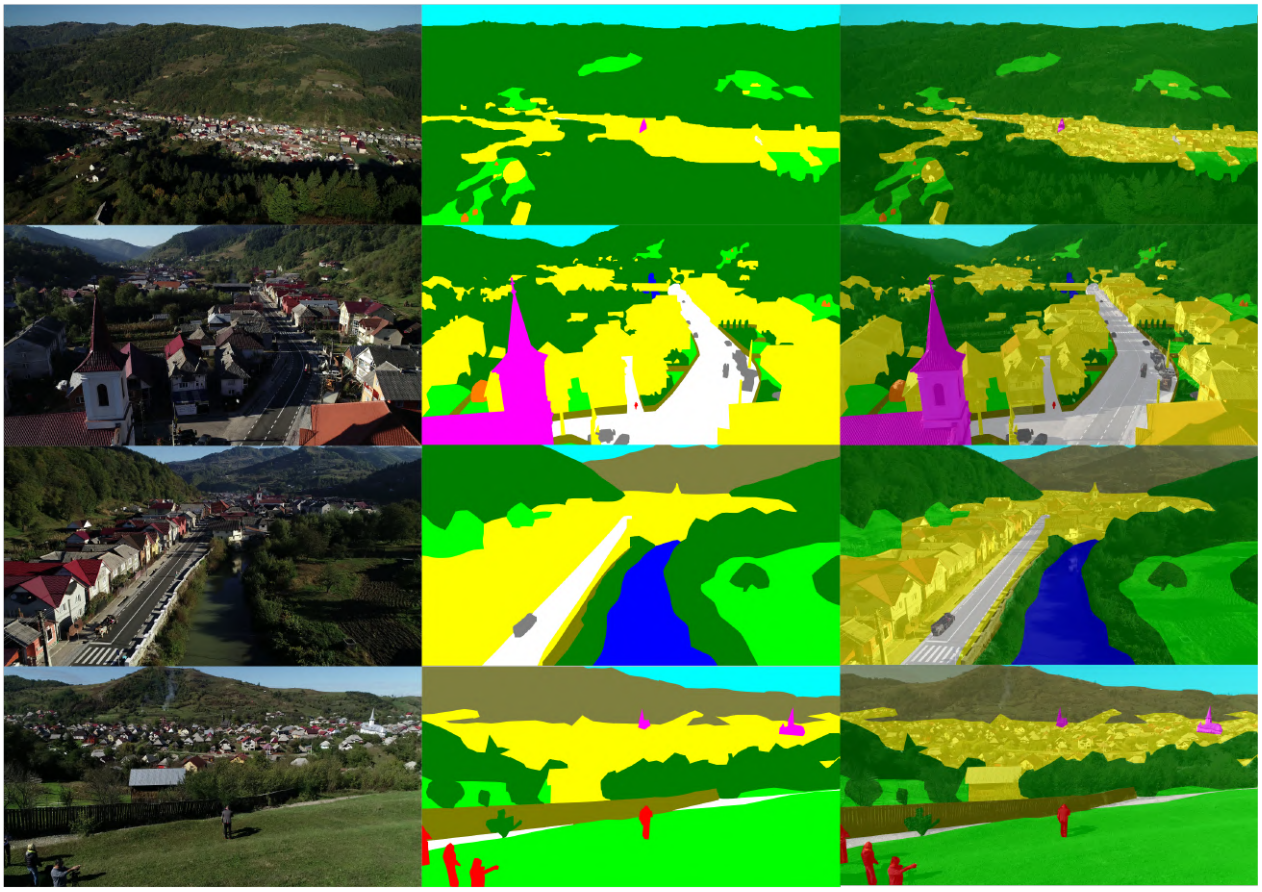

Ilustración 7.1: Imágenes, máscaras y superposiciones de las dos anteriores de cuatro de las imágenes presentes en el dataset rural. En el margen izquierdo se encuentran las imágenes reales, en la parte central las correspondientes máscaras, y en el margen derecho las superposiciones de ambas.

7.1.1.2. Distribución de las clases del dataset rural

Conociendo algunas de las imágenes del dataset, ahora se va a mostrar la distribución del mismo. Para ello, en la ilustración 7.2 se puede observar la distribución en la que se encuentran repartidas las distintas clases en las imágenes del dataset, realizado por el mismo grupo que desarolló el dataset. Como se observa, las clases con una mayor cantidad de píxeles son *forest*, *residential* y *land*, mientras que las clases que posee un menor número de píxeles en el dataset son las clases *car*, *person* y *haystack*.

Por otra parte, los mismos creadores del dataset también han aportado una gráfica en la que se puede contemplar la cantidad de imágenes en la que aparece cada clase del dataset, esto es lo que se observa en la ilustración 7.3.

Tal y como se observa, existe una correlación clara entre la cantidad de imágenes en las que aparece cada clase y el número de píxeles que se posee de cada clase en el dataset. En aquellas clases que sean de objetos más pequeños, como por ejemplo puede ser la clase *haystack* o la clase *person*, aparecen en casi la mitad de las imágenes del dataset, pero al ser clases de

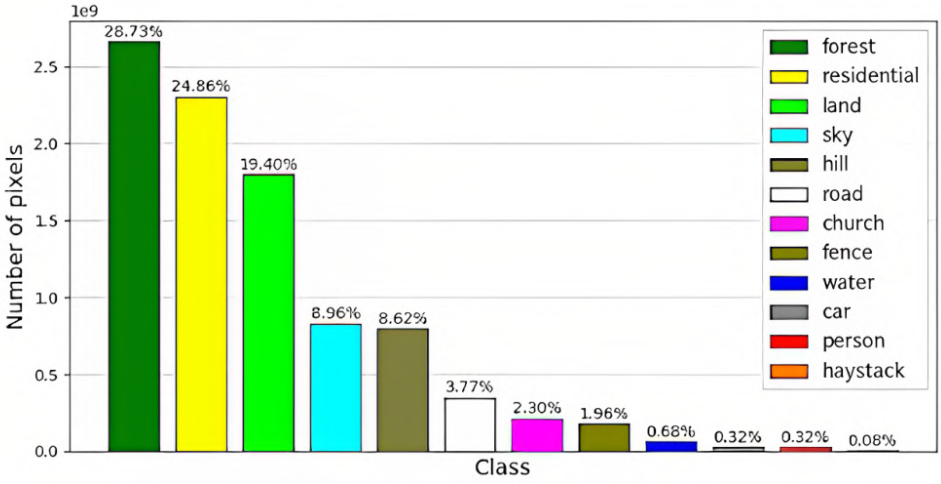

Ilustración 7.2: Gráfica en la que se muestra la cantidad de píxeles por clase en el dataset rural completo.

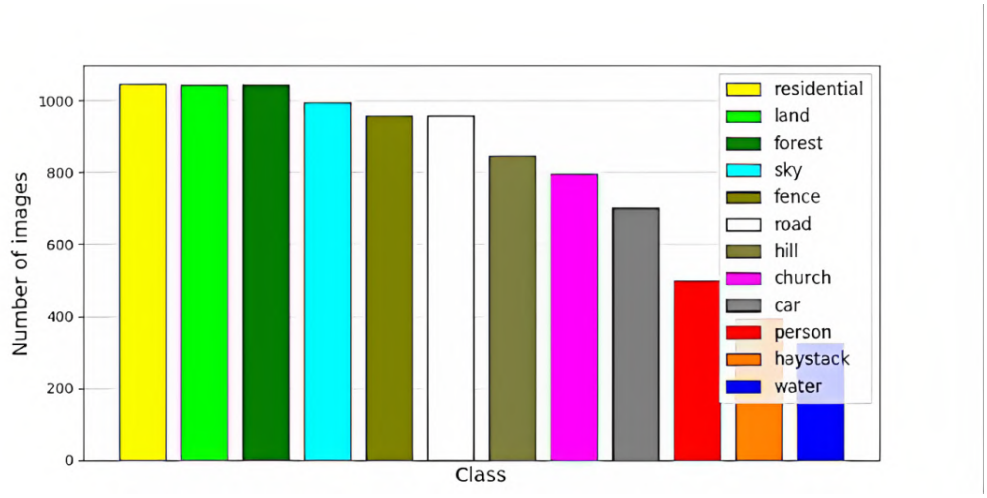

Ilustración 7.3: Gráfica en la que se muestra la cantidad de imágenes en la que aparece cada clase del dataset rural.

objetos pequeños la cantidad de píxeles de dichas clases en todo el dataset es muy pequeño en comparación con las clases *forest* y *residential*, en las que además de aparecer en casi todas las imágenes del dataset la cantidad de píxeles por imagen de las mismas es muy elevado.

7.1.2. Dataset urbano

El segundo dataset [17] [18] que se ha elegido se trata del dataset UAVid, este dataset recoge imágenes centradas es zonas urbanas, la idea es que con este dataset el modelo de segmentación semántica de imágenes aéreas que se va a montar sea lo más rígido posible. Este dataset, al contrario de lo que ocurría con el dataset rural, las imágenes y etiquetas ya

vienen dadas, es decir, no hizo falta obtener los frames de los vídeos.

7.1.2.1. Estructura del dataset urbano

Este dataset posee un número menor de imágenes con respecto al anterior, y es que en esta ocasión se poseen 200 imágenes de entrenamiento, 70 de validación y 150 de testeo, aunque estas últimas no poseen su correspondiente máscara, por lo que serán imágenes que podremos usar para comprobar como de capaz es el modelo en segmentar imágenes con las que no ha sido ni entrenado ni validado.

Este dataset presenta distintas clases de segmentación. En el cuadro 7.2 se observan las clases que tienen segmentadas este dataset y el color con el que se encuentran en las máscaras.

Cuadro 7.2: Representación de las clases del dataset urbano.

Color	Clase
	Building
	Road
	Static car
	Tree
	Low vegetation
	Human
	Moving car
	Background clutter

En la ilustración 7.4 se puede observar, al igual que se realizó con el dataset anterior, cuatro imágenes de este dataset urbano, para ello, en el margen izquierdo se encuentran las imágenes reales del dataset, en la parte central sus correspondientes máscaras y finalmente en el margen derecho se encuentran las superposiciones de las imágenes y las máscaras. De esta forma, se observa como las correspondencias de imágenes y máscaras son correcta.

7.1.2.2. Distribución de las clases del dataset urbano

En esta ocasión, los autores del dataset no han incluido gráficas acerca de como se encuentran organizadas las distintas clases en el dataset en cuanto al número de píxeles por clase y cantidad de imágenes en las que aparecen las mismas. Por ello, se ha desarrollado un pequeño script en Python para visualizar como se encuentra estructurado este dataset. En la ilustración 7.5 se puede observar la cantidad de píxeles que existen en el dataset por cada clase, en este caso, la cantidad de píxeles que se tienen son bastante menos que en comparación con el primer dataset, puesto que en este caso es del orden de 10^8, mientras que con el dataset anterior era de en torno a 10^9 la cantidad de píxeles.

A su vez, en la ilustración 7.6 se puede observar la cantidad de imágenes en las que se encuentra cada clase. Como se observa, la distribución de las clases en las imágenes es bastante equitativa, ya que 6 de sus 8 clases se encuentra en más de 250 imágenes de un total de 270

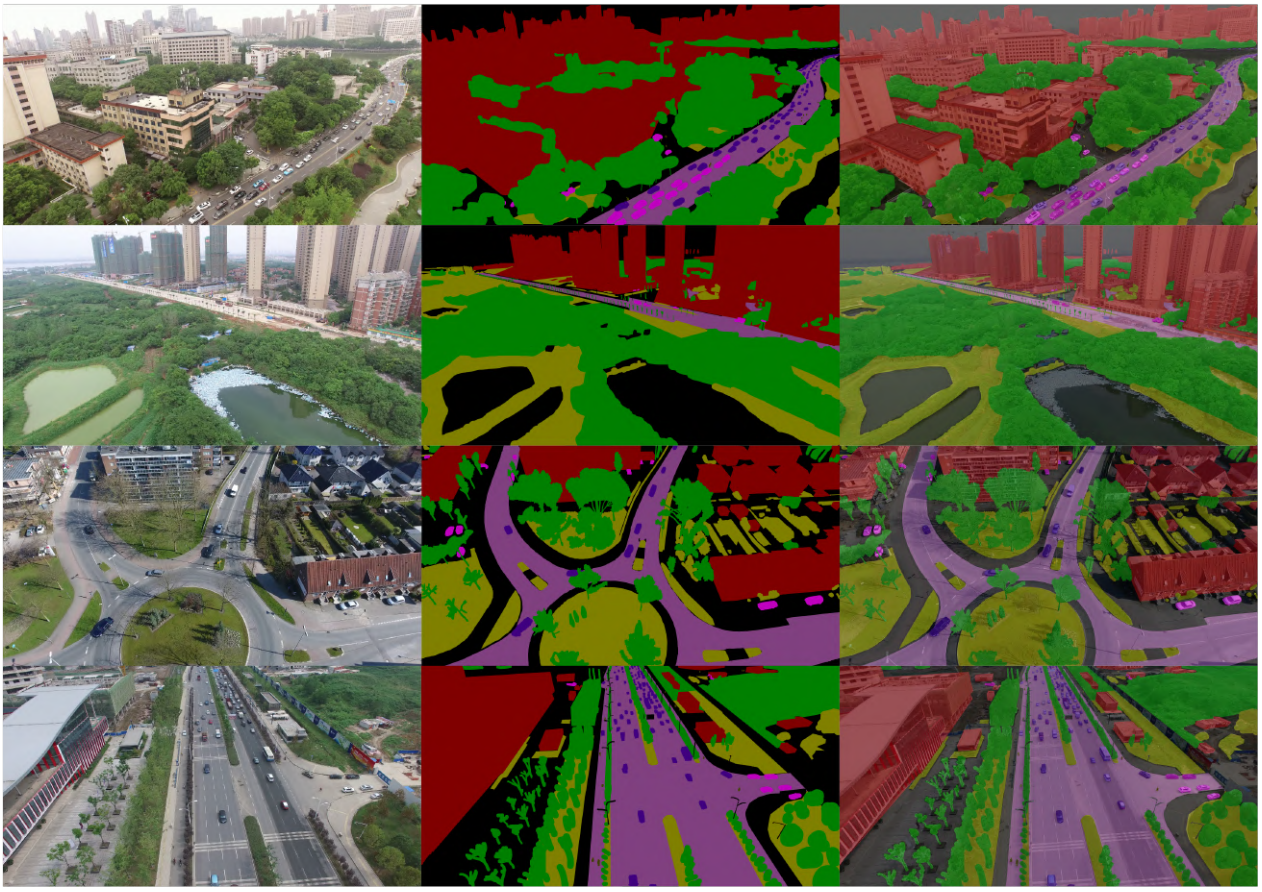

Ilustración 7.4: Imágenes, máscaras y superposiciones de las dos anteriores de cuatro de las imágenes presentes en el dataset urbano. En el margen izquierdo se encuentran las imágenes reales, en la parte central las correspondientes máscaras, y en el margen derecho las superposiciones de ambas.

imágenes que posee el dataset. Por lo que la distribución de las clases en el mismo es bastante equilibrado, pero no así la cantidad de píxeles como se observó anteriormente.

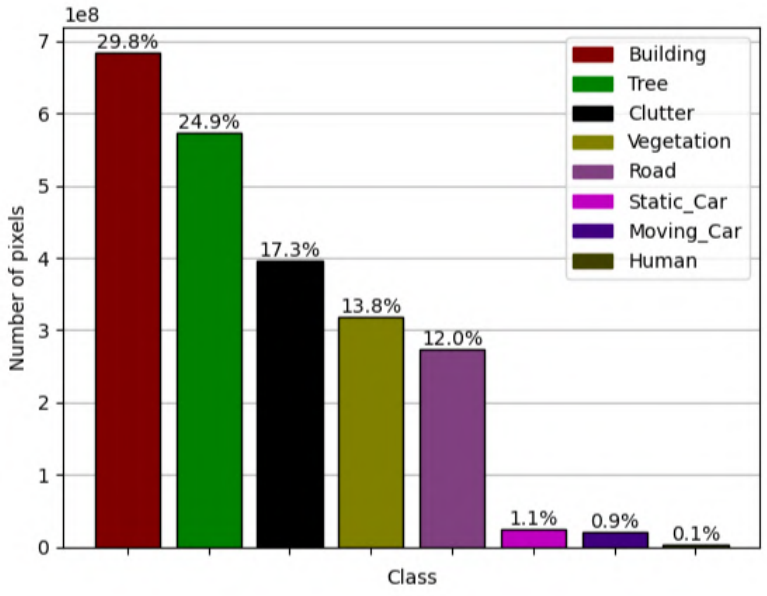

Ilustración 7.5: Gráfica en la que se muestra la cantidad de píxeles por clase en el dataset urbano completo.

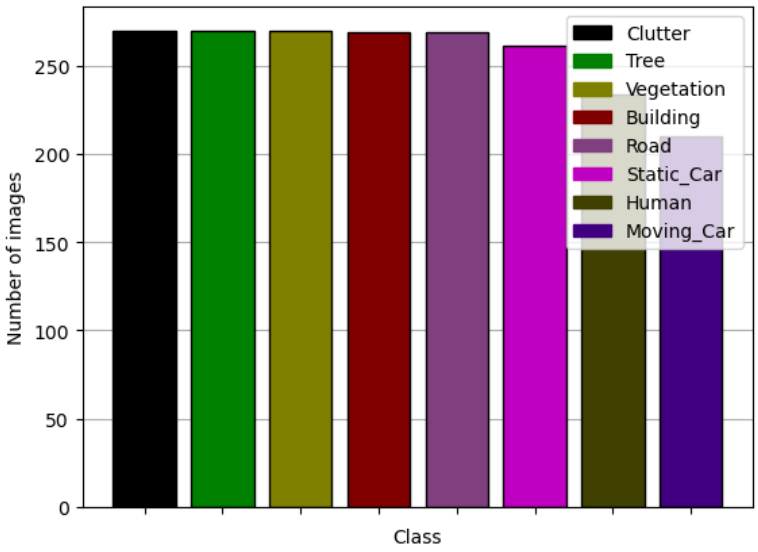

Ilustración 7.6: Gráfica en la que se muestra la cantidad de imágenes en la que aparece cada clase.

7.2. Modelos de segmentación semántica usados

En el capítulo del estado del arte (capítulo 2) se observaron algunas de las distintas arquitecturas existentes para la segmentación semántica. En esta sección, se comentarán las arquitecturas para modelos de segmentación semántica que se utilizarán para el posterior entrenamiento de los distintos datasets. En este sentido, para este trabajo de fin de grado se

han trabajo principalmente con dos arquitecturas distintas:

- ResNetUNet
- DeepLabV3

Se ha decidido utilizar estas dos arquitecturas por las siguientes razones, y es que se ha observado cuando se encontraba buscando distintas arquitecturas, que la arquitectura ResNetUNet obtenía muy buenos resultados en un periodo de tiempo relativamente corto, además, se trata de una arquitectura que es una combinación de dos arquitecturas distintas, la arquitectura ResNet y la arquitectura Unet.

ResNet se trata de una red que está basada en redes neuronales residuales con saltos normalmente comprendidos entre dos y tres capas [8]. La idea principal que se encuentra detrás de este tipo de redes es que cuando se posee una red neuronal compleja y muy profunda es posible que se comience a tener problemas con la desaparición del gradiente y con el sobreentrenamiento. Se entiende por sobreentrenamiento cuando la red ya no es capaz de aprender nuevas características y comienza a memorizar las características de las imágenes de entrenamiento. El sobreentrenamiento por tanto hace que las métricas que se obtienen con las imágenes de entrenamiento sean sumamente buenas, pero con otras imágenes el modelo no es capaz de resolver el problema de manera satisfactoria. Por ello, aparece el término de redes residuales por el año 2016 en el artículo Deep Residual Learning for Image Recognition desarrollado por el equipo de Microsoft Research liderado por Kaiming [23].

Las ventajas que presenta este tipo de redes residuales son claras, y es que al poseer saltos entre capas, el gradiente es capaz de ir retrocediendo e ir saltando las capas por las que no están conectadas obteniendo de esta forma un gradiente el cual ha sido menos manipulado por las distintas capas de la red. Por otra parte, otra de las ventajas de este tipo de redes es que cuando se llega al final de cada bloque residual se tiene información que ha pasado por las distintas capas, y otra información que no ha pasado por dichas capas, de esta manera se posee dos fuentes de información y con esto hará que aumente el rendimiento de la propia red.

Por esto, la primera de las arquitecturas que se ha elegido ha sido la ResNetUNet, esta implementación que se ha encontrado parte de un modelo pre-entrenado con Resnet18, y la arquitectura hace uso de estos pesos entrenados. En este caso, Resnet18 está preentrenado con el dataset ImageNet, el cual ha sido explicado en el capítulo 2.

Por otra parte, se ha elegido también como arquitectura la arquitectura DeepLabV3. En esta ocasión se ha decidido escoger una de las arquitecturas más recientes y que se encuentre implementada por Pytorch y que además tenga un pre-entrenamiento. Se ha decidido utilizar una arquitectura ya creada por Pytorch para ver lo transparente que puede llegar a ser la implementación de la propia arquitectura que se desea utilizar, de esta manera realizando una pequeña modificación en la última de sus capas, este modelo es capaz de entrenar y segmentar cualquier dataset que se tenga. Además, se ha querido que esta segunda arquitectura que se ha escogido también posea un pre-entrenamiento, para que de esta forma, cuando el modelo comience a entrenarse ya tenga unos pesos por defecto, de esta manera los pesos no comenzarán en 0 y será mejor siempre partir de un valor, que partir desde 0. De esta

forma cuando se lleve a cabo la comparación de las arquitecturas, ambas son arquitecturas pre-entrenadas y por tanto no será necesario llevar a cabo una consideración por parte de algunas de ellas en caso de que no estuviesen pre-entrenadas.

DeepLabV3 se trata de la versión siguiente a DeepLabV2 y ha sido desarrollada por el equipo de investigación de Google Brain. Esta arquitectura funciona de manera distinta a la arquitectura ResNetUNet, y es que DeepLabV3 utiliza una red convolucional para extraer características de las imágenes que se le pasan como entrada, seguida de una conocida como *dilated convolutions*, que aumenta el tamaño del campo receptivo sin aumentar la cantidad de parámetros entrenables, es decir, la neurona puede "ver" una región más grande de la entrada en cada capa, haciendo que de esta manera aumente su capacidad para entender el contexto de la imagen en lugar de centrarse solo en píxeles individuales. Luego, se utiliza una técnica llamada *atrous spatial pyramid pooling* (ASPP) para capturar características a diferentes escalas y niveles de detalle. Ya finalmente, se utiliza una capa de convolución para obtener una salida de segmentación para cada píxel de la imagen. El pre-entrenamiento de esta arquitectura se ha realizado con el dataset Pascal VOC 2012, el cual consta de 20 clases más una clase de fondo.

7.3. Métricas utilizadas

En esta sección se van a explicar las métricas que se han usado y que se analizarán a lo largo de esta memoria. Así pues, las métricas que se han usado son [14] [3]:

- **Precision**. Esta métrica mide la capacidad del modelo de realizar predicciones precisas en una clase específica.
- **Recall**. Mientras que la precisión lo que mide es que tan bueno es el modelo entrenado, el recall mide la capacidad del modelo para identificar todos los casos positivos de una clase.
- **Accuracy**. Esta métrica informa sobre el porcentaje de casos que el modelo ha detectado de forma correcta.
- **F1**. Como se ha visto, los valores de precisión y recall son claves para conocer la efectividad de un modelo, por ello, la métrica de F1 lo que trata de hacer es combinar ambas métricas en una sola, asumiendo que precisión y recall poseen el mismo grado de interés.
- ***IOU (Intersection Over Union)***. Esta medida nos da un valor que será resultado de la comparación píxel a píxel de las imágenes reales con las imágenes predichas por el modelo.
- **DICE**. Se trata de otra medida similar al IOU en la que se intenta comparar las imágenes reales con las predicha por el modelo entrenado.

Las expresiones usadas para calcular las métricas anteriores se basan en los TP (*True Positives*), TN (*True Negatives*), FP (*False Positives*) y FN (*False Negatives*). A continuación se

explica de una manera más detallada el significado de cada uno de ellos:

- **TP**. Se tratan de los positivos verdaderos, y corresponden a cuando el modelo determina a un píxel una cierta clase, y la máscara confirma que se trata de dicha clase.
- **TN**. Se tratan de los negativos verdaderos, y corresponden a cuando el modelo determina que un píxel no pertenece a una clase, y efectivamente en la máscara real dicho píxel no corresponde a dicha clase.
- **FP**. Se tratan de los falsos positivos, y es cuando el modelo determina que un píxel pertenece a una clase cuando verdaderamente en la máscara dicho píxel no corresponde a la clase predicha.
- **FN**. Se tratan de los falsos negativos, y es cuando el modelo determina que un píxel no pertenece a una clase, cuando verdaderamente en la máscara dicho píxel corresponde a la clase.

Por tanto, teniendo en cuenta las definiciones anteriores, a continuación, se van mostrar las expresiones utilizadas para cada métrica:

- **Expresión para la precisión:**

$$\text{Precisión} = \frac{TP}{TP + FP} \tag{7.1}$$

- **Expresión para el recall:**

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7.2}$$

- **Expresión para el accuracy:**

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.3}$$

- **Expresión para el F1:**

$$\text{F1} = 2 \cdot \frac{Precision * Recall}{Precision + Recall} \tag{7.4}$$

- **Expresión para IOU:**

$$\text{IOU} = \frac{TP}{FP + TP + FN} \tag{7.5}$$

- **Expresión para DICE:**

$$\text{DICE} = \frac{2TP}{FP + 2 * TP + FN} \tag{7.6}$$

En las distintas ejecuciones que se han realizado para el entrenamiento de los modelos, se ha observado que los valores obtenidos por las métricas F1 y DICE son siempre exactamente

iguales entre sí, por lo que se ha llevado a cabo una demostración matemática para comprobar si realmente ambas métricas son equivalentes, a continuación se demuestra como ambas métricas son equivalentes, operando sobre la expresión de F1:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = \tag{7.7}$$

Sustituyendo precision y recall por sus correspondientes expresiones:

$$= 2 \cdot \frac{\frac{TP}{TP+FP} \cdot \frac{TP}{TP+FN}}{\frac{TP}{TP+FP} + \frac{TP}{TP+FN}} = \tag{7.8}$$

$$= 2 \cdot \frac{\frac{TP^2}{(TP+FP)(TP+FN)}}{\frac{TP(TP+FN)+TP(TP+FP)}{(TP+FP)(TP+FN)}} = \tag{7.9}$$

$$= \frac{2TP^2}{(TP+FP)(TP+FN)} : \frac{TP(TP+FN)+TP(TP+FP)}{(TP+FP)(TP+FN)} = \tag{7.10}$$

$$= \frac{2TP^2}{TP(TP+FN)+TP(TP+FP)} = \tag{7.11}$$

$$= \frac{2TP^2}{TP(TP+FN+TP+FP)} = \tag{7.12}$$

$$= \frac{2TP}{FP+2TP+FN} \tag{7.13}$$

Como se observa en las expresiones anteriores, al operar sobre la expresión inicial de F1 se llega a la misma expresión que la de DICE. De esta forma, en las métricas que se van a mostrar aquí no se va a enseñar dicho valor ya que se deduce del propio valor de F1.

7.4. Cantidad de imágenes y tratamiento de máscaras

Para obtener una cantidad de imágenes de entrenamiento, validación y testeo, se ha realizado de tal manera que de forma aleatoria en cada conjunto de datos, se obtenga un 80 % del total de imágenes como entrenamiento, un 10 % de validación y el 10 % final será usado como testeo. En este sentido, para el primero de los datasets, es decir, el dataset rural se tendría un total de 652 imágenes de entrenamiento, 82 imágenes para validación y otras 82 para testeo. Mientras que para el caso del segundo de los datasets, el dataset urbano, se ha observado

como la cantidad de imágenes era bastante inferior a las cifras que se tiene para el primero de los datasets, por ello, se ha realizado *data augmentation* el cual ha consistido en realizar un volteo horizontal a las imágenes y máscaras, de esta forma, se pasó de tener un total de 270 imágenes a 540. De esta manera, se tiene un total de 432 imágenes de entrenamiento, 54 imágenes para validación, y otras 54 que serán usadas de testeo.

Para el entrenamiento, es necesario que cada máscara en vez de sus píxeles venir dados por colores con tres canales de color (RGB) deben de ser indicados con un valor, es decir, que cada clase que sea referenciada por un valor en un único canal de color en la máscara. Por ello, se ha escrito un pequeño script en el que lo que realiza es recorrer las máscaras que se encuentran en una ruta y cambia sus valores de RGB por tan solo un número, en el cuadro 7.3 se observa como ahora cada color viene representado por un solo dígito, lo mismo ocurre para el dataset urbano, en el que cuyos valores de RGB se han representado por un solo dígito tal y como se recoge en el cuadro 7.4.

Cuadro 7.3: Nuevos valores de píxeles identificativos para cada clase del dataset rural.

Color	RGB	Clase	Valor de clase representativo
	[127, 127, 0]	Fence	1
	[0, 255, 0]	Land	2
	[0, 127, 0]	Forest	3
	[255, 255, 0]	Residential	4
	[255, 127, 0]	Haystack	5
	[255, 255, 255]	Road	6
	[255, 0, 255]	Church	7
	[127, 127, 127]	Car	8
	[0, 0, 255]	Water	9
	[0, 255, 255]	Sky	10
	[127, 127, 63]	Hill	11
	[255, 0, 0]	Person	12

Cuadro 7.4: Nuevos valores de píxeles identificativos para cada clase del dataset urbano.

Color	RGB	Clase	Valor de clase representativo
	[0, 0, 0]	Background clutter	1
	[128, 0, 0]	Building	2
	[128, 64, 128]	Road	3
	[192, 0, 192]	Static car	4
	[128, 0, 0]	Tree	5
	[128, 128, 0]	Low vegetation	6
	[64, 64, 0]	Human	7
	[64, 0, 128]	Moving car	8

De esta manera, las máscaras que se usarán para el entrenamiento solo tendrán un canal de color, en esta ocasión se ha elegido el rojo. A modo de ejemplo, en la ilustración 7.7 se puede observar una máscara de un solo canal.

Ilustración 7.7: Máscara de una imagen con un solo canal de color.

7.5. Primeros entrenamientos

Para la realización del entrenamiento, se han ido realizando de manera iterativa mejoras en el mismo de manera que los valores de las métricas sean correctos, es decir, que se estén tomando todas las imágenes de entrenamiento y validación. Además, se han realizado diversos experimentos con distintos tamaños de imágenes para observar cual sería el tamaño correcto de imágenes a utilizar, así como la cantidad de imágenes que posee cada *batch*, en el que se ha definido finalmente utilizar un *batch size* de 13, este número se debe a que es la cantidad más alta de *batches* que pudo soportar la GPU en la que se llevó a cabo el entrenamiento, aunque primeramente se observarán los resultados con un *batch size* de 4. Finalmente, indicar que la política que se ha tomado para la parada del aprendizaje es considerar que si el coste de validación que se realiza en cada época no mejora en tres épocas consecutivas, se asume que se ha encontrado el punto de inflexión en la curva de aprendizaje en donde ahora se comienza el procedimiento definido como sobreentrenamiento, por ello, se finaliza automáticamente el entrenamiento y se queda con el modelo entrenado anterior al penúltimo de los modelos, que corresponderá con el modelo que tendrá un mejor coste de validación.

Ahora se va a explicar los distintos valores de las métricas que se han ido obteniendo a lo largo de los entrenamientos que se han ido realizando. Para ello, se observarán exclusivamente las métricas sobre el conjunto de testeo, ya que se trata del conjunto que no ha visto el modelo. Ya que la intención es segmentar imágenes con las que no se tiene un etiquetado y que el modelo no ha sido entrenado con ellos, en este sentido, con el conjunto de testeo se obtendrán unas métricas más ajustadas a las imágenes no etiquetadas.

7.5.1. Resultados obtenidos sobre las clases en el dataset rural

Por ello, lo primero que se comenzó fue entrenando la red neuronal utilizando la arquitectura ResNetUNet con el primero de los datasets, es decir, el dataset rural, el entrenamiento se realizó con un tamaño de *batch* de 4 imágenes por *batch* y con imágenes de 1376x736, además

de entrenar el modelo durante 10 épocas únicamente. De esta forma, los valores que se obtuvieron sobre el conjunto de testeo fueron las que se muestran en la ilustración 7.8.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Hill	0.9000	0.9479	0.9916	0.9233	0.8579
Forest	0.9178	0.9392	0.9507	0.9284	0.8663
Residential	0.9158	0.9232	0.9628	0.9195	0.8510
Land	0.9219	0.8817	0.9624	0.9014	0.8204
Church	0.9033	0.8975	0.9967	0.9004	0.8189
Road	0.8745	0.8348	0.9875	0.8542	0.7455
Fence	0.7280	0.6924	0.9889	0.7098	0.5501
Person	0.7055	0.7933	0.9989	0.7468	0.5960
Car	0.6478	0.4856	0.9980	0.5551	0.3842
Sky	0.9876	0.9924	0.9983	0.9900	0.9802
Haystack	0.6280	0.1673	0.9992	0.2643	0.1523
Water	0.9336	0.8959	0.9979	0.9144	0.8423
Global	0.9164	0.9164	0.9861	0.9164	0.8459

Ilustración 7.8: Métricas obtenidas con el conjunto de testeo sobre el modelo entrenado con el dataset rural, utilizando la arquitectura ResNetUNet entrenado con 10 épocas.

Análogamente, entrenando con los mismos hiperparámetros y dataset pero tan solo cambiando la arquitectura de ResNetUNet a DeepLabV3, se obtuvieron como valores de las métricas para el conjunto de testeo las que se muestran en la ilustración 7.9.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Hill	0.9322	0.9129	0.9916	0.9225	0.8561
Forest	0.9341	0.9360	0.9552	0.9351	0.8781
Residential	0.9180	0.9388	0.9671	0.9283	0.8661
Land	0.9199	0.9067	0.9668	0.9133	0.8404
Church	0.9315	0.9205	0.9976	0.9260	0.8621
Road	0.8833	0.8869	0.9902	0.8851	0.7939
Fence	0.8124	0.7552	0.9921	0.7827	0.6430
Person	0.7765	0.7344	0.9991	0.7549	0.6063
Car	0.7770	0.6502	0.9986	0.7072	0.5470
Sky	0.9770	0.9920	0.9973	0.9844	0.9693
Haystack	0.6706	0.1126	0.9992	0.1929	0.1067
Water	0.9447	0.8946	0.9980	0.9190	0.8501
Global	0.9264	0.9264	0.9877	0.9264	0.8628

Ilustración 7.9: Métricas obtenidas con el conjunto de testeo sobre el modelo entrenado con el dataset rural, utilizando la arquitectura DeepLabV3 entrenado con 10 épocas.

Para realizar una comparación acerca de los valores de las métricas obtenidas en las ilustraciones 7.8 y 7.9 se ha creado una tabla comparativa en la que se muestran los valores obtenidos para cada métrica utilizando ambas arquitecturas, esta tabla corresponde con la que se observa en la ilustración 7.10. Como se aprecia, esta tabla no es más que la unión de las dos anteriores, de esta manera, se podrá analizar los resultados obtenidos utilizando ambas arquitecturas. En este sentido, se llevará a cabo una exhaustiva comparación de los resultados obtenidos para cada clase.

7.5.1.1. Análisis de los resultados obtenidos sobre las clases en el dataset rural

Nombre Clase	Precisión ResNetUNet	Precisión DeepLabV3	Recall ResNetUNet	Recall DeepLabV3	Accuracy ResNetUNet	Accuracy DeepLabV3	F1 ResNetUNet	F1 DeepLabV3	IOU ResNetUNet	IOU DeepLabV3
Hill	0.9000	0.9322	0.9479	0.9129	0.9916	0.9916	0.9233	0.9225	0.8576	0.8561
Forest	0.9178	0.9341	0.9392	0.9360	0.9507	0.9552	0.9284	0.9351	0.8663	0.8781
Residential	0.9158	0.9180	0.9232	0.9388	0.9628	0.9671	0.9195	0.9283	0.8510	0.8661
Land	0.9219	0.9199	0.8817	0.9067	0.9624	0.9668	0.9014	0.9133	0.8204	0.8404
Church	0.9033	0.9315	0.8975	0.9205	0.9967	0.9976	0.9004	0.9260	0.8189	0.8621
Road	0.8745	0.8833	0.8348	0.8869	0.9875	0.9902	0.8542	0.8851	0.7455	0.7939
Fence	0.7280	0.8124	0.6924	0.7552	0.9889	0.9921	0.7098	0.7827	0.5501	0.6430
Person	0.7055	0.7765	0.7933	0.7344	0.9989	0.9991	0.7468	0.7549	0.5960	0.6063
Car	0.6478	0.7750	0.4856	0.6502	0.9980	0.9986	0.5551	0.7072	0.3842	0.5470
Sky	0.9876	0.9770	0.9924	0.9920	0.9983	0.9973	0.9900	0.9844	0.9802	0.9693
Haystack	0.6280	0.6706	0.1673	0.1126	0.9992	0.9992	0.2643	0.1929	0.1523	0.1067
Water	0.9336	0.9447	0.8959	0.8946	0.9979	0.9980	0.9144	0.9190	0.8423	0.8501
Global	0.9164	0.9264	0.9164	0.9264	0.9861	0.9877	0.9164	0.9264	0.8458	0.8628

Ilustración 7.10: Comparación de los valores obtenidos para el conjunto de testeo del dataset rural con las arquitecturas ResNetUNet y DeepLabV3 con entrenamiento de 10 épocas.

Primeramente, se va a comenzar analizando los resultados de la tabla 7.10 por filas, es decir, por clases. La primera de las clases corresponde con la clase *hill* esta clase posee una precisión de 0.9000 para ResNetUNet y una precisión de 0.9322 para DeepLabV3, además, en cuanto al recall, se observa como para la arquitectura ResNetUNet el valor es mayor con respecto a la arquitectura DeepLabV3 (0.9479 frente a 0.9129). En cuanto a la métrica accuracy, no se va ser tenida en cuenta para el estudio, ya que aunque se trata de un valor interesante que siempre hay que tener en cuenta para comprobar que tan bien o no, está realizando el modelo la segmentación semántica, en esta ocasión al no estar las clases balanceadas (tal y como se observó en el subcapítulo 7.1.1.2) la métrica accuracy puede estar dando valores que son bastante buenos cuando en realidad el modelo no está segmentado la clase de manera correcta. Por este motivo, las métricas precisión, recall, F1 e incluso IOU pueden dar resultados más exactos a la segmentación que está realizando el modelo. Por otra parte, los valores obtenidos para la métrica F1, que si se recuerda se asume que tanto la precisión como el recall tienen la misma importancia, poseen valores muy ajustados entre las arquitecturas, puesto que en este caso dichos valores son de 0.9233 para ResNetUNet y de 0.9225 para DeepLabV3. Lo mismo ocurre con la métrica IOU, los valores que se han obtenido son bastante ajustados, puesto

que para la arquitectura ResNetUNet se obtuvo un valor de 0.8576 mientras que para la de DeepLabV3 dicho valor es de 0.8561, es decir, con ambas arquitecturas se posee un IOU de un 85 %, lo que quiere decir que el 85 % de los píxeles entre la máscara y la máscara predicha concuerdan entre sí. En este caso se han obtenido mejores resultados con la arquitectura ResNetUNet.

Analizando ahora la clase *forest*, se observa como en esta ocasión se tiene un mejor valor de precisión utilizando la arquitectura DeepLabV3, un 0.9341, aunque en recall el valor es algo superior con la arquitectura ResNetUNet. Pero al poseer una mayor diferencia de valores en la precisión que en el recall, esto se ve reflejado en la métrica F1, en donde se asume que ambas métricas anteriores importan de la misma manera y por tanto se obtiene un valor mayor en DeepLabV3 con respecto a ResNetUNet, 0.9351 frente a 0.9284. Además, para la métrica IOU se obtiene un valor más alto para la arquitectura DeepLabV3. En esta ocasión, se ha observado como se obtienen unos mejores resultados para la arquitectura DeepLabV3 para esta clase en concreto.

Siguiendo ahora con la clase *residential* los valores que se han obtenido de precisión son bastante ajustados, siendo mayor en la arquitectura DeepLabV3, esto no ocurre con el recall, en donde la diferencia es mayor teniendo un valor de 0.9232 para ResNetUNet y 0.9388 para DeepLabV3. Tal y como se esperaba, al poseer en ambas métricas mejores valores en DeepLabV3, esto se extrapola a la métrica F1, en donde el valor asciende a 0.9283 en la arquitectura DeepLabV3 frente al 0.9195 en ResNetUNet. En esta ocasión, también se tiene un mejor resultado en IOU haciendo uso de la arquitectura DeepLabV3.

En relación a la clase *land*, se observa como se posee un valor de precisión mayor para la arquitectura ResNetUNet, mientras que en el recall se posee un mayor valor para DeepLabV3. Sin embargo, para la métrica F1, se observa como se obtiene un mejor valor para la arquitectura DeepLabV3. Esta tendencia también se repite con la métrica IOU, en el que con un 0.9014 de DeepLabV3 supera a ResNetUNet con un 0.8404. Observándose de esta manera como DeepLabV3 es el que está dando mejores resultados por un momento de manera general.

La clase *church* si se recuerda, se trata de una clase con una menor aparición en las imágenes, esta clase corresponde con poco más del 2 % de los píxeles del dataset, aunque no por ello se obtienen unos valores bajos, puesto que para la métrica F1 con la arquitectura ResNetUNet se obtiene un 0.9004 y con DeepLabV3 el valor asciende a 0.9260. La diferencia se hace notar aún más en la métrica IOU, en el que se obtiene un 0.8189 para ResNetUNet, mientras que para DeepLabV3 la cifra aumenta hasta 0.8621, de esta manera, se obtiene de una forma bastante clara como se están teniendo mejores resultados para la arquitectura DeepLabV3.

Observando los resultados obtenidos para la clase *road*, vemos que se obtienen los valores más bajos por el momento, ya que como precisión se tienen valores de 0.8745 para la arquitectura ResNetUNet, mientras que para DeepLabV3 dicho valor es de 0.8833, y en cuento al recall se obtiene un 0.8348 para ResNetUNet y un 0.8869 para DeepLabV3, en ambos casos se poseen unos mayores valores haciendo uso de la arquitectura DeepLabV3. Lo mismo ocurre con el valor de las métricas F1 e IOU, en el que en ambos casos se obtienen valores más altos para la arquitectura DeepLabV3, siendo éstos de 0.8851 para F1 y de 0.7939 para IOU.

Las clases *fence*, *person* y *car* son clases que no han obtenido unos buenos valores de seg-

mentación semántica, y es que se han obtenido como intersección para esas clases los valores de 0.6430, 0.6063 y 0.5470 respectivamente para la arquitectura DeepLabV3 que es donde mejores resultados se han obtenido, es decir, ninguna de las clases son capaces de intersectar más del 65 % de los píxeles. A pesar de observar que los valores de precisión y de recall no son bajos, al poseer estas clases intersecciones tan bajas, demuestra de forma cuantitativa como el modelo no está segmentando de la mejora manera estas tres clases.

Por otra parte, estas clases son un buen ejemplo para observar de manera cuantitativa el porqué la métrica accuracy no es una buena métrica a tener exclusivamente en cuenta si se tienen datasets no balanceados. En esta ocasión si tan solo se fijara en los valores de accuracy obtenidos por ejemplo para la clase *fence*, se tendría un 0.9889 para la arquitectura ResNetUNet, mientras que utilizando la arquitectura DeepLabV3 se tendría un 0.9921, analizando ambos resultados se llegaría a la misma conclusión que se ha llegado, y es que la arquitectura DeepLabV3 funciona mejor para esta clase. Pero el problema radica en que además de esa información, también se está obteniendo una falsa seguridad y es que los valores son bastante buenos, fíjese en el 0.9921, y es que al ver este dato se está dando por hecho que el modelo está segmentando muy bien, ya que más del 99 % de las veces acierta, y realmente esto no es así. Esto ocurre como bien se ha indicado cuando se poseen datasets cuyas clases no están balanceadas. Por ello, siempre es mejor analizar los resultados obtenidos con las demás métricas que se han ido utilizando, ya que son valores más ajustados a como realmente está segmentado el modelo entrenado.

La clase *sky* se trata de la clase que mejor segmenta el modelo, y es que se obtiene como precisión un 0.9876 para la arquitectura ResNetUNet, mientras que para DeepLabV3 este valor baja hasta el 0.9770, el cual sigue siendo realmente un buen valor de precisión. En cuanto al recall los valores mejoran aún más, y es que se obtiene un 0.9924 y un 0.9920 para las arquitecturas ResNetUNet y DeepLabV3 respectivamente. Esto valores se traducen en la métrica F1 de manera que para ResNetUNet el valor obtenido es de un 0.9900, mientras que para DeepLabV3 el valor es de 0.9844. Como se observa, con las métricas obtenidas se puede concluir que se trata de la mejor clase que es capaz de segmentar el modelo, además, si analizamos los valores obtenidos con la métrica IOU se obtiene un 0.9802 para ResNetUNet y un 0.9693 para DeepLabV3. De esta manera, con estos valores para esta clase, se puede deducir que en esta ocasión, se obtienen mejores resultados con la arquitectura ResNetUNet, a diferencia de las clases anteriores en donde se obtenían mejores valores de las métricas haciendo uso de la arquitectura DeepLabV3.

La clase *haystack* es la clase que posee el menor número de píxeles de todo el dataset, siendo además la segunda clase con menos apariciones en todo el dataset. Por ello, los valores que se han obtenido para esta clase son realmente malos, solamente observando los valores de IOU se aprecia como tan solo el 15,23 % intersecta para el caso en el que se utiliza la arquitectura ResNetUNet, mientras que para DeepLabV3 este valor baja a un 10,67 %. De esta manera, se puede afirmar que esta clase realmente no es capaz de ser identificada para una correcta segmentación haciendo uso de ninguna de las dos arquitecturas que se han elegido, esto se debe a que son clases muy específicas, con pocas apariciones y con muy poca densidad de píxeles en las imágenes en las que aparece.

Finalmente se encuentra la clase *water*, se trata de la clase en las que menos imágenes aparece

en el dataset, además, la cantidad de píxeles total no llega ni al 1 % del total de píxeles del dataset, correspondiendo también así una de las clases más perjudicadas por la no existencia de un balanceamiento de clases. No obstante, los resultados que se obtienen son relativamente admirables, y es que se ha obtenido como precisión un 0.9336 para ResNetUNet y un 0.9447 para DeepLabV3, aunque los valores de recall si son algo inferiores a los obtenidos en la precisión, son bastante altos teniendo en cuenta la cantidad de píxeles totales de esta clase en el dataset, y es que se han obtenido como recall un 0.8959 para ResNetUNet mientras que para DeepLabV3 este valor se ha quedado en un 0.8946. Con estos resultados, más el IOU que son de 0.8423 y 0.8501 para ResNetUNet y DeepLabV3 respectivamente, se puede concluir que en esta ocasión la arquitectura con la que se tiene unos mejores valores es la arquitectura DeepLabV3.

7.5.1.2. Análisis de los resultados obtenidos globalmente en el dataset rural

Para concluir el análisis de esta dos arquitecturas con el dataset rural, se ha añadido a la tabla 7.10 una fila la cual tiene como nombre de clase *global*, esta clase es la que proporciona de una forma generalizada toda la información de las clases anteriores recogidas en una sola clase. Como se observa, los valores de precisión, recall y F1 son iguales, y esto es debido a que se han tenido en cuenta todos los valores de las distintas clases, por ello y según la forma en la que matemáticamente se deducen las expresiones de precisión y recall se puede deducir que si se tiene el mismo número de píxeles para el recuento (es decir, el mismo número de TP, TN, FP y FN) los valores de precisión, recall y F1 globales son idénticos.

Sabiendo esto, los valores obtenidos de forma global de precisión, recall y F1 son de 0.9164 para el caso en el que se utiliza la arquitectura ResNetUNet y 0.9264 para cuando se hace uso de la arquitectura DeepLabV3. Por otra parte, los valores de IOU son diferentes, en este caso de forma global se tiene que las máscaras predichas de testeo intersectan con las máscaras reales en un 84,58 % para el caso de ResNetUNet, mientras que esta intersección asciende hasta el 86,28 % para el caso de la arquitectura DeepLabV3.

De esta forma, sacando unas conclusiones preliminares tan solo habiendo realizado la predicción con el dataset rural, se puede deducir que la arquitectura con la que se ha obtenido unos mejores resultados tanto por la cantidad de clases en las que se ha observado que es mejor como por los valores obtenidos de forma global, es con la arquitectura DeepLabV3.

Esto se puede observar de una manera mucho más visual en la ilustración 7.11, en donde tal y como se observa aparece una gráfica en la que se muestra los valores obtenidos en la métrica IOU para todas las clases y de forma global, utilizando las arquitecturas ResNetUNet y DeepLabV3.

Cómo se observa en la ilustración 7.11, se pueden obtener las mismas conclusiones que se obtuvieron con el análisis de los datos, y es que para 10 de las 12 clases presentes en el dataset se obtienen mejores valores de intersección con la arquitectura DeepLabV3, y lo mismo ocurre de forma global en el que esta arquitectura supera el valor obtenido por la arquitectura ResNetUNet.

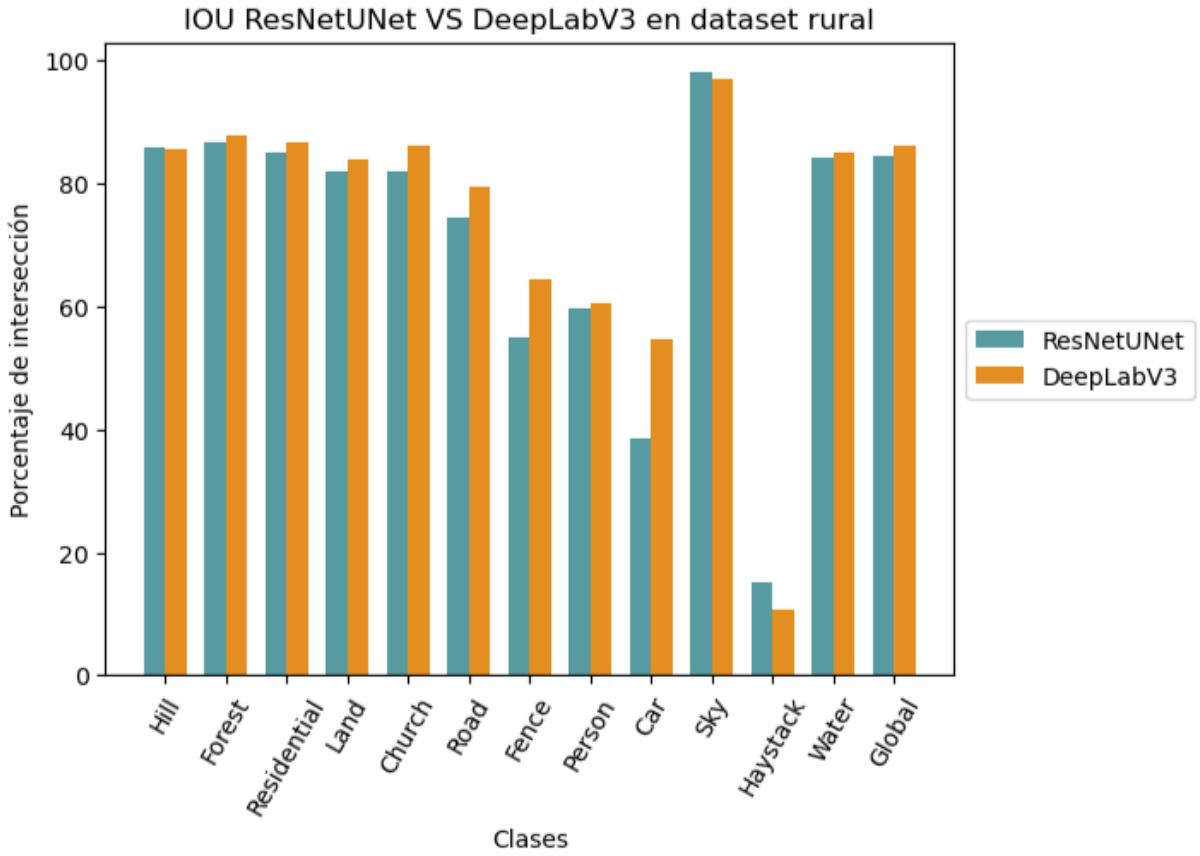

Ilustración 7.11: Comparación de los valores obtenidos en los primeros entrenamientos para la métrica IOU, por clases y globalmente sobre el conjunto de testeo del dataset rural utilizando las arquitecturas ResNetUNet y DeepLabV3.

7.5.2. Resultados obtenidos sobre las clases en el dataset urbano

Habiendo realizado la comparación de las métricas obtenidas para el conjunto de testeo del dataset rural utilizando ambas arquitecturas, ahora se va a proceder a mostrar los valores de las métricas obtenidas para el segundo de los datasets, es decir, el dataset urbano. Por ello, se ha entrenado utilizando los mismos hiperparámetros que se utilizaron para el dataset rural, y se ha entrenado primeramente utilizando la arquitectura ResNetUNet, de esta forma, los resultados que se han obtenido para el conjunto de testeo son los que se figuran en la ilustración 7.12. De la misma manera que se hizo anteriormente, se ha vuelto a entrenar utilizando el mismo conjunto solo que cambiando la arquitectura utilizada, en esta ocasión utilizando DeepLabV3 y con este cambio de arquitectura, los valores que se han obtenido para el conjunto de testeo son los que se muestran en la ilustración 7.13.

Para poder comparar los resultados obtenidos de manera más cómoda, en la ilustración 7.14 se observan las métricas obtenidas para la arquitectura ResNetUNet y DeepLabV3 con el conjunto de testeo del dataset urbano.

En esta ocasión, como se observa en la tabla comparativa mostrada en la ilustración 7.14,

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Clutter	0.7338	0.7010	0.9091	0.7170	0.5589
Building	0.9288	0.9359	0.9563	0.9324	0.8733
Road	0.8212	0.8464	0.9636	0.8336	0.7147
Static_car	0.4426	0.4371	0.9913	0.4398	0.2819
Tree	0.8454	0.8379	0.9239	0.8416	0.7266
Vegetation	0.7649	0.8018	0.9338	0.7829	0.6432
Human	1.0000	0.0000	0.9988	0.0000	0.0000
Moving_car	0.7096	0.4928	0.9950	0.5816	0.4101
Global	0.8359	0.8359	0.9590	0.8359	0.7181

Ilustración 7.12: Métricas obtenidas con el conjunto de testeo sobre el modelo entrenado con el dataset urbano, utilizando la arquitectura ResNetUNet entrenado con 10 épocas.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Clutter	0.7541	0.7570	0.9195	0.7555	0.6071
Building	0.9310	0.9460	0.9601	0.9384	0.8840
Road	0.8440	0.8436	0.9664	0.8438	0.7298
Static_car	0.5706	0.4746	0.9931	0.5182	0.3497
Tree	0.8397	0.8662	0.9278	0.8528	0.7433
Vegetation	0.8280	0.7864	0.9438	0.8067	0.6760
Human	0.0000	0.0000	0.9988	0.0000	0.0000
Moving_car	0.7113	0.3724	0.9945	0.4889	0.3235
Global	0.8520	0.8520	0.9630	0.8520	0.7422

Ilustración 7.13: Métricas obtenidas con el conjunto de testeo sobre el modelo entrenado con el dataset urbano, utilizando la arquitectura DeepLabV3 entrenado con 10 épocas.

se encuentra por filas las distintas clases de las que está compuesto el dataset urbano, en este caso, como ya se había comentado en capítulos anteriores este dataset está formado por un menor número de clases, 8 en total. En la tabla se puede observar como en cada fila aparecen los resultados obtenidos por cada clase con las imágenes de testeo, y como en las columnas aparecen las mismas métricas que se han utilizado anteriormente con las mismas arquitecturas.

7.5.2.1. Análisis de los resultados obtenidos sobre las clases en el dataset urbano

Tal y como se hizo con el anterior análisis, se van a ir observando los resultados obtenidos en la tabla 7.14 clase por clase. Hay que tener en cuenta que al igual que el dataset rural, este dataset urbano tampoco está balanceado, por lo que no será posible fijarse en tan solo la métrica accuracy, sino que hay que tener en cuenta el resto de métricas.

Comenzando por orden de las filas, la primera de las clases es *clutter* esta clase si se recuerda

Nombre Clase	Precisión ResNetUNet	Precisión DeepLabV3	Recall ResNetUNet	Recall DeepLabV3	Accuracy ResNetUNet	Accuracy DeepLabV3	F1 ResNetUNet	F1 DeepLabV3	IOU ResNetUNet	IOU DeepLabV3
Clutter	0.7338	0.7541	0.7010	0.7570	0.9091	0.9195	0.7170	0.7555	0.5589	0.6071
Building	0.9288	0.9310	0.9359	0.9460	0.9563	0.9601	0.9324	0.9384	0.8733	0.8840
Road	0.8212	0.8440	0.8464	0.8436	0.9636	0.9664	0.8336	0.8438	0.7147	0.7298
Static_Car	0.4426	0.5706	0.4371	0.4746	0.9913	0.9931	0.4398	0.5182	0.2819	0.3497
Tree	0.8454	0.8397	0.8379	0.8662	0.9239	0.9278	0.8416	0.8528	0.7266	0.7433
Vegetation	0.7649	0.8280	0.8018	0.7864	0.9338	0.9438	0.7829	0.8067	0.6432	0.6760
Human	0.0000	0.0000	0.0000	0.0000	0.9988	0.9988	0.0000	0.0000	0.0000	0.0000
Moving_Car	0.7096	0.7113	0.4928	0.3724	0.9950	0.9945	0.5816	0.4889	0.4101	0.3235
Global	0.8359	0.8520	0.8359	0.8520	0.9590	0.9630	0.8359	0.8520	0.7181	0.7422

Ilustración 7.14: Comparación de los valores obtenidos para el conjunto de testeo del dataset urbano con las arquitecturas ResNetUNet y DeepLabV3 con entrenamiento de 10 épocas.

del subcapítulo 7.1.2, corresponde con la clase que engloba a todos los píxeles que no forman parte de ninguna de las clases definidas en este dataset. Como precisión de esta clase se ha obtenido un 0.7338 para la arquitectura ResNetUNet, mientras que para la arquitectura DeepLabV3 se ha obtenido un valor de 0.7541. En cuanto a los valores del recall, para el caso de la arquitectura ResNetUNet dicho valor es de 0.7010 mientras que para DeepLabV3 es de 0.7570. Analizando la intersección de las imágenes predichas y las reales, se aprecia que para la arquitectura ResNetUNet se ha obtenido un valor de 0.5589 mientras que para DeepLabV3 el valor asciende a 0.6071. Si bien, se tratan de valores relativamente bajos, pero se recuerda que se trata de una clase que recoge todos los píxeles que no pertenecen a ninguna de las clases presentes en el dataset, por ello, es que se hayan obtenido unos valores tan bajos.

La siguiente de las clases es *building*, esta clase es la que mayor número de píxeles posee en todo el dataset además de aparecer en todas las imágenes del mismo. Como valores obtenidos en la precisión, para la arquitectura ResNetUNet ha sido de 0.9288 mientras que para la arquitectura DeepLabV3 fue de 0.9310. En lo que respecta a los valores de recall se han obtenido 0.9359 y 0.9460 para ResNetUNet y DeepLabV3 respectivamente. A la vista de estos resultados, se observa como se trata de una clase que al poseer un mayor número de píxeles su segmentación mejora considerablemente, los valores son bastante más altos con respecto a la clase anterior y además. Asimismo, los valores obtenidos con la métrica IOU son algo más bajos pero congruentes con lo analizado hasta ahora y es que el 87.33 % de las máscaras predichas intersectan con las máscaras reales para la arquitectura ResNetUNet, mientras que para la arquitectura DeepLabV3 este valor asciende al 88.40 %.

En lo que respecta a la clase *road*, como precisión se ha obtenido un 0.8212 para la arquitectura ResNetUNet y un 0.8440 para DeepLabV3, mientras que como recall se registraron valores de 0.8464 para ResNetUNet y un 0.8436 para DeepLabV3. En esta ocasión, los resultados obtenidos son bastante ajustados, por ello, si se observan los valores obtenidos para la métrica F1 se tiene para ResNetUNet un valor de 0.8336 mientras que para DeepLabV3 se tiene un valor de 0.8438, por tanto, con estos valores más los valores obtenidos para IOU que han sido de 0.7147 y 0.7298 se llega a la conclusión que para esta clase se obtiene unos mejores resultados con la arquitectura DeepLabV3.

La clase *static car* se trata de una clase que posee un número de píxeles inferior al resto de clases que se han observado hasta ahora, puesto que la totalidad de píxeles de esta clase apenas supera el 1 % del total de píxeles del dataset. Esto se traduce en unos valores en las métricas muy bajos, puesto que como valores de precisión se han obtenido 0.4426 y 0.5706 para las arquitecturas ResNetUNet y DeepLabV3, es decir, al ser valores bajos está indicando que en esta clase no se está realizando una buena segmentación, si a esto, además le añadimos los valores obtenidos como recall los cuales han sido de un 0.4371 y un 0.4746, se podría concluir que se trata de una clase con una segmentación bastante inexacta. Además, el porcentaje de intersección entre las máscaras predichas y las reales son de un 28.19 % para ResNetUNet y de un 34.97 % para DeepLabV3, reafirma lo que ya se había deducido, y es que la segmentación de esta clase es bastante mala.

Seguidamente, otra de las clases es *tree*, esta es la segunda de las clases con un mayor número de píxeles totales en el dataset además de aparecer en todas las imágenes del mismo. Esto se traduce que como valores de precisión en las imágenes de testeo se ha obtenido un 0.8454 para ResNetUNet y un 0.8397 para DeepLabV3, mientras que la diferencia entre los valores obtenidos para el recall es algo mayor, debido a que para la arquitectura ResNetUNet se ha obtenido un valor de 0.8379 y para la arquitectura DeepLabV3 dicho valor asciende a un 0.8662. Analizando los valores de la métrica F1 se aprecia como para ResNetUNet se consigue un 0.8416 y como para DeepLabV3 dicho valor es de 0.8528. En este caso, se evidencia un mejor desempeño de la arquitectura DeepLabV3. Por otra parte, si se analizan los valores obtenidos por la métrica IOU se alcanza un valor de 0.7266 para ResNetUNet y un 0.7433 para DeepLabV3, coincidiendo de esta manera como se tienen mejores resultados por poco que sea con la arquitectura DeepLabV3.

La clase *vegetation* ha obtenido como precisión un valor de 0.7649 para ResNetUNet mientras que para DeepLabV3 este valor es de 0.8280. Con respecto al recall, los valores obtenidos son de 0.8018 y de 0.7864 para ResNetUNet y DeepLabV3 respectivamente. En esta ocasión en el recall los valores bajan y aumentan para las arquitecturas ResNetUNet y DeepLabV3 al compararlos con los valores de precisión, por lo que para decidir que arquitectura es mejor en este caso, habrá que fijarse en los valores obtenidos en F1 los cuales son de 0.7829 para la arquitectura ResNetUNet y de 0.8067 para DeepLabV3. De esta manera se puede concluir afirmando que la arquitectura DeepLabV3 da mejores resultados para la segmentación semántica de esta clase. Aunque los valores de la métrica IOU no son muy altos, con 0.6432 para ResNetUNet y 0.6760 para DeepLabV3, confirma que la mejor arquitectura en este caso es la mencionada.

La siguiente de las clases es *human* esta clase es sin duda la que menor número de píxeles posee en el dataset, correspondiendo solamente con el 0.1 % de todo el dataset. Es por ello que en este caso el modelo no es capaz de identificar correctamente ningún pixel de esta clase en las imágenes de testeo. Por otra parte, se puede observar como de accuracy se obtiene para ambas arquitecturas un valor de 0.9988, se reitera una vez más como esta métrica en datasets no balanceados no es efectiva .

La última de las clases de este dataset es *moving car*, se trata también de una clase que aunque aparezca en la mayoría de clases no aparece en todas de las del dataset, asimismo, la cantidad de píxeles que constituyen esta clase sobre el total de píxeles del dataset es de tan

solo un 0.9 %. Por esta razón, los valores que se obtienen en las métricas son bastante bajas. En este sentido, si se fija en los valores obtenidos para F1, se han obtenido valores de 0.5816 para el caso de la arquitectura ResNetUNet, mientras que para la arquitectura DeepLabV3 el valor es de 0.4889, lo que resulta valores sumamente bajos. Con respecto a los valores de IOU, se puede asumir de manera consensuada que ninguno de los modelos está realizando una segmentación correcta de esta clase, ya que los valores de IOU han sido de 0.4101 y 0.3235 para ResNetUNet y DeepLabV3, respectivamente.

7.5.2.2. Análisis de los resultados obtenidos globalmente en el dataset urbano

Por último, al igual que se realizó con el dataset rural, se han obtenido las métricas de forma global teniendo en cuenta todas las clases del dataset urbano. Al igual que ocurría en el dataset anterior, al ser los valores globales y tener en cuenta todas las clases anteriores, los valores de precisión, recall y F1 son exactamente iguales, en este caso, dichos valores son de 0.8359 para la arquitectura ResNetUNet, mientras que para la arquitectura DeepLabV3 el valor es de 0.8520. De esta manera, se observa como se obtienen unos mejores resultados para la arquitectura DeepLabV3, además, si se observan los valores obtenidos para IOU, los cuales son de 0.7181 y de 0.7422 para ResNetUNet y DeepLabV3 respectivamente, se llega a la misma conclusión, y es que para este dataset también se obtienen mejores resultados en las métricas cuando el modelo ha sido entrenado con la arquitectura DeepLabV3. En este caso, al igual que se hizo anteriormente, se va a visualizar en forma de gráfica los valores obtenidos para la intersección de las clases y de forma global para este dataset haciendo uso de las dos arquitecturas, esta gráfica es la que se observa en la ilustración 7.15

En la ilustración 7.15 se observa como 6 de las 8 clases presentes en este dataset urbano poseen mejores valores de intersección cuando el modelo ha sido entrenado con la arquitectura DeepLabV3. Sin embargo, en una de las clases, concretamente en la clase *human*, no se tienen mejores ni peores resultados simplemente no se obtienen intersección alguna, por ello, los valores de las barras de ambas arquitecturas son nulas. Finalmente, destacar como de forma global de observa como se obtienen mejores resultados al utilizar la arquitectura DeepLabV3.

7.5.3. Resultados generales preliminares

Como se ha observado previamente, para los entrenamientos de ambos datasets, se han obtenido mejores resultados en las métricas cuando se utilizó la arquitectura DeepLabV3. Asimismo, hay que recalcar la importancia de utilizar distintas métricas para tomar una decisión como puede ser el de elegir la mejor arquitectura para la segmentación de un dataset, y es que si las clases del dataset no están balanceadas, como es el caso de ambos datasets aquí utilizados, la métrica de accuracy puede no estar dando resultados precisos, como se explicó anteriormente, por lo tanto, es recomendable utilizar distintas métricas para comprobar la efectividad de un modelo.

Estos resultados obtenidos se trataron de una buena primera aproximación al objetivo planteado. Sin embargo, como indicaron mis dos tutores, tener un modelo que se le tiene que

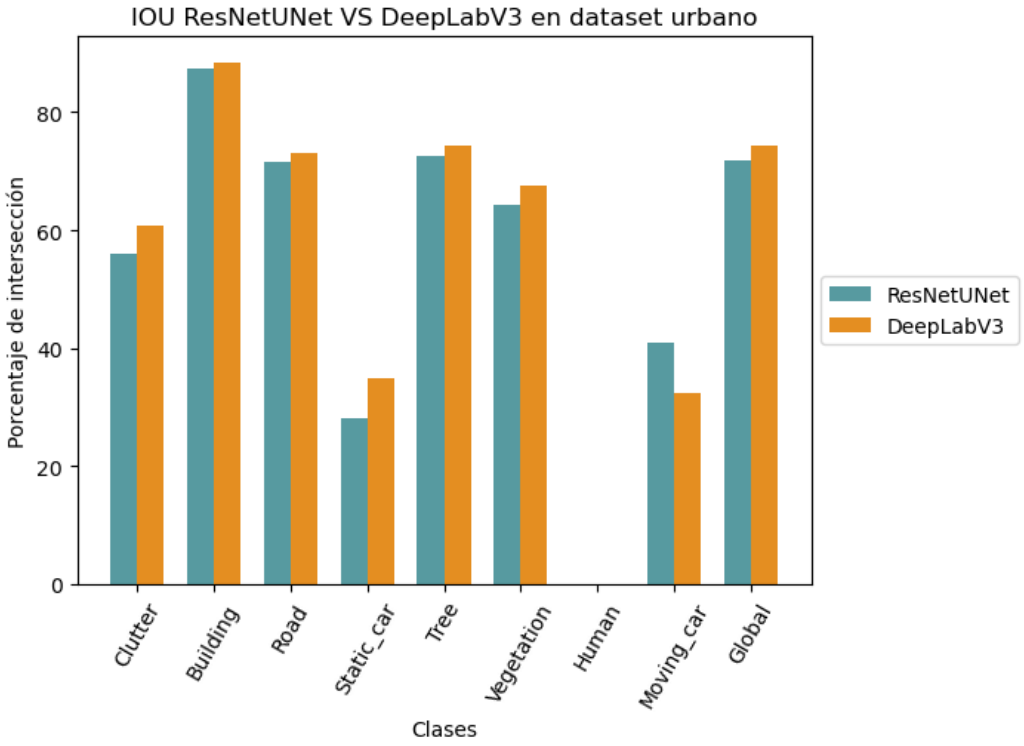

Ilustración 7.15: Comparación de los valores obtenidos en los primeros entrenamientos para la métrica IOU por clases y globalmente sobre el conjunto de testeo del dataset urbano utilizando las arquitecturas ResNetUNet y DeepLabV3.

pasar imágenes de una dimensión de 1376x736 no es la mejor idea. Además, tener unos *batches* de tamaño tan pequeño no es lo ideal por varias razones, y es que la GPU no se está aprovechando en cada iteración en la que se obtiene un *batch*. Asimismo, con *batches* más grandes se tiene una mayor representación de la distribución de datos, lo que puede hacer que el entrenamiento del modelo sea más estable y menos propenso a fluctuaciones debido al gradiente, y así como una generalización más rápida de los nuevos datos.

De la misma manera, establecer una cantidad de épocas fijas no sería la mejor de las opciones, y es que al hacer esto se está asumiendo erróneamente que el mejor modelo siempre se encontrará en la época 10 cuando esto no debe de ser así. Para ello, se establecerá un número máximo de épocas, 50 en esta ocasión, este valor se trata de un valor por el cual se conoce que los modelos que se van a entrenar tendrán un número menor de épocas. Seguidamente se ha establecido un criterio de parada del entrenamiento, y se trata que si el modelo no mejora su coste de validación en tres épocas consecutivas, se asume que el modelo ya no es capaz de aprender más características y por tanto ha comenzado el sobreentrenamiento, sabiendo esto, el modelo que posee el coste de validación más bajo será el modelo almacenado tres épocas antes de la parada del entrenamiento. En cuanto al tamaño de las imágenes, se ha considerado que un tamaño de 480x256 es un tamaño de imagen apropiado para el tipo de segmentación que se está realizando. Al disminuir el tamaño de imágenes considerablemente, la cantidad de imágenes de cada *batch* podría aumentar, por lo que se realizaron pruebas

para observar cual era el tamaño de *batch* que podía almacenar la GPU sin que se quedara sin memoria en la misma, y este valor es de 13 imágenes por *batch*, por lo que de este modo, se ha elegido este valor como tamaño de *batch*.

7.6. Mejorando los entrenamientos

Con las modificaciones mencionadas, es decir, implementación de parada automática, disminución del tamaño de las imágenes y aumento del tamaño de *batch*, se ha procedido a entrenar nuevamente los modelos nombrados anteriormente, para de esta forma realizar una comparación justa sabiendo que los modelos comparados son los que minimizan el coste de validación.

7.6.1. Resultados obtenidos sobre las clases en el dataset rural

Por ello, primero se comenzará mostrando los resultados de testeo habiendo entrenado con la arquitectura ResNetUNet con el conjunto de testeo del dataset rural. En este caso, el entrenamiento concluyó en que la época 23 se alcanzó el costo de validación más bajo. Los resultados obtenidos con el conjunto de testeo se muestran en la ilustración 7.16. Mientras que las métricas obtenidas para el mismo conjunto y dataset pero utilizando DeepLabV3, se llegó a que la mejor época fue la 23 al igual que con la arquitectura ResNetUNet, y en esta ocasión las métricas que se obtuvieron sobre el conjunto de testeo fueron las que se muestran en la ilustración 7.17.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Hill	0.9137	0.9194	0.9901	0.9166	0.8460
Forest	0.9275	0.9314	0.9451	0.9294	0.8681
Residential	0.9007	0.9312	0.9658	0.9157	0.8445
Land	0.9038	0.8853	0.9576	0.8944	0.8090
Church	0.9250	0.8907	0.9962	0.9075	0.8307
Road	0.8179	0.8311	0.9888	0.8245	0.7013
Fence	0.7682	0.6636	0.9917	0.7121	0.5529
Person	0.7725	0.6942	0.9995	0.7312	0.5764
Car	0.7409	0.3039	0.9983	0.4310	0.2747
Sky	0.9840	0.9919	0.9983	0.9879	0.9761
Haystack	0.7719	0.0433	0.9991	0.0821	0.0428
Water	0.9681	0.9049	0.9989	0.9354	0.8787
Global	0.9147	0.9147	0.9858	0.9147	0.8429

Ilustración 7.16: Métricas obtenidas con el conjunto de testeo del dataset rural sobre el modelo entrenado utilizando la arquitectura ResNetUNet entrenado con 23 épocas.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Hill	0.9086	0.9244	0.9900	0.9164	0.8458
Forest	0.9416	0.9247	0.9480	0.9330	0.8745
Residential	0.8878	0.9342	0.9642	0.9104	0.8355
Land	0.9062	0.9049	0.9612	0.9056	0.8275
Church	0.9234	0.9170	0.9964	0.9202	0.8522
Road	0.8694	0.8395	0.9913	0.8542	0.7455
Fence	0.7870	0.7385	0.9932	0.7620	0.6155
Person	0.7614	0.7070	0.9995	0.7332	0.5788
Car	0.7746	0.5693	0.9987	0.6563	0.4884
Sky	0.9803	0.9840	0.9975	0.9822	0.9649
Haystack	0.9193	0.0873	0.9991	0.1595	0.0867
Water	0.9636	0.9014	0.9991	0.9314	0.8717
Global	0.9190	0.9190	0.9865	0.9190	0.8502

Ilustración 7.17: Métricas obtenidas con el conjunto de testeo del dataset rural sobre el modelo entrenado utilizando la arquitectura DeepLabV3 entrenado con 23 épocas.

En la ilustración 7.18 se observa la tabla resumen con los valores obtenidos utilizando ambas arquitecturas.

7.6.1.1. Análisis de los resultados obtenidos sobre las clases en el dataset rural

Nombre Clase	Precisión ResNetUNet	Precisión DeepLabV3	Recall ResNetUNet	Recall DeepLabV3	Accuracy ResNetUNet	Accuracy DeepLabV3	F1 ResNetUNet	F1 DeepLabV3	IOU ResNetUNet	IOU DeepLabV3
Hill	0.9137	0.9086	0.9194	0.9244	0.9901	0.9900	0.9166	0.9164	0.8460	0.8458
Forest	0.9275	0.9416	0.9314	0.9247	0.9451	0.9480	0.9294	0.9330	0.8681	0.8745
Residential	0.9007	0.8878	0.9312	0.9342	0.9658	0.9642	0.9157	0.9104	0.8445	0.8355
Land	0.9038	0.9062	0.8853	0.9049	0.9576	0.9612	0.8944	0.9056	0.8090	0.8275
Church	0.9250	0.9234	0.8907	0.9170	0.9962	0.9964	0.9075	0.9202	0.8307	0.8522
Road	0.8179	0.8694	0.8311	0.8395	0.9888	0.9913	0.8245	0.8542	0.7013	0.7455
Fence	0.7682	0.7870	0.6636	0.7385	0.9917	0.9932	0.7121	0.7620	0.5529	0.6155
Person	0.7725	0.7614	0.6942	0.7070	0.9995	0.9995	0.7312	0.7332	0.5764	0.5788
Car	0.7409	0.7746	0.3039	0.5693	0.9983	0.9987	0.4310	0.6563	0.2747	0.4884
Sky	0.9840	0.9803	0.9919	0.9840	0.9983	0.9975	0.9879	0.9822	0.9761	0.9649
Haystack	0.7719	0.9193	0.0433	0.0873	0.9991	0.9991	0.0821	0.1595	0.0428	0.0867
Water	0.9681	0.9636	0.9049	0.9014	0.9989	0.9991	0.9354	0.9314	0.8787	0.8717
Global	0.9147	0.9190	0.9147	0.9190	0.9858	0.9865	0.9147	0.9190	0.8429	0.8502

Ilustración 7.18: Comparación de las métricas del conjunto de testeo del dataset rural con las arquitecturas ResNetUNet y DeepLabV3 con 23 épocas respectivamente.

Como se observa en la tabla comparativa que se muestra en la ilustración 7.18 se han obtenido las mismas métricas que se habían obtenido para el caso anterior y utilizando las mismas

arquitecturas, por ello, se va a a realizar una comparación entre los resultados obtenidos en esta ocasión con los datos obtenidos anteriormente. En esta ocasión, se va a centrar en el estudio de las métricas F1, ya que proporciona información suficiente sobre la precisión y el recall. Además, también se tendrán en cuenta la métrica IOU.

Comenzando por la clase *hill*, en la que se puede apreciar como los valores han cambiado con respecto al entrenamiento anterior, ahora ResNetUNet posee mayor precisión pero menor recall, mientras que con DeepLabV3 ocurre lo contrario. Esto se traduce en la métrica F1 en donde en ambas arquitecturas los valores disminuyen sutilmente, 0.9166 y 0.9164 frente a los 0.9233 y 0.9225 obtenidos anteriormente para ResNetUNet y DeepLabV3 respectivamente. Además, los valores obtenidos para la métricas IOU también han disminuido, puesto a que ahora se tienen valores por debajo del 85 % frente a los valores del entrenamiento anterior que se encontraban por encima de este tanto %. En este sentido, se observa la no mejoría de la segmentación semántica de esta clase en el dataset de testeo.

Siguiendo con el orden de aparición de las distintas clases en la tabla comparativa, ahora se va a analizar la clase *forest*. En esta ocasión, los valores de precisión y recall obtenidos para la arquitectura ResNetUNet mejoran con respecto a la ejecución anterior, viéndose reflejado en el valor obtenido en la métrica F1 que ha sido de 0.9294 frente al 0.9284 obtenido en la ejecución anterior. Sin embargo, los valores de precisión y recall obtenidos para la arquitectura DeeplabV3 no han mejorado, y esto se refleja en un menor valor de F1, en esta ocasión, dicho valor es de 0.9330 frente al 0.9351 anterior. Esta tendencia también se ve reflejada en la métrica IOU en donde hay una mejora con la arquitectura ResNetUNet pero no así con la arquitectura DeepLabV3. Con este entrenamiento ha mejorado la segmentación semántica con la arquitectura ResNetUNet pero no con la otra de las arquitecturas.

La clase *residential* posee valores sutilmente menores tanto de precisión como de recall en ambas arquitecturas. De esta forma, lo valores obtenidos en la métrica F1 también lo son, puesto que anteriormente se habían obtenido valores de 0.9195 y 0.9283 y ahora estos valores han sido de 0.9157 y 0.9104 para las arquitecturas ResNetUNet y DeepLabV3 respectivamente. A su vez, los valores de IOU también han sido algo menores, apreciando así como no existe aún una mejoría en los resultados.

Lo mismo que ha ocurrido con la clase *residential* también ha ocurrido con la clase *land* y es que los valores que se han obtenido para las métricas de esta ejecución tampoco han mejorado con respecto a las obtenidas anteriormente y es que para la métrica F1 los valores obtenidos han sido de 0.8944 y 0.9056 frente a los 0.9014 y 0.9133 obtenidos en la ejecución anterior con las arquitecturas ResNetUNet y DeepLabV3 respectivamente.

En el caso de la clase *church*, se han obtenido mejores resultados en esta ocasión en la precisión de la arquitectura ResNetUNet, aunque en la precisión de la otra de las arquitecturas y en el recall de ambas los valores han fluctuado a la baja. La ganancia que se ha obtenido en la precisión de ResNetUNet y la poca disminución del recall, ha hecho que en esta ocasión el valor de la métrica F1 aumente llegando a los 0.9075 frente a los 0.9004 obtenido en la ejecución anterior para la arquitectura ResNetUNet. Con respecto al valor F1 para DeepLabV3 el valor ha pasado de 0.9260 al 0.9202, se trata al igual que ha ocurrido con las clases anteriores de una disminución bastante baja. Mientras que con la métrica IOU se ha obtenido un mejor

resultado en IOU para la arquitectura ResNetUNet, ya que el valor ha pasado de 0.8189 a 0.8307, mientras que para DeepLabV3 el valor ha disminuido pasando de 0.8621 a 0.8522.

Continuando con la clase *road*, se trata de la clase que más ha empeorado en esta nueva ejecución, y es que la segmentación que realiza esta clase ahora es peor que la obtenida anteriormente. Esto se ve reflejado en la métrica F1 en donde los valores en esta nueva ejecución han bajado hasta un 0.0300 con respecto a la ejecución anterior, obviamente, esto ha hecho que los valores obtenidos para la métrica IOU hayan bajado hasta casi un 0.0500 mostrando un empeoramiento de los resultados.

La siguiente de las clases es la clase *fence*, en esta ocasión, los valores obtenidos indican que han mejorado para la arquitectura ResNetUNet, ya que como se observa en la tabla, en cuanto a la precisión ha pasado de 0.7280 a 0.7682, mientras que en la arquitectura DeepLabV3 ha disminuido de 0.8134 a 0.7870. Con respecto al recall en ambos casos los valores han disminuido respecto a la ejecución anterior. Esto hace que los valores de F1 obtenidos pasen de 0.7098 a 0.7121 para el caso de la arquitectura ResNetUNet, es decir, se posee una mejora, mientras que para DeepLabV3 se pasa de 0.7827 a 0.7620. Claramente, con estos resultados, a la vista estaba de lo que ocurriría con la métrica IOU y es que se ha obtenido una mejor intersección con la arquitectura ResNetUNet que con la arquitectura DeepLabV3, ya que en el primero de los casos ha pasado de tener un 55,01 % de intersección a 55,29 %.

La clase *person* es una de las clases de las que no se han obtenido unos mejores resultados con respecto a la ejecución anterior, y es que como precisión se han obtenido en este caso 0.7725 y 0.7614 frente a los 0.7055 y 0.7765 obtenidos anteriormente para las arquitecturas ResNetUNet y DeepLabV3, si bien aunque la precisión de DeepLabV3 no ha mejorado, si lo ha hecho considerablemente la otra de las arquitecturas. Aunque con la métrica recall ha sucedido lo contrario, y es que ha disminuido el valor para la arquitectura ResNetUNet y ha aumentado para DeepLabV3, ya que los valores obtenidos en esta ejecución han sido de 0.6942 y de 0.7070 mientras que en la ejecución anterior dichos valores eran de 0.7933 y de 0.7344 para ResNetUNet y DeepLabV3 respectivamente. Esto implica que tanto los valores de F1 como lo de IOU disminuyan con respecto a la ejecución anterior.

Análogamente la clase *car* es otra de las clases las cuales no llega a mejorar en ninguna de las métricas con respecto a la ejecución anterior, por ejemplo, si se observan los valores obtenidos en la métrica F1, se aprecia que en esta ocasión los valores han sido de 0.4310 y de 0.6563 para ResNetUNet y DeepLabV3 respectivamente, mientras que los valores obtenidos en la ejecución anterior han sido de 0.5551 y de 0.7072 para las mismas arquitecturas nombradas, visualizando un empeoramiento de los datos, y extrapolándose los mismos a la métrica IOU.

Asimismo, con la clase *sky* se ha mejorado la precisión con la arquitectura DeepLabV3, aumentando de 0.9770 a 0.9803. Sin embargo, en cuanto al recall, los valores han bajado, aunque esto no afecta significativamente la calidad de la segmentación semántica, ya que en el caso de la arquitectura ResNetUNet sigue superando el 99 % mientras que para DeepLabV3 continúa superando el 98 %. Esto a su vez, hace que los valores de F1 tanto de ResNetUNet como en DeepLabV3 disminuyan sutilmente, y es que se ha pasado de 0.9900 y de 0.9844 a valores de 0.9879 y 0.9822, aunque en esta ocasión existe una disminución, no implica un

importante mal funcionamiento del modelo. Lo mismo ocurre con los valores de la métrica IOU, los cuales han bajado relativamente con respecto a los de la ejecución anterior, pero siguen siendo más que aceptables en términos de calidad de segmentación, puesto que se recuerda que se tratan de testeo con imágenes con las que no ha sido ni entrenado ni validado el modelo.

Esta nueva ejecución confirma nuevamente que la clase *haystack* no está siendo segmentada correctamente, y es que en este caso se obtienen unos mejores resultados de precisión como lo son los valores 0.7719 y 0.9193 frente a los 0.6280 y 0.6706 obtenidos anteriormente para las arquitecturas ResNetUNet y DeepLabV3 respectivamente. Sin embargo, estos altos valores obtenidos en la precisión van asociados a una disminución muy importante y considerable del recall, y es que en esta ocasión se han obtenido valores de 0.0433 y 0.0873 frente a los valores 0.1673 y de 0.1126 obtenidos en la ejecución anterior para las arquitecturas ResNetUNet y DeepLabV3. Esto implica que los valores tanto de F1 como de IOU sean extremadamente bajos como para afirmar que esta clase es incapaz de ser segmentada semánticamente por ninguno de los dos modelos aquí planteados.

Por último se encuentra la clase *water*, esta clase ha mostrado mejoras en todos los valores de las métricas en comparación con la ejecución anterior, reflejándose así en los valores de F1 que se han situado en 0.9354 y 0.9314 en comparación con los valores anteriores de 0.9144 y 0.9190 para ResNetUNet y DeepLabV3 respectivamente. Además, como valores de IOU se ha pasado de intersectar en esta clase el 84 % y 85 % para ResNetUNet y DeepLabV3 a más del 87 % en ambos casos. Como se puede observar, en este caso se obtiene una mejora significativa de la segmentación semántica de esta clase.

7.6.1.2. Análisis de los resultados obtenidos globalmente en el dataset rural

Finalmente, al igual que en la tabla comparativa 7.10 de la ejecución anterior, en esta ocasión también existe una última fila la cual recoge de manera global el resultado de las métricas teniendo en cuenta todas las clases. Por lo tanto, ahora se va a analizar estos resultados en comparación con los obtenidos inicialmente. Recordando que, los valores de precisión, recall y F1 son siempre iguales por las razones que ya se explicaron anteriormente. En esta ejecución se han obtenido valores de 0.9147 y 0.9190 para precisión, recall y F1 usando las arquitecturas ResNetUNet y DeepLabV3 respectivamente. Mientras que en la ejecución anterior, se obtuvieron para las mismas métricas los valores 0.9164 y 0.9264, esto quiere decir que en ambos casos se poseen unos mejores resultados en la primera de las ejecuciones del entrenamiento, aunque es importante destacar que la diferencia entre estos valores es bastante pequeña, puesto que para la arquitectura ResNetUNet la diferencia apenas es de 0.0017, mientras que la diferencia de la arquitectura DeepLabV3 es de 0.0074, es decir, se tratan de unas diferencias bastante mínimas. Además, en cuanto a los valores obtenidos en la métrica IOU, en este caso se han obtenido los valores 0.8429 y 0.8502 para ResNetUNet y DeepLabV3 respectivamente, mientras que en la ejecución anterior estos resultados eran de 0.8458 y 0.8628, esto implica una diferencia de 0.0029 y de 0.0126, por tanto, aunque se tienen mejores resultados con la primera de las ejecuciones del entrenamiento, los cambios son mínimos en ambos casos.

Al igual que se hizo anteriormente, en la ilustración 7.19 se observa un diagrama de barras en el que se aprecia como se tienen para cada clase y de forma global dos barras distintas siendo estas correspondientes a las dos arquitecturas que se han analizado.

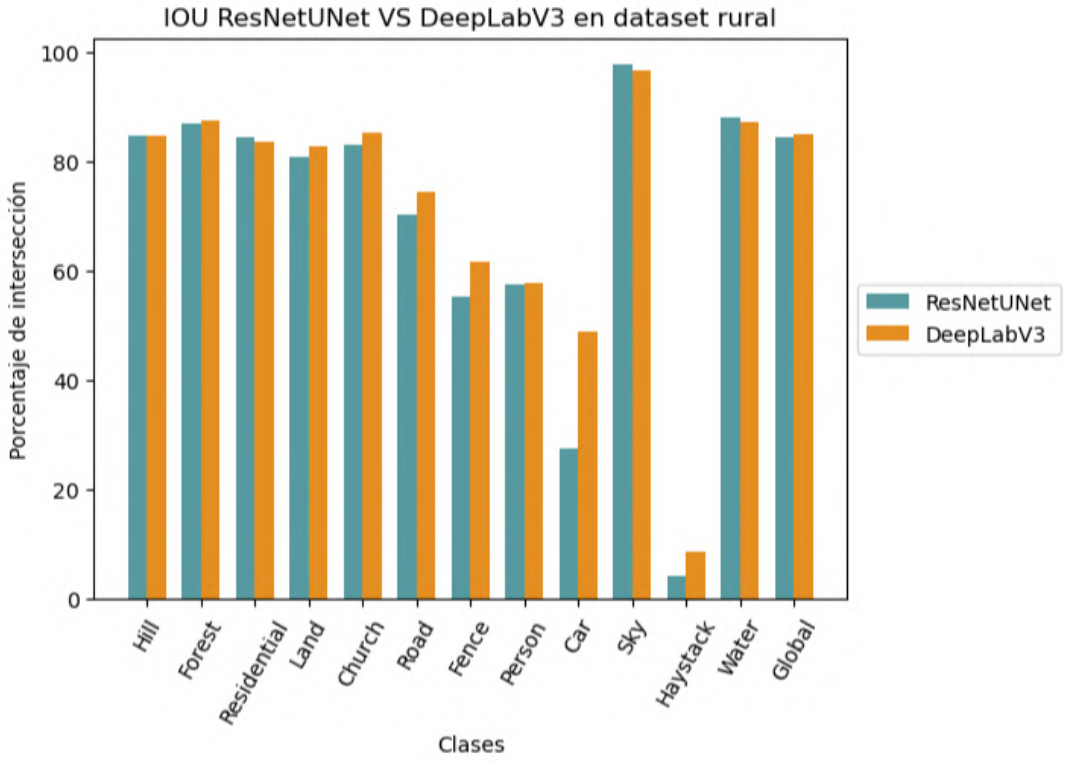

Ilustración 7.19: Comparación de los valores obtenidos para la métrica IOU por clases y globalmente sobre el conjunto de testeo del dataset rural utilizando las arquitecturas ResNetUNet y DeepLabV3.

En este caso, analizando la gráfica presente en la ilustración 7.19 se aprecia como se poseen unos valores de forma general más altos en las clases en las que se hace uso de la arquitectura DeepLabV3, tan solo en dos de las clases se observa como existe una ligera disminución de los valores de intersección. Pero tal y como se ha ido observando con los valores globales anteriormente, se obtiene una mejor intersección de forma global cuando se hace uso de la arquitectura DeepLabV3. Por tanto, de esta manera se puede afirmar que para este dataset, esta arquitectura proporciona unos mejores resultados.

En este caso, aunque para el conjunto de testeo analizado se han tenido peores resultados en esta ocasión, este entrenamiento realmente es el que se ha realizado correctamente. Ya que es el que minimiza el coste de validación, el que hace uso de un tamaño de imagen más adecuado y el que aprovecha al máximo la GPU en la que se realiza el entrenamiento al ocupar toda la memoria disponible en cada *batch*. Aún así, analizando tan solo los valores de esta ejecución, se observa como claramente se obtienen resultados de métricas más altos cuando se utiliza la arquitectura DeepLabV3, viéndose como de esta manera, al igual que en los dos entrenamientos anteriores, tanto con el dataset rural como en el urbano, esta arquitectura parece ser mejor que la arquitectura ResNetUNet.

En la ilustración 7.20 se pueden observar algunas predicciones sobre el conjunto de testeo del dataset rural con el modelo que mejores resultados ha proporcionado, es decir, con la arquitectura DeepLabV3 que acaba de ser analizada. Para ello, en la ilustración se muestran por filas las distintas imágenes, seguidas de su correspondiente máscara y finalmente la máscara predicha por el modelo.

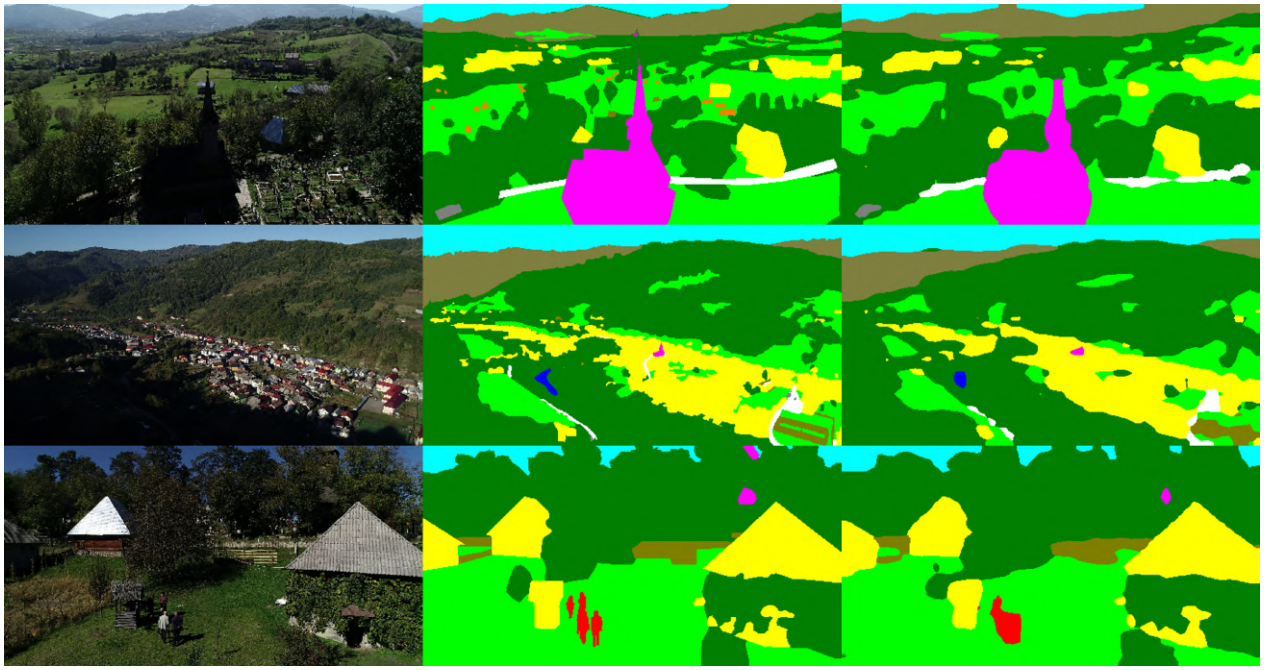

Ilustración 7.20: Algunas de las predicciones realizadas por el modelo con la arquitectura DeepLabV3 con el conjunto de testeo del dataset rural. En el margen izquierdo se encuentran las imágenes reales, en la parte central las correspondientes máscaras y en el margen derecho se encuentran las predicciones.

7.6.2. Resultados obtenidos sobre las clases en el dataset urbano

Una vez teniendo los resultados obtenidos para el dataset rural, se va a realizar el mismo procedimiento pero utilizando el dataset urbano. Por ello, se ha entrenado el modelo utilizando primeramente la arquitectura ResNetUNet, en este caso, el mejor modelo del entrenamiento se ha obtenido en la época 16, y se han obtenido las métricas que aparecen en la ilustración 7.21. Y en la ilustración 7.22, se observan las métricas obtenidas con el dataset urbano pero haciendo uso de la arquitectura DeepLabV3, cuyo mejor modelo lo obtuvo en la época 27. En la ilustración 7.23 se observa la tabla comparativa de ambas arquitecturas con el dataset urbano.

Siguiendo los mismos pasos que se han realizado anteriormente, ahora se van a analizar los resultados obtenidos con este segundo entrenamiento para el dataset urbano. Para ello, al igual que se hizo con el segundo entrenamiento del dataset rural se va a analizar más en profundidad los resultados de las métricas F1 e IOU con los obtenidos para este mismo dataset pero en la primera ejecución del entrenamiento.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Clutter	0.7851	0.7279	0.9085	0.7554	0.6069
Building	0.8979	0.9340	0.9515	0.9156	0.8443
Road	0.8363	0.8756	0.9639	0.8555	0.7475
Static_car	0.5230	0.4404	0.9910	0.4782	0.3142
Tree	0.8106	0.8803	0.9186	0.8440	0.7301
Vegetation	0.8075	0.6829	0.9369	0.7400	0.5873
Human	0.0000	0.0000	0.9988	0.0000	0.0000
Moving_car	0.6732	0.5717	0.9931	0.6183	0.4475
Global	0.8311	0.8311	0.9578	0.8311	0.7111

Ilustración 7.21: Métricas obtenidas con el conjunto de testeo del dataset urbano sobre el modelo entrenado utilizando la arquitectura ResNetUNet entrenado con 16 épocas.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Clutter	0.8506	0.7959	0.9332	0.8224	0.6983
Building	0.9283	0.9555	0.9676	0.9417	0.8898
Road	0.8620	0.8812	0.9676	0.8715	0.7723
Static_car	0.6673	0.5577	0.9935	0.6076	0.4364
Tree	0.8562	0.8724	0.9286	0.8642	0.7609
Vegetation	0.7793	0.7905	0.9451	0.7849	0.6459
Human	0.0000	0.0000	0.9989	0.0000	0.0000
Moving_car	0.6440	0.4932	0.9919	0.5586	0.3875
Global	0.8632	0.8632	0.9658	0.8632	0.7594

Ilustración 7.22: Métricas obtenidas con el conjunto de testeo del dataset urbano sobre el modelo entrenado utilizando la arquitectura DeepLabV3 entrenado con 27 épocas.

7.6.2.1. Análisis de los resultados obtenidos sobre las clases en el dataset urbano

En esta ocasión, en la tabla 7.23 la primera de las clases es la clase *clutter*. En esta ocasión existe una importante mejoría de los resultados obtenidos, y es que tanto la precisión como el recall poseen valores mayores, implicando un aumento en la precisión de más de 0.05 y de casi un 0.1 para las arquitecturas ResNetUNet y DeepLabV3. Asimismo, los valores de recall aumentan pasando de 0.7010 y de 0.7570, a valores de 0.7279 y 0.7959, mostrando así un aumento significativo. Estos valores de precisión y recall se reflejan directamente en los valores de las métricas de F1 e IOU, en la primera se pasan de valores de 0.7170 y 0.7555 a valores de 0.7554 y 0.8224 para ResNetUNet y DeepLabV3. Mientras que para IOU, se pasa de intersectar para esta clase del 55.89 % y del 60.71 %, al 60.69 % 69.83 %. Se observa como para esta clase el aumento de intersección y de segmentación semántica es considerablemente superior al que se tenía en un primer entrenamiento, teniendo además un mayor aumento en los valores de las métricas asociadas a la arquitectura DeepLabV3.

Nombre Clase	Precisión ResNetUNet	Precisión DeepLabV3	Recall ResNetUNet	Recall DeepLabV3	Accuracy ResNetUNet	Accuracy DeepLabV3	F1 ResNetUNet	F1 DeepLabV3	IOU ResNetUNet	IOU DeepLabV3
Clutter	0.7851	0.8506	0.7279	0.7959	0.9085	0.9332	0.7554	0.8224	0.6069	0.6983
Building	0.8979	0.9283	0.9340	0.9555	0.9515	0.9676	0.9156	0.9417	0.8443	0.8898
Road	0.8363	0.8620	0.8756	0.8812	0.9639	0.9676	0.8555	0.8715	0.7475	0.7723
Static_Car	0.5230	0.6673	0.4404	0.5577	0.9910	0.9935	0.4782	0.6076	0.3142	0.4364
Tree	0.8106	0.8562	0.8803	0.8724	0.9186	0.9286	0.8440	0.8642	0.7301	0.7609
Vegetation	0.8075	0.7793	0.6829	0.7905	0.9369	0.9451	0.7400	0.7849	0.5873	0.6459
Human	0.0000	0.0000	0.0000	0.0000	0.9988	0.9989	0.0000	0.0000	0.0000	0.0000
Moving_Car	0.6732	0.6440	0.5717	0.4932	0.9931	0.9919	0.6183	0.5586	0.4475	0.3875
Global	0.8311	0.8632	0.8311	0.8632	0.9578	0.9658	0.8311	0.8632	0.7111	0.7594

Ilustración 7.23: Comparación de las métricas del conjunto de testeo del dataset urbano con las arquitecturas ResNetUNet y DeepLabV3 entrenados con 16 y 27 épocas respectivamente.

La siguiente de las clases es la clase *building*, en este caso, en cuanto a la precisión de esta clase los valores han disminuido sutilmente en comparación con el entrenamiento anterior, sin embargo, el recall, aumenta dicho valor para la arquitectura DeepLabV3, en donde se pasa de un 0.9460 a 0.9555. Este aumento del recall se refleja para esta misma arquitectura en el valor de la métrica F1, en donde se pasa del 0.9384 a 0.9417, sin embargo para el caso de ResNetUNet este valor disminuye. Además, en cuanto al valor de IOU, se pasa de intersectar para esta clase el 88.40 % de los píxeles al 88.98 %, si bien el aumento no es significativamente mayor, es un valor que aporta una idea acerca de la mejora que tiene la arquitectura DeepLabV3 para esta clase en este nuevo entrenamiento.

Con respecto a la clase *road*, se trata de una clase la cual no tuvo una buena segmentación semántica a la vista de los resultados de las métricas obtenidas en la anterior ejecución. Sin embargo, ahora se han obtenido unos mejores resultados para las métricas. La precisión ha representado un aumento de aproximadamente un 0.01. Mientras que para los valores obtenidos del recall, han aumentado más de un 0.03 para ambas arquitecturas. Esto se traduce en mejoras de los resultados obtenidos para la métrica F1, en donde se ha pasado de tener un 0.8336 y 0.8438 a valores de 0.8555 y 0.8715 para ResNetUNet y DeepLabV3 respectivamente. Obviamente, el aumento de estos valores hace que la intersección de los píxeles de esta clase aumente, pasando de un 71,47 % y 72,98 %, a poseer una intersección del 74,75 %, y 77,23 % para ResNetUNet y DeepLabV3 respectivamente, obteniendo un aumento considerable en la arquitectura DeepLabV3.

La siguiente de las clases es *static car*, esta clase corresponde con una de las dos clases que peores resultados de segmentación semántica proporcionaron en la anterior ejecución. En este caso, observando los valores de la métrica F1, se ha pasado de 0.4398 y 0.5182 a valores de 0.4782 y 0.6076 para las arquitecturas ResNetUNet y DeepLabV3, es decir, el aumento ha sido más notable para la segunda de las arquitecturas. En cuanto a la intersección, se pasa de intersectar el 28,19 % y el 34,97 % para ResNetUNet y DeepLabV3 a intersectar el 31,42 % y el 43,64 % para las mismas arquitecturas, es decir, en el caso de DeepLabV3 el aumento de intersección es de aproximadamente un 9 %. Si bien, siguen siendo unos valores relativamente

bajos para considerar que se trata de una buena segmentación semántica de estas imágenes, se puede afirmar que aún así los datos obtenidos son mejores con la arquitectura DeepLabV3 en este segundo entrenamiento.

La clase *tree*, obtuvo unos buenos resultados en la primera de las ejecuciones con este dataset. Si bien ahora los datos han mejorado de manera general, viéndose reflejado en los valores de F1 obtenidos, y es que para el caso de ResNetUNet se ha obtenido en el primero de los entrenamientos un valor de 0.8416 y ahora este valor es de un 0.8440, aunque no se trata de un elevado aumento, es más que suficiente para indicar que este nuevo entrenamiento mejora los resultados para esta arquitectura, además, también lo hace para la arquitectura DeepLabV3 en donde se pasa de 0.8528 a un 0.8642. Este cambio parece lineal con el obtenido en la métrica IOU, en donde se han obtenido valores de 72,66 % y 74,33 % para las arquitecturas ResNetUNet y DeepLabV3 en el primer entrenamiento, y estos valores ahora aumentan llegando hasta el 73,01 % y 76,09 %. Aunque se han tenido mejores resultados en ambas arquitecturas, es evidente que con la arquitectura DeepLabV3, los valores obtenidos han sido mejores.

Por su parte, la clase *vegetation* ha aumentado la precisión para la arquitectura ResNetUNet pero no así para DeepLabV3, justo el caso contrario de lo que ocurrió en la métrica recall, donde se ha reducido para ResNetUNet pero ha aumentado para DeepLabV3. El aumento de las respectivas métricas no ha sido suficiente para aportar mejores resultados en las métricas F1 e IOU, en donde para la primera de las métricas se ha pasado de poseer unos valores de 0.7829 y 0.8067 para las arquitecturas ResNetUNet y DeepLabV3 a tener en esta ejecución valores de 0.7400 y 0.7849 para las mismas arquitecturas, como se observa, los valores obtenidos en ambos casos son menores. Lo mismo ocurre con IOU, en donde se tenían valores de 0.6432 y 0.6760, ahora se poseen valores de 0.5873 y 0.6459, que como bien se adelantaba con anterioridad, son valores menores a los obtenidos en el entrenamiento previo.

En cuanto a la clase *human* ocurre exactamente lo mismo que ocurría con el primero de los entrenamientos, y es que los valores obtenidos son iguales en todas las métricas, es decir, de 0 a excepción del accuracy, por lo ya indicado con anterioridad. Esto se debe a que los modelos que han sido entrenados utilizando la arquitectura ResNetUNet y DeepLabV3 no son capaces de identificar esta clase, puesto que se trata de una clase en la que a pesar de aparecer en la mayoría de las imágenes del dataset, el número de píxeles en las que aparece es ínfimo, y es por ello que el modelo no es capaz de identificar ningún píxel de esta clase para llevar a cabo su segmentación semántica. De esta manera, se afirma que esta clase en las imágenes de este dataset no es posible de ser segmentada semánticamente.

Finalmente, la última de las clases de este dataset es *moving car*. Los valores obtenidos en esta ocasión en la métrica F1, han sido de 0.5816 y 0.4889 para el caso del primer entrenamiento para las arquitecturas ResNetUNet y DeepLabV3, mientras que para este entrenamiento, dichos valores han sido de 0.6183 y 0.5586 para las mismas arquitecturas, por lo que se observa como los resultados obtenidos en este caso han aumentado. Esto mismo ocurre con la métrica IOU, en donde se ha pasado de porcentajes de 41,01 % y 32,35 %, a porcentajes del 44,75 % y 38,75 %. De esta forma, se aprecia como con este entrenamiento los resultados obtenidos para esta clase han aumentado considerablemente, si bien se trata aún de valores bajos para una buena segmentación, los valores obtenidos han aumentado con respecto a los

que ya se tenían del entrenamiento anterior.

7.6.2.2. Análisis de los resultados obtenidos globalmente en el dataset urbano

Por último, al igual que en las tablas comparativas que se han visto anteriormente, aparecen los datos globales. Por ello, si se comparan los valores del primer entrenamiento con los valores que se han obtenido en este último se observa como para los valores de precisión, recall y F1 se obtuvieron en un principio valores de 0.8359 y 0.8520, para las arquitecturas ResNetUNet y DeepLabV3 respectivamente. En cambio, en este último entrenamiento se han tenido los valores 0.8311 y 0.8632 para las arquitecturas ResNetUNet y DeepLabV3 respectivamente. En este sentido, para el caso en el que se hace uso de la arquitectura ResNetUNet los valores no consiguen mejorar, pero en cambio con la arquitectura DeepLabV3 los valores mejoran un 0.0112. Por otra parte, si se fija en los valores proporcionados por la métrica IOU, se puede observar como en esta ocasión, los resultados de esta métrica obtenida de forma global también son mayores a los obtenidos por la arquitectura DeepLabV3, puesto que se ha pasado de intersectar en el 74,22 % al 75,94 % en términos de intersección global, lo que implica un 1,72 % de mejora en cuanto a la intersección se refiere. En resumen, la comparativa de este dataset ha hecho deducir que tanto de forma por clase, como de manera general, la arquitectura que mejor proporciona valores y por tanto una mejor segmentación semántica es la arquitectura DeepLabV3.

Tal y como se ha ido realizando anteriormente, en la ilustración 7.24 se muestra la gráfica de barras de ambas arquitecturas desglosadas por clases y de forma global.

En la ilustración 7.24, se aprecia como se obtiene unos resultados muy parecidos a los obtenidos en la sección anterior en los primeros entrenamientos (sección 7.5.2.2), y es que en esta ocasión se vuelven a tener que 6 de las 8 clases obtienen mejores resultados con la arquitectura DeepLabV3. Además, una de las clases no presenta ningún valor de intersección para ninguna de las arquitecturas, de esta manera no se tienen resultados. Tan solo una de las clases, la clase *Moving car* presenta mejores resultados al testearlo con el modelo con el que se utilizó la arquitectura ResNetUNet. Al fijarse en los valores globales, es fácil deducir que la mejor de las arquitecturas en este caso es la arquitectura DeepLabV3, puesto que se trata de la que en mayor medida a proporcionado unos mejores resultados de intersección.

Al igual que se hizo anteriormente con el dataset rural, en este caso se va a mostrar algunas de las predicciones realizadas por el modelo con la arquitectura DeepLabV3 con el dataset urbano. En este sentido, en la ilustración 7.25 se puede observar como se encuentran tres imágenes, con sus correspondientes máscaras y predicciones por el modelo, se encuentra ordenadas de la misma manera que anteriormente, es decir, en el margen izquierdo se encuentran las imágenes reales, en la parte central se encuentran las máscaras de dichas imágenes, y finalmente en el margen derecho se encuentran las predicciones realizadas por el modelo.

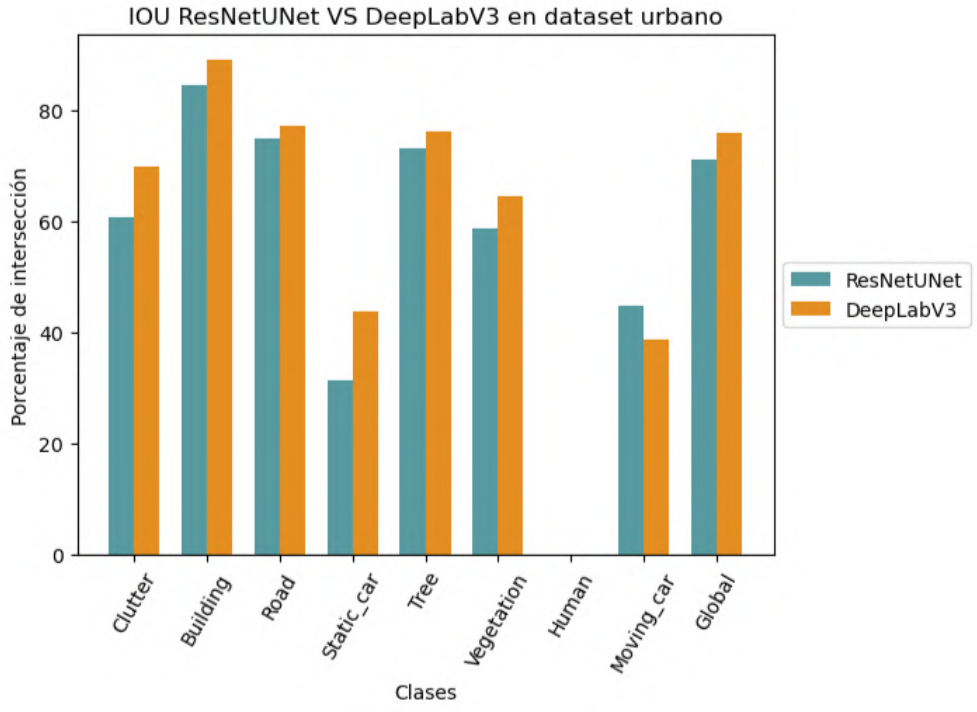

Ilustración 7.24: Comparación de los valores obtenidos para la métrica IOU por clases y globalmente sobre el conjunto de testeo del dataset urbano utilizando las arquitecturas ResNetUNet y DeepLabV3.

7.6.3. Resultados generales finales

Una vez realizadas las dos comparaciones de los resultados obtenidos de forma que los entrenamientos sean parados cuando el coste de validación no mejore en tres épocas consecutivas, así como la disminución del tamaño de las imágenes y el aumento de imágenes por cada *batch*. Se puede concluir que la arquitectura que proporciona unos mejores resultados a la vista de los resultados estudiados y analizados, es la arquitectura DeepLabV3. Como se ha visto, algunos de los valores de las métricas no han mejorado en este segundo entrenamiento, e incluso han empeorado en cierta medida en algunas de las clases, esto se debe a que aunque el entrenamiento ahora se para cuando se detecta que se está produciendo un sobreentrenamiento, el tamaño de la imágenes se ha reducido, y esto hace que la calidad de las mismas empeore en cierta medida.

En este caso, es factible el empeoramiento de algunas de las clases, puesto que implica una serie de ventajas, y es que como bien se ha indicado el tamaño de las imágenes se ha reducido de un tamaño de 1376x736 a un tamaño de 480x256. Esto ha implicado que aunque la calidad de las imágenes disminuya y que por ejemplo clases como *human* del dataset urbano no sea detectable, ha hecho posible que se puedan incluir más imágenes en cada *batch* de entrenamiento. Esto es sumamente importante para garantizar que el gradiente calculado por cada *batch* fluya de una manera más armónica y con menos ruido. Por otra parte, esto último explicado unido con una mayor cantidad de épocas de entrenamiento, ha hecho que

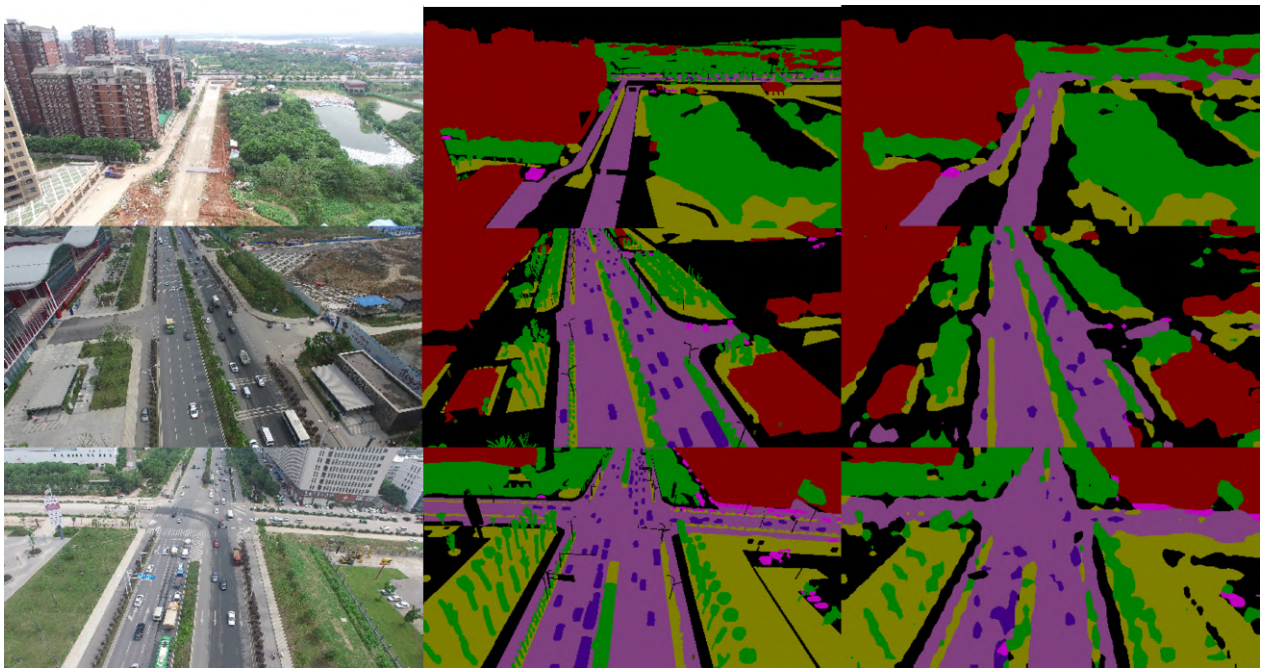

Ilustración 7.25: Algunas de las predicciones realizadas por el modelo con la arquitectura DeepLabV3 con el conjunto de testeo del dataset urbano. En el margen izquierdo se encuentra las imágenes reales, en la parte central las correspondientes máscaras y en el margen derecho se encuentran las predicciones.

la mayoría de las clases definidas en el dataset urbano hayan mejorado considerablemente al utilizar la arquitectura DeepLabV3. Por todo esto, se concluye afirmando que para el tipo de segmentación semántica que se está llevando a cabo de imágenes aéreas, la mejor arquitectura que proporciona unos mejores valores a las métricas es la arquitectura DeepLabV3.

7.7. *Transfer learning*

Una vez realizado los entrenamientos con las arquitecturas anteriores y obtenido como resultado que la arquitectura DeepLabV3 es la que mejores resultados de métricas proporciona para la segmentación semántica de imágenes, a continuación se va a realizar un *transfer learning* sobre los entrenamientos.

El *transfer learning* se basa en dado el conocimiento que tienen los modelos con los entrenamientos anteriores, usar dichos modelos entrenados como base de otro entrenamiento con intención que los resultados sean mejores.

Por ello, se ha tenido que reidentificar las distintas clases de los datasets, y es que como se desea partir del entrenamiento de uno de los modelos, es necesario que las distintas clases de ambos datasets correspondan entre sí, para poner un ejemplo, la clase *residential* que aparece en el cuadro 7.3 con un valor representativo de 4 para el dataset rural, en el caso del dataset urbano, la clase equivalente es *building* y ésta posee como valor representativo el 2 tal como

se aprecia en el cuadro 7.4. Por ello, en el cuadro 7.5 se observa las nuevas correspondencias de las clases entre los datasets para que concuerden entre sí.

Cuadro 7.5: Correspondencia de valores de píxeles entre ambos datasets para el *transfer learning*.

Color	Clase dataset rural	Clase dataset urbano	Valor de clase representativo
	Fence	-	1
	Land	Low vegetation	2
	Forest	Tree	3
	Residential	Building	4
	Haystack	-	5
	Road	Road	6
	Church	-	7
	Car	Static car, Moving car	8
	Water	-	9
	Sky	-	10
	Hill	-	11
	Person	Human	12
	-	Clutter	13

Una vez realizada la correspondencia anterior, se han tenido en cuenta dos posibilidades diferentes para realizar *transfer learning*, el primero de ellos consiste en partiendo del modelo entrenado con el dataset rural, entrenar nuevamente el modelo solo con las imágenes del dataset urbano, de esta manera, lo que se pretende es que dado que el conocimiento adquirido por el aprendizaje del dataset rural posee características de algunas de las clases presentes en el dataset urbano, el entrenamiento, y por tanto el aprendizaje del modelo para el dataset urbano debería de mejorar con respecto al anterior.

Por otra parte, otra de las posibilidades es que partiendo del mismo modelo entrenado, es decir, del modelo rural, ya que posee mejores valores, se entrene nuevamente el modelo con las imágenes tanto del dataset rural como del urbano, en esta ocasión lo que se desea es mejorar los valores de las métricas obtenidas para ambos datasets, y por tanto el modelo resultante será capaz de identificar las clases que se encuentran en ambos datasets.

A continuación se observarán en las siguientes secciones como se llevarán a cabo las dos posibilidades nombradas.

7.7.1. *Transfer learning* con imágenes del dataset urbano

En esta sección se realizará el *transfer learning* de manera que como resultado tan solo se desea un modelo que sea capaz de segmentar imágenes aéreas urbanas. De esta forma, lo que se desea es mejorar la segmentación semántica de este dataset. En este sentido, existen dos posibles *transfer learning*, el primero de ellos es partiendo del modelo entrenado con el dataset rural y todas sus clases, es decir, el modelo ya entrenado y analizado anteriormente. Por otra

parte, otra de las posibilidades es en vez de partir del modelo entrenado con el dataset rural con todas sus clases, partir del modelo entrenado con el dataset rural pero poniendo aquellas clases que no se encuentran en el dataset urbano como la clase *clutter*, la cual si pertenece al dataset urbano. A continuación, se muestra los resultados obtenidos llevadas a cabo ambas posibilidades.

7.7.1.1. Con entrenamiento de todas las clases del dataset rural

Tal y como se ha indicado anteriormente, en esta ocasión lo que se ha realizado es a partir del modelo entrenado para el dataset rural, se ha vuelto a entrenar el modelo pero en esta ocasión con las imágenes del dataset urbano. Una vez terminado el entrenamiento, se han obtenido las métricas sobre el conjunto de testeo, y los valores obtenidos se muestran en la ilustración 7.26. Cómo se aprecia, en la ilustración 7.26, no se observan los mismos nombres que en la ilustración 7.22, que se trata del entrenamiento del dataset urbano anteriormente analizado sin *transfer learning* y esto se debe a que se ha realizado la correspondencias de las distintas clases de ambos datasets, recordando que en el cuadro 7.5 se encuentra la susodicha correspondencia, además, se puede observar como ahora se han unificado las clases *static car* y *moving car* en una sola clase llamada *car*, y esto se debe a que con el dataset el cual se ha partido como entrenamiento, es decir, el dataset rural, no existía una diferenciación de las clases *car* en función de si estaban moviéndose o estaban estáticos, por ello, se han unificado estas dos clases.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Forest	0.8849	0.8685	0.9319	0.8767	0.7804
Residential	0.9369	0.9482	0.9630	0.9426	0.8913
Land	0.7735	0.8030	0.9522	0.7880	0.6501
Road	0.8446	0.8822	0.9690	0.8630	0.7590
Person	0.0000	0.0000	0.9991	0.0000	0.0000
Car	0.6008	0.5132	0.9859	0.5536	0.3827
Clutter	0.7980	0.7759	0.9317	0.7868	0.6486
Global	0.8664	0.8664	0.9618	0.8664	0.7643

Ilustración 7.26: Valores de las métricas obtenidas para el *transfer learning* realizado para el dataset urbano, entrenando con todas las clases del dataset rural.

Ahora se van a analizar los valores de las métricas obtenidas en este caso, con los valores que se obtuvieron tan solo entrenando el dataset urbano, es decir, los valores que se encuentran en la ilustración 7.22. Como se observa, la clase *forest* que corresponde con la clase *tree*, con este *transfer learning* que se ha realizado ha aumentado considerablemente precisión, pero ha disminuido levemente el recall. Esto hace que el valor obtenido como F1 aumente, puesto que en esta ocasión se posee un valor de 0.8767 frente al 0.8642 obtenido anteriormente. Además, se ha logrado una mejor intersección de esta clase, ya que el valor de la métrica IOU ha aumentado de un 76.09 % a intersectar un 78.04 % de los píxeles. En resumen, los resultados

muestran una mejora en las métricas para esta clase en comparación con los resultados anteriores.

Algo similar ocurre con la siguiente de las clases mostradas en la ilustración 7.26, y es que la clase *residential*, categorizada anteriormente con el nombre *building*, ha experimentado una mejora en la precisión, pero no así con el recall, el cual ha disminuido. A pesar de esta disminución en el recall, la métrica F1 ha experimentado un ligero aumento, alcanzando un valor de 0.9426 en comparación con el valor anterior de 0.9417. En cuanto a la intersección, también ha aumentado pasando de un 88,98 % a un 89,13 %. Como se observa, como ha sido beneficiada esta clase por el *transfer learning* realizado.

La siguiente de las clases es *land*, que se corresponde con la clase *vegetation*, al contrario de las clases anteriores, esta clase ha obtenido menos precisión pero se ha visto favorecida por el aumento del recall. El importante aumento del valor del recall, se traduce en un aumento del valor obtenido en la métrica F1, y es que si antes se poseía un valor de 0.7849, ahora se posee un valor de 0.7880, si bien el aumento no ha sido significativamente mayor, ha aumentado. Este incremento también se ve reflejado en el valor de la métrica IOU, y es que sin el *transfer learning* se obtuvo un valor de 0.6459, ahora dicho valor ha aumentado llegando hasta el 0.6501. Suponiendo así una mejora al realizar este tipo de técnica.

La clase *road* se trata de una de las clases que no han mejorado los valores de las métricas obtenidas, a excepción del valor obtenido en el recall, y es que este se trata del único valor que ha aumentado pasando de un 0.8812 a un 0.8822, tratándose de un aumento casi inapreciable. El poco aumento del valor del recall y la disminución del valor de la precisión, hace que el valor obtenido como F1 haya también sido menor en este entrenamiento que en el anterior, y es que en este caso se ha pasado de un valor de 0.8715 a un 0.8630. Un 1,33 % de intersección menos en los píxeles es lo que ha supuesto la realización de este *transfer learning*. De esta manera, se ha observado como al realizar el *trasfer learning* para esta clase, ha hecho que los valores no mejoren, sino que empeoren.

La siguiente de las clases es la clase *person*, esta clase, al igual que sucedía anteriormente con la misma clase y para este mismo dataset, no puede ser detectada por la arquitectura que se ha estudiado y entrenado, por ello, los valores que se han obtenido han sido de ceros. Además, se puede observar, como el valor obtenido como accuracy se trata de un valor muy alto, y esto es debido a lo explicado en secciones anteriores. Por tanto, la clase *person* es incapaz de ser detectada por la arquitectura, de esta manera no se obtienen ni mejores ni peores resultados.

La clase *car* proviene de la unión de dos clases, *static car* y *moving car*, en este caso, se observa una disminución en los valores obtenidos en comparación con los valores anteriores, y es que si antes se poseían valores de precisión de 0.6673 y 0.6440, para *static car* y *moving car* respectivamente, en este caso el valor obtenido para la clase *car* ha sido de 0.6008, esto representa una disminución significativa en la precisión obtenida. Por otra parte, en cuanto a los valores de recall obtenidos anteriormente habían sido de 0.5577 y 0.4932 para *static car* y *moving car*, mientras que el valor obtenido ahora ha sido de 0.5132, este valor es mayor que el obtenido para la clase *moving car* pero teniendo en cuenta que se está realizando un *transfer learning* y además se están unificando las dos clases en una sola, se trata de unos

valores muy bajos. El valor obtenido para la métrica F1 no ha mejorado y lo mismo ocurre con la métrica IOU, en donde anteriormente intersectaban un 43,64 % y un 38,75 % para las clases *static car* y *moving car* ahora este valor disminuye hasta intersectar en tan solo un 38,27 %. Esto indudablemente representa un deterioro en los datos de esta clase, que si ya se trataba de una clase con unos valores muy bajos que no poseían una segmentación semántica muy buena, con los datos obtenidos, esta segmentación ha empeorado considerablemente.

La última de las clases es la clase *clutter*, esta clase también ha obtenido valores más bajos con respecto a los obtenidos antes del *transfer learning*, y es que los valores de precisión y de recall han disminuido. Lo que ha llevado que el valor de F1 disminuya en un 0.0365 con respecto a lo que se obtuvo sin el *transfer learning*. Si se observa la intersección de esta clase el valor obtenido baja de un 69,83 % a un 64,86 %, lo que supone una bajada muy considerable de los valores. De esta forma, se puede concluir que para esta clase, el *transfer learning* no ha mejorado los datos.

Al igual que se ha realizado anteriormente, se ha incluido una fila en la que se recogen los valores de las métricas de manera global considerando todas las clases anteriores descritas, teniendo en cuenta que al considerar todas las clases, los valores de precisión, recall y F1 serán los mismos. Por ello, como valores de precisión, recall y F1, se han obtenido un valor de 0.8664 frente al 0.8632 que se había obtenido para las mismas métricas sin *transfer learning* lo que supone un aumento de un 0.0032. A su vez, en cuanto a la intersección el valor obtenido ha aumentado pasando de un 0.7594 a un 0.7643, esto indica que se ha logrado una mejor intersección global entre las clases del dataset. En resumen, aunque algunas clases han sido afectadas negativamente por el *transfer learning*, obteniendo valores inferiores a los anteriores, hay clases en las que se ha mejorado considerablemente los resultados. Por lo tanto, en general, el *transfer learning* proporciona una mejor segmentación en comparación con la no utilización del mismo.

7.7.1.2. Con entrenamiento de las clases pertenecientes al dataset urbano

Una vez analizados los valores de las métricas obtenidas anteriormente, se ha observado que algunas clases no han mostrado mejoras en comparación con los valores obtenidos antes del *transfer learning*. Una de las clases más afectada ha sido la clase *clutter*, y es que llegados a este punto hay que preguntarse ¿por qué no mejoran los valores obtenidos en la clase clutter?, y la respuesta a esta pregunta no es fácil, una de las posibles razones es que como bien se ha indicado anteriormente, se está aplicando *transfer learning* utilizando el entrenamiento del dataset rural, y este dataset, no posee ninguna clase similar con la clase *clutter*, ya que se recuerda que las clases que se encuentra en el dataset rural son las que se muestran en el cuadro 7.1. Por ello, una de las hipótesis es que hay clases que se encuentran en el dataset rural que no están como clases separadas en el dataset urbano, como pueden ser las clases *sky*, *water*, *church*, entre otras, por tanto, estas clases en la segmentación que tiene que hacer el modelo con el entrenamiento del dataset urbano es segmentarlo como *clutter*. Esto en el entrenamiento hace que el modelo le cueste más aprender que ya por ejemplo la clase *sky* es ahora la clase *clutter* para el dataset urbano. Por ello, lo que se va a realizar ahora es no partir del entrenamiento del dataset rural tal y como se había mostrado en la ilustración

7.22, sino que ahora se va a entrenar nuevamente el modelo pero modificando las clases de entrada, ya que ahora van a tener las clases que solo aparece en el dataset urbano, en el cuadro 7.6 se muestra como se encuentran ahora la distribución de las clases para este nuevo entrenamiento.

Cuadro 7.6: Correspondencias de las clases del dataset rural y urbano para el *transfer learning*.

Color	Clases dataset rural	Clases dataset urbano
	Forest	Tree
	Residential, Church	Building
	Land	Low vegetation
	Road	Road
	Car	Static car, Moving car
	Person	Human
	Sky, Hill, Fence, Water, Haystack	Background clutter

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Forest	0.9633	0.9312	0.9828	0.9470	0.8993
Residential	0.9336	0.9466	0.9580	0.9400	0.8869
Land	0.9285	0.9415	0.9682	0.9350	0.8779
Road	0.9272	0.9300	0.9697	0.9286	0.8667
Person	0.8836	0.8233	0.9923	0.8524	0.7428
Car	0.8198	0.6616	0.9990	0.7323	0.5776
Clutter	0.8357	0.5163	0.9983	0.6383	0.4687
Global	0.9341	0.9341	0.9812	0.9341	0.8764

Ilustración 7.27: Testeo del modelo entrenado con el dataset rural poniendo las clases que no se encuentran en el dataset urbano como *clutter*.

Ahora se va a llevar a cabo un breve análisis de lo que ha supuesto este nuevo entrenamiento del dataset rural teniendo en cuenta que las clases que no se encuentran en el dataset urbano ahora han sido categorizadas como la clase *clutter*. En este sentido, si se comparan los valores de la tabla de la ilustración 7.27 con los que se había obtenido anteriormente y que se encuentran en la tabla de la ilustración 7.17, se puede observar como los cambios van fluctuando levemente con respecto a los valores que ya se tenían y en alguno de los casos se puede apreciar como se posee una mejor precisión y recall. Por ello, si se observa la última de las filas de la tabla, se encuentra el global teniendo en cuenta todas las clases, y si se compara los valores que se han obtenido ahora con los valores globales que se habían obtenido anteriormente teniendo en cuenta el resto de las clases se observa como de esta manera se tienen unos mejores resultados, puesto que para los valores de precisión, recall y F1 se habían obtenido un valor de 0.9190, mientras que en este caso se ha obtenido un valor de 0.9341. Este valor supone un aumento considerable, recuérdese que se tratan de valores que se han obtenido sobre el conjunto de testeo y por ello, el modelo no ha sido ni entrenado ni validado con las

mismas imágenes, por lo que tener una precisión de más de un 93 % es una cifra bastante buena. Además, si se observa el valor de la intersección de todos los píxeles de las imágenes con su correspondiente máscara, se observa como este valor ha aumentado pasando de un 85,02 % a un 87,64 %, se trata por tanto de un valor que ha aumentado considerablemente y que de esta manera está indicando la buena segmentación semántica que está realizando el modelo sobre las imágenes de testeo.

Llegados a este punto, lo que se va a realizar es, partiendo del modelo que se acaba de analizar, es decir, el entrenamiento del dataset rural con la modificación de las clases para que aquellas que no se encuentren en el dataset urbano pertenezcan ahora a la clase *clutter* la cual si lo posee el dataset urbano, pues con este modelo ya entrenado se va a volver a entrenar pero tan solo con las imágenes del dataset urbano. De esta manera lo que se está realizando es el *transfer learning* sobre un entrenamiento el cual posee como salidas los mismos valores de clases de los ya entrenados. Una vez entrenado este modelo al que se ha realizado *transfer learning*, se han obtenido los valores para las distintas métricas para el conjunto de testeo, y en esta ocasión se han obtenido los resultados que se muestran en la ilustración 7.28

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Forest	0.8403	0.8553	0.9228	0.8477	0.7357
Residential	0.9198	0.9459	0.9610	0.9327	0.8738
Land	0.8173	0.7846	0.9415	0.8006	0.6675
Road	0.8320	0.8895	0.9656	0.8598	0.7541
Person	0.0000	0.0000	0.9988	0.0000	0.0000
Car	0.6603	0.4785	0.9855	0.5549	0.3840
Clutter	0.7940	0.7556	0.9231	0.7744	0.6318
Global	0.8491	0.8491	0.9569	0.8491	0.7378

Ilustración 7.28: Resultados de las métricas sobre el conjunto de testeo al realizar *transfer learning* con el dataset urbano, entrenando con las clases pertenecientes al dataset urbano.

Con los resultados presentados en la ilustración 7.28, se van a comparar con los obtenidos en el *transfer learning* anterior, cuyos resultados se muestran en la ilustración 7.26. Al contrario de lo que se pensaba que podría ocurrir, los valores obtenidos en esta ocasión son peores de los que se habían tenido anteriormente con el otro *transfer learning*. Y es que son muy pocos los valores que mejoran, véase que las clases *forest*, *residential*, y *clutter* son las clases que en ninguno de los valores obtenidos en las métricas mejoran con respecto al *transfer learning* anterior, y teniendo en cuenta que son las clases que más apariciones y mayor cantidad de píxeles tienen en la totalidad del dataset, hace que este *transfer learning* tenga claramente peores resultados. Aunque hay que destacar que por ejemplo la clase *land* mejora puesto que anteriormente se obtuvo valores de precisión y de recall de 0.7735 y 0.8030 y ahora estos valores son de 0.8173 y de 0.7846, si bien, el recall no aumenta con respecto al valor obtenido anteriormente, si es suficiente el aumento de la precisión para obtener un mejor valor de F1, el cual pasa de 0.7880 a un 0.8006. Lo mismo ocurre con la métrica IOU, que se pasa de intersectar el 65,01 % de los píxeles de esta clase a intersectar el 66,75 %.

Esto no ocurre con la clase *road*, en el que la precisión ha bajado de un 0.8446 a un 0.8320, esta bajada asociada a un crecimiento mínimo del valor proporcionado por la métrica recall, cuyo valor tan solo ha aumentado un 0.0073 hace que tanto el F1 como el IOU no mejoren con los obtenidos anteriormente.

La clase *person* como bien se ha indicado anteriormente, es incapaz de ser identificada en este dataset urbano, ni habiendo realizado un *transfer learning*, es por ello que los valores obtenidos son exactamente iguales a los ya analizados con anterioridad.

Por otra parte, el hecho de haber comenzado el entrenamiento utilizando las clases del dataset rural con modificaciones, donde las clases ausentes en el dataset urbano se catalogaron como *clutter*, no ha resultado en una mejora en los valores de las métricas para la clase *clutter*. Al contrario a lo esperado inicialmente, los valores obtenidos en este entrenamiento han sido más bajos en comparación con los anteriores.

Por todo ello, como se observa de manera global, los resultados obtenidos han sido más inferiores a los obtenidos anteriormente, y es que se han obtenidos valores de precisión, recall y F1 de 0.8491 mientras que este valor con el *transfer learning* anterior era de 0.8664, lo que supone una bajada considerable. Esto va asociado además a una disminución de la intersección de manera global que en donde antes superaba el 76 %, ahora no llega ni al 74 %.

De hecho, la comparación que se acaba de realizar es con respecto al otro *transfer learning* que se llevó a cabo, puesto que se trata de encontrar el modelo que mejor resultados obtenga, pero si comparamos los valores obtenidos en este *transfer learning* con los resultados que se obtuvieron en el entrenamiento de este mismo dataset sin aplicarle *transfer learning*, es decir, con los valores que se observan en la ilustración 7.22, se observa como con el *transfer learning* existe un empeoramiento de los resultados obtenidos, ya que anteriormente se había obtenido un valor de precisión, recall y F1 de 0.8632, mientras que este valor ahora es de 0.8491. Aunque no es una disminución significativa en las métricas, es lo suficientemente notable como para afirmar que en esta ocasión el *transfer learning* no ha mejorado los resultados obtenidos con anterioridad.

Llegados a este punto, si lo que se desea es un modelo de segmentación semántica de imágenes aéreas urbanas, el mejor modelo que se podría escoger sería el modelo realizado con *transfer learning*, realizando dicho *transfer learning* a partir del entrenamiento del dataset rural con todas sus clases.

7.7.2. *Transfer learning* con imágenes del dataset rural y urbano

En esta sección se llevará a cabo el *transfer learning* con el objetivo de obtener un modelo que sea capaz de identificar las clases que se encuentran presentes tanto en el dataset rural como en el urbano. Para lograr lo dicho, al igual que se hizo anteriormente, se han realizado dos *transfer learning* distintos.

7.7.2.1. Con entrenamiento de todas las clases del dataset rural

En este caso, el *transfer learning* que se va a realizar consiste en, a partir del modelo del dataset rural del cual se obtuvieron sobre el conjunto de testeo las métricas que aparecen en la ilustración 7.17 entrenar nuevamente a partir de dicho modelo con imágenes tanto del dataset rural como del dataset urbano. Una vez entrenado este nuevo modelo que será capaz de identificar las clases que se encuentran en ambos modelos, se han obtenido las métricas sobre el conjunto de entrenamiento y los resultados se muestran en la ilustración 7.29

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Hill	0.9466	0.9476	0.9963	0.9471	0.8995
Forest	0.9208	0.9252	0.9512	0.9230	0.8570
Residential	0.9434	0.9380	0.9694	0.9407	0.8880
Land	0.8853	0.8867	0.9613	0.8860	0.7953
Church	0.9548	0.9366	0.9981	0.9456	0.8669
Road	0.8461	0.8844	0.9827	0.8649	0.7619
Fence	0.8423	0.7348	0.9946	0.7849	0.6459
Person	0.8496	0.5444	0.9989	0.6636	0.4966
Car	0.7071	0.5201	0.9951	0.5993	0.4279
Sky	0.9806	0.9849	0.9981	0.9828	0.9661
Haystack	1.0000	0.0064	0.9995	0.0127	0.0064
Water	0.9598	0.8896	0.9990	0.9234	0.8577
Clutter	0.7237	0.7525	0.9697	0.7378	0.5846
Global	0.9069	0.9069	0.9857	0.9069	0.8297

Ilustración 7.29: Testeo con los conjuntos de los datasets rural y urbano al realizarle un *transfer learning* partiendo del entrenamiento del dataset rural con todas sus clases.

Como se observa en la ilustración 7.29, aparecen las métricas de las distintas clases sobre el conjunto de testeo. Además, en este caso la tabla es mayor debido a que se encuentran las 13 clases distintas que se encuentran en ambos datasets, fijarse que por ejemplo las clases *static car* y *moving car* del dataset urbano se han unificado como bien se indicó anteriormente en la clase *car*, puesto que para el dataset rural existe la clase *car* pero no distingue si está en movimiento o no el mismo. El resto de las clases se encuentran recogidas entre los datasets, como puede ser la clase *tree* que ahora la recoge la clase *forest*, o *building*, que ahora es *residential*. Por tanto, todas las máscaras se han modificado para que la correspondencia de imágenes sean correcta entre los datasets.

- **Testeo dataset rural**

Aunque, como se observa en la ilustración 7.29, los datos parecen que son prometedores a primera vista, puesto que se ven valores relativamente altos, sin embargo, como esta ilustración se trata de los valores de testeo sobre las imágenes tanto del dataset rural como del dataset urbano, no muestra con claridad y por separado las clases según los datasets. Por ello, lo que se ha hecho ha sido separar las imágenes de testeo de ambos datasets y se han calculado las

métricas según las imágenes de testeo de cada dataset. En este sentido, las métricas que se han obtenido para el dataset rural son las que se muestran en la ilustración 7.30.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Hill	0.9454	0.9477	0.9937	0.9466	0.8985
Forest	0.9541	0.9446	0.9606	0.9493	0.9036
Residential	0.9190	0.9398	0.9723	0.9293	0.8679
Land	0.9266	0.9376	0.9722	0.9321	0.8728
Church	0.9483	0.9058	0.9968	0.9266	0.8632
Road	0.8763	0.8985	0.9931	0.8872	0.7973
Fence	0.8662	0.7503	0.9945	0.8041	0.6723
Person	0.8322	0.6613	0.9996	0.7370	0.5836
Car	0.8028	0.7716	0.9991	0.7869	0.6487
Sky	0.9784	0.9863	0.9974	0.9823	0.9652
Haystack	1.0000	0.0104	0.9991	0.0205	0.0104
Water	0.9762	0.8983	0.9990	0.9356	0.8790
Global	0.9384	0.9384	0.9905	0.9384	0.8839

Ilustración 7.30: Testeo del conjunto del dataset rural sobre el *transfer learning* realizado con ambos datasets y partiendo del entrenamiento con todas las clases del dataset rural.

Con los resultados mostrados en la ilustración 7.30 se pueden comparar con los datos obtenidos con el testeo del entrenamiento del dataset rural antes de hacer el *transfer learning* para determinar si este *transfer learning* realizado mejora los resultados obtenidos en el dataset rural, por lo tanto, se compararán con los resultados observados en la ilustración 7.17.

Comenzando con el análisis, se empezará con la clase *hill*, esta clase ha mejorado considerablemente la precisión y el recall. El aumento de estos valores se ve reflejada en el valor de la métrica F1, en donde se ha pasado de un valor de 0.9164 a 0.9466, es decir, ha aumentado más de un 0.0322. Lo mismo ocurre con la intersección de píxeles de esta clase, se ha pasado de intersectar el 84,58 % al 89,85 %, es decir, está intersectando un 5 % más de píxeles de la clase *hill*. Por lo que, a la vista de los resultados obtenidos, se puede afirmar que con este *transfer learning*, se ha mejorado la segmentación semántica de esta clase.

La siguiente de las clase es *forest*, esta clase también ha experimentado un aumento de los valores de las métricas, aunque lo ha hecho en menor medida que la clase anterior, pero el aumento ha sido suficiente para hacer que aumente el valor de F1 en un 0.016 llegando hasta un 0.9493. Con respecto al IOU, éste ha aumentado un 3 %, siendo ahora capaz de intersectar el 90,36 % de los píxeles de la clase *forest*. En este sentido, esta clase también ha mejorado la segmentación semántica al hacer uso de esta técnica.

La clase *residential* también ha mejorado sustancialmente la segmentación semántica de esta clase si a los datos no referimos, y es que la precisión ha aumentado considerablemente, el recall también lo ha hecho pero en menor medida. Pero como se ha visto hasta ahora, un aumento de la precisión y del recall hace que aumente el valor de F1, y así es, en esta ocasión

se ha pasado de un 0.9104 a un 0.9293. Mientras que en intersección, ésta ha aumentado en más un 3 %, puesto que se ha pasado de un 83,55 % a un 86,79 %.

Si bien, todas las clases analizadas hasta ahora han mejorado los valores de las métricas, la clase *land* no iban a ser una excepción, ya que han aumentado los valores de precisión y de recall. En cuanto a los valores de F1 e IOU, estos anteriormente eran de 0.9056 y 0.8275, mientras que ahora han aumentado hasta los 0.9321 y 0.8728 respectivamente. Cómo se observa, existe un importante aumento de los valores, sobre todo el de la intersección, el cual ha supuesto un incremento del 5 %.

En el caso de la clase *church*, la precisión ha aumentado, pero no lo ha hecho así el valor del recall, y es que éste último antes era de 0.9170 y ahora es de 0.9058. Sin embargo, dado que el incremento en la precisión ha sido mayor que la disminución en el recall, el valor de F1 ha aumentado obteniendo un valor de 0.9266 frente al 0.9202 que se tuvo anteriormente. En este caso, el incremento de IOU ha supuesto poco más del 1 %, llegando en este caso al 86,32 %.

Seguidamente, la clase *road* ha sido beneficiada considerablemente por este *transfer learning*, y es que han mejorado todos los valores de las métricas, puesto que en la precisión ha pasado de un 0.8694 a un 0.8763 y el recall de un 0.8395 a un 0.8985. En cuanto al valor de la métrica F1, ha aumentado de un 0.8542 a un 0.8872. Este incremento ha hecho que la clase *road* pase de intersectar el 74,55 % de los píxeles a un 79,73 % de los píxeles, haciendo así que esta clase haya pasado de una clase que se segmenta regular, a una clase cuya segmentación la realiza algo mejor.

Aunque las siguientes tres clases que se mencionan a continuación no pueden afirmarse que tengan una muy buena segmentación puesto que sus valores son algo bajos, sus valores han aumentado en comparación a los valores que se poseían. Estas clases son *fence*, *person* y *car*, las cuales han obtenido unas intersecciones de 67,23 %, 58,38 % y 64,87 % respectivamente. A pesar de ser valores bajos, han sido valores más altos a los obtenidos anteriormente.

La clase *sky* se trata de la clase que mejores valores de métricas poseía, y es por ese mismo motivo por lo que el incremento de estos valores se ha visto menos influenciado. La poca disminución de la precisión y el poco aumento del recall hace que el valor de F1 se pueda considerar prácticamente igual al que se obtuvo anteriormente y es que antes el valor era de 0.9822 y ahora es de 0.9823. Mientras que en el caso de la métrica IOU, también se posee una mínima variación, puesto que antes se ha tenido un 0.9649 y ahora se ha obtenido un 0.9652 de intersección de esta clase. Aunque los cambios han sido mínimos, se logran obtener mejores valores con el uso del *transfer learning*.

En cuanto a la clase *haystack* los datos obtenidos son bastante curiosos, y es que en este caso se ha obtenido una precisión de 1.0000, lo cual resulta sorprendente teniendo en cuenta que se trata de un clase con una escasa aparición, y que no aparece en todas las imágenes del dataset, da a pensar que este valor se trata un valor ficticio, y que realmente puede ser debido a que en el testeo de las imágenes tan solo se muestre una imagen con esta clase y que sea capaz de detectarla en esa imagen, además anteriormente, la precisión de la misma clase era de 0.9193. Otra forma de detectar que esta clase no se está segmentando de forma correcta es observando el valor del recall, y es que en este caso, dicho valor es de 0.0104 un valor que se puede considerar prácticamente nulo. Aunque anteriormente el valor era de 0.0873, que se

trata sin duda de un valor mayor que el obtenido en esta ocasión, se trata igualmente de un valor muy pequeño. Esto se refleja en la métrica F1, en donde el valor antes era de 0.1595 y ahora es de 0.0205, se tratan de dos valores muy bajos siendo aún peor el obtenido en este entrenamiento con *transfer learning*. A la vista de estos valores, no es necesario ni siquiera analizar la métrica IOU debido a los valores tan bajos que se tiene.

En último lugar se encuentra la clase *water*, esta clase se ha visto beneficiada por este *transfer learning*. Como la disminución del recall ha sido poca pero el aumento de la precisión ha sido mayor, el valor que se ha obtenido en la métrica F1 ha sido mayor en este caso, y es que se ha obtenido un 0.9356 frente al 0.9314 que se había obtenido antes del *transfer learning*. En cuanto a la intersección de esta clase, se observa como el aumento ha sido pequeño, pasando de intersectar el 87,17 % de los píxeles al 87,90 % de los píxeles, se trata por tanto de una disminución bastante pequeña, pero que supone un aumento por mínimo que sea para el *transfer learning*.

Si bien, como se ha ido observando, se obtienen unos mejores resultados en el dataset rural en este caso al aplicarle el *transfer learning*. Pero para tener una visión general de lo que ha supuesto la mejora en las distintas clases, se observará la última fila de la ilustración que se ha analizado, y es que se tiene el global correspondiente a todas las clases hasta ahora analizadas. En esta ocasión se posee un valor de precisión, recall y F1 de 0.9384 frente al 0.9190 que se tenía anteriormente. Asimismo, la intersección de las clases en global ha pasado de intersectar en un 85,02 % a un 88,39 %. Este incremento de más del 4 % indica una mejora en la segmentación en comparación con el modelo que no utilizó *transfer learning*.

- **Testeo dataset urbano**

Una vez analizados los resultados obtenidos para el conjunto de testeo del dataset rural, se va a proceder a analizar los resultados pero sobre el conjunto de testeo del dataset urbano, ya que se recuerda que en este caso, al haber entrenado con los dos datasets, el modelo ha de ser capaz de identificar las clases que se encuentran en ambos datasets, por ello, se ha llevado a cabo la obtención de los resultados. Obteniendo para el conjunto de testeo del dataset urbano las métricas que se observan en la ilustración 7.31.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Forest	0.8472	0.8983	0.9340	0.8720	0.7730
Residential	0.9215	0.9583	0.9648	0.9395	0.8860
Land	0.7839	0.8650	0.9461	0.8225	0.6985
Road	0.8510	0.8740	0.9650	0.8624	0.7580
Person	0.0000	0.0000	0.9988	0.0000	0.0000
Car	0.6835	0.4777	0.9850	0.5624	0.3912
Clutter	0.8755	0.6814	0.9280	0.7664	0.6212
Global	0.8612	0.8604	0.9602	0.8608	0.7556

Ilustración 7.31: Testeo del conjunto del dataset urbano sobre el *transfer learning* realizado con ambos datasets y partiendo del entrenamiento con todas las clases del dataset rural.

A la vista de los resultados obtenidos como se muestra en la ilustración 7.31, se va a analizar con respecto a los valores obtenidos con el entrenamiento de este mismo dataset pero sin haberle realizado *transfer learning*, es decir, se van a comparar los resultados con los obtenidos en la ilustración 7.22.

Por ello, primero se comenzará analizando los valores de las métricas obtenidos para la clase *forest*, esta clase, si se recuerda, corresponde con la clase *tree*, y como se observa se ha obtenido un valor más bajo de precisión. Aunque con respecto al recall, en esta ocasión el valor ha aumentado ligeramente este aumento implica que el valor de la métrica F1 sea mayor en este caso, debido a que se ha pasado de tener un 0.8642 a un 0.8720. Con estos resultados es fácil deducir que la intersección de esta clase será mayor, y así, es, ya que antes se intersectaban el 76,09 % de los píxeles y ahora este porcentaje ha aumentado llegando hasta el 77,30 %. Obteniendo de esta manera, mejores resultados en esta ocasión.

En relación al aumento mostrado en la clase anterior, cabe destacar que no se ha producido lo mismo en la clase *residential* y es que su precisión ha disminuido aunque hay que resaltar que el recall ha aumentado levemente. Aunque este incremento en el recall es positivo, no ha sido suficiente para compensar la disminución en la precisión, en el que se ha pasado de 0.9417 a 0.9395. Ocurre lo mismo con el valor obtenido para la intersección, y es que este valor antes era de 88,98 %, mientras que ahora ha disminuido hasta el 88,60 %. Aunque esta disminución es mínima y se podría considerar que los valores son casi iguales, teniendo en cuenta el *transfer learning* que estamos realizando es suficiente para afirmar que en este caso no ha mejorado los valores que ya se habían obtenido.

La siguiente clase es *land*, en este caso, al compararlas con la clase *vegetation*, que es su clase equivalente, se observa que tanto su precisión como su recall han aumentado. Esto va a asociado a su vez a un aumento de la métrica F1, y es que anteriormente se obtuvo un valor de 0.7849, ahora este valor ha aumentado hasta un 0.8225. Esto se traduce en que la intersección para esta clase ha aumentado casi un 5 % siendo ahora capaz de intersectar casi el 70 % de los píxeles. Esto supone un aumento considerable de los valores obtenidos.

En el caso de la clase *road*, no se han obtenido mejores resultados con este *transfer learning*, ya que se han obtenido unos valores más bajos de precisión y de recall. Estos valores más bajos a los obtenidos han llevado a que la métrica F1 haya disminuido pasando de un 0.8715 a un 0.8624. Lo mismo ocurre con la métrica IOU en donde antes superaba el 77 % y ahora apenas llega al 76 %. De esta manera, se observa que esta clase no ha experimentado mejoras significativas por el *transfer learning* realizado.

Con respecto a la clase *person*, es importante destacar que no se han obtenido resultados significativos en comparación con su equivalente *human*. Ambas métricas obtienen valores nulos, lo cual representa una de las limitaciones más relevantes de este dataset y arquitectura en particular.

La clase *car* se trata de una clase que posee unos bajos valores, tanto después de haber realizado el *transfer learning* como antes de hacerlo cuando se tenían las dos clases por separadas. Aunque la precisión ha aumentado ligeramente, hay que tener en cuenta que en esta ocasión, como la clase *car* es la unificación de estas dos clases se esperaría un mayor aumento de precisión. Por otro lado, el recall también ha disminuido si los comparamos con

los valores obtenidos anteriormente. Esto se traduce en un leve aumento de la métrica F1 con respecto a la clase *moving car* pero no así para *static car*. Finalmente, observando los valores obtenidos para la intersección de esta clase, dicho valor ha sido del 39,12 %, se trata de un valor más bajo que el obtenido para la clase *static car*. Por ello mismo, se puede considerar que esta clase para las imágenes de este dataset el modelo no está realizando una correcta segmentación semántica.

Finalmente, se encuentra la clase *clutter* a pesar de que ha sufrido un aumento de la precisión, no ha sido suficiente para amortizar la bajada del recall. Esto hace que, la métrica F1 haya disminuido de 0.8224 a 0.7664. Como es lógico, esta bajada de la métrica F1 también se puede ver reflejada en la intersección de los píxeles de esta clase con respecto a las correspondientes máscaras, y es que la intersección ha disminuido de un 69,83 % a un 62,12 %, lo que supone una disminución de más de un 7 %. Por esta misma razón, se puede afirmar que para la clase *clutter* el *transfer learning* no ha mejorado sus valores.

Analizando por último las métricas obtenidas globalmente, se observa que en este caso, como los valores de precisión, recall y F1 son distintos, a diferencia de lo que ocurría en análisis anteriores, en este caso, esto es debido a que en esta tabla no se encuentra el total de píxeles del conjunto de testeo, debido a que habrán otros píxeles que sean de clases del dataset rural, y por ello, estos valores en esta ocasión han cambiado. Dicho esto, se observa como se posee una precisión de un 0.8612, lo que supone una bajada en cuanto a la precisión ya que anteriormente dicho valor ha sido de 0.8632. Lo mismo ocurre con el recall, el cual ha pasado de un 0.8632 a un 0.8604. Esta sutil bajada de los valores de precisión y recall se observa también en el valor obtenido en la métrica F1, y es que anteriormente se obtuvo un valor de 0.8632, y ahora este valor es de 0.8608. Con respecto a la intersección de manera global, ésta también ha sufrido un leve descenso pasando de intersectar un 75,94 % a un 75,56 %. Cómo se observa, las disminuciones de los valores han sido bastante bajas, pero como lo que se desea con un *trasfer learning* es mejorar los datos obtenidos, no se puede considerar que en este caso los valores lo hayan hecho.

Habiendo analizado clase por clase con los datos obtenidos antes de realizar el *transfer learning*, y habiendo visto que no hay mejora de los datos, es fácil deducir que el *transfer learning* realizado en la subsección anterior en el que tan solo se quería realizar una segmentación de las clases existentes en el dataset urbano son mejores que con el *transfer learning* realizado en esta ocasión. Esto se deduce fácilmente tan solo observando los valores obtenidos de forma global, y es que si observamos la ilustración 7.26 se aprecian como los valores de precisión, recall y F1 han sido de 0.8664, mientras que este valor ahora ha sido de 0.8612, 0.8604 y 0.8608 respectivamente. Además, en cuanto a la intersección de todas las clases, anteriormente se había obtenido un valor de 76,43 %, mientras que este valor ahora es de 75,56 %, lo que supone una disminución. Cómo se observa, en este caso el *transfer learning* realizado anteriormente en el que tan solo se quería segmentar las clases presentes en el dataset urbano, posee una mejor segmentación para el mismo dataset. De esta manera, se afirma que tan solo se mejoran los resultados para el dataset rural pero no así para el dataset urbano, en donde anteriormente con el otro *transfer learning* se habían obtenido unos mejores resultados.

7.7.2.2. Con entrenamiento de las clases pertenecientes al dataset urbano

En esta ocasión, se va a realizar lo mismo que se hizo en el segundo apartado de la subsección anterior, es decir, se va a coger como base el modelo entrenado con el dataset rural pero con aquellas clases que no se encuentren en el dataset urbano puestas como clase *clutter*, pero en esta ocasión se va a entrenar dicho modelo con las imágenes del dataset rural (con las clases que no están en el dataset urbano puesta como clase *clutter*) y con las imágenes del dataset urbano. De esta manera, lo que se pretende con este entrenamiento es analizar como segmenta las clases que se encuentran en el dataset urbano en ambos datasets si se entrenan con los dos.

Por ello, una vez que se ha realizado el entrenamiento nombrado, se han obtenido las métricas para el conjunto de testeo del dataset rural y urbano, y se han obtenido como resultados las que se muestran en la ilustración 7.32.

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Forest	0.8996	0.9068	0.9445	0.9032	0.8234
Residential	0.9184	0.9494	0.9656	0.9336	0.8755
Land	0.8991	0.8726	0.9572	0.8856	0.7948
Road	0.8705	0.8784	0.9806	0.8744	0.7768
Person	0.8005	0.5140	0.9987	0.6520	0.4556
Car	0.7143	0.4744	0.9920	0.5701	0.3987
Clutter	0.8728	0.8659	0.9533	0.8694	0.7689
Global	0.8959	0.8959	0.9703	0.8959	0.8115

Ilustración 7.32: Testeo del entrenamiento del dataset rural y urbano al realizarle un *transfer learning* partiendo del entrenamiento del dataset rural únicamente con las clases que aparecen en el dataset urbano.

- **Testeo dataset rural**

Tal y como ocurría con el *transfer learning* mencionado en la subsección anterior, en esta ocasión los valores de las métricas que se están observando incluyen las imágenes de testeo tanto del dataset rural como del urbano. Por ello, para hacer un análisis de ambos datasets por separados se han obtenido las métricas para cada conjunto de testeo de cada dataset. En este sentido, se han obtenido como métricas para el conjunto de testeo del dataset rural las que aparecen en la ilustración 7.33.

Por tanto, a continuación se analizarán los resultados presentados en la ilustración 7.33 comparándolos con los datos obtenidos con el conjunto de testeo con el entrenamiento de este dataset rural sin haberle realizado ningún tipo de *transfer learning* pero con el entrenamiento que se realizó con la clase *clutter* en aquellas clases en las que no aparecía en el dataset urbano, es decir, estos valores se compararán con los obtenidos en la ilustración 7.27.

Por ello mismo, se comenzará analizando la clase *forest*, esta clase ha aumentado considerablemente sus valores de precisión y de recall. Este aumento se ve reflejado directamente en

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Forest	0.9602	0.9605	0.9733	0.9604	0.9238
Residential	0.9543	0.9640	0.9786	0.9592	0.9215
Land	0.9582	0.9364	0.9787	0.9472	0.8997
Road	0.9126	0.9077	0.9942	0.9101	0.8351
Person	0.8067	0.6587	0.9986	0.7552	0.5689
Car	0.7525	0.7959	0.9989	0.7736	0.6308
Clutter	0.9544	0.9689	0.9876	0.9616	0.9260
Global	0.9549	0.9549	0.9871	0.9549	0.9137

Ilustración 7.33: Testeo del entrenamiento del dataset rural al realizarle un *transfer learning* partiendo del entrenamiento del dataset rural únicamente con las clases que aparecen en el dataset urbano.

la métrica F1 y es que se ha mejorado de 0.9400 a 0.9604, lo que lo hace un valor bastante elevado. Esto mismo ocurre con la intersección de los píxeles de esta clase, y es que ha aumentado de intersectar el 88,69 % a intersectar el 92,38 %, esto quiere decir, que para el conjunto de testeo más del 92 % de los píxeles de esta clase los está segmentando de forma correcta, lo que supone un dato muy bueno de segmentación semántica.

Y es que este aumento de los valores no solo es para esta clase, también lo ha hecho así la clase *residential*, en donde el incremento de los valores de precisión y de recall han hecho que el aumento en la métrica F1 sea de más de un 0.02, llegando en esta ocasión al 0.9592. Por último, la métrica IOU ha pasado de intersectar un 87,79 % a un 92,15 %. De esta forma, se observa como estas dos clases analizadas hasta el momento han supuesto una gran mejora en los resultados.

Las dos clases anteriores suponen una importante cantidad de píxeles en el dataset, y así lo es también la clase *land*, que aporta en este sentido una mejora en los valores obtenidos, viéndose reflejado en las métricas F1 e IOU, en donde se pasan de 0.9286 y 0.8667 a valores de 0.9472 y 0.8997 como valores de F1 y de intersección respectivamente. De esta manera, se deduce como las clases más predominantes en el conjunto de este dataset han mejorado considerablemente los resultados, teniendo aproximadamente más de un 90 % de intersección. Teniendo en cuenta que los datos sobre los que se ha sacado esta métricas corresponden al conjunto de testeo, se tratan de resultados muy buenos.

La clase *road* supone un menor número de apariciones en el dataset, pero los resultados que se han obtenido demuestra que han mejorado en todas las métricas, y es que si bien la precisión ha aumentado más de un 0.03, el recall lo ha hecho aún más pasando de un 0.8233 a un 0.9077. Esto ha hecho que el valor de F1 aumente hasta un 0.9101, y que la intersección se incremente de 74,28 % a un 83,51 %, esto es, proporcionando de esta manera un 9 % más de intersección de los píxeles de esta clase.

Por contra, se encuentra la clase *person*, esta clase es la única que no ha mejorado los valores en ninguna de las métricas, ni en precisión ni en recall. De este modo, es normal que el valor

de F1 haya decaído de un 0.7323 a un 0.7252. Asimismo lo que ocurre con la métrica IOU es similar, y es que ha disminuido de un 57,76 % aun 56,89 %. Esta clase por tanto no ha conseguido ninguna mejoría en los resultados al aplicarle *transfer learning*.

Sin embargo, no ocurre lo mismo con la clase *car*, y es que en esta ocasión a pesar de tener una menor precisión, el incremento del recall ha sido significativo, aumentando casi un 0.18. Este aumento considerable del recall ha supuesto que se haya obtenido un mejor valor para la métrica F1, en este caso se ha pasado de un 0.6383 a un 0.7736, como se observa el cambio ha sido considerablemente bueno pues el aumento ha supuesto casi un 0.14. El incremento también ha sido notorio con la métrica IOU, y es que se ha pasado de intersectar el 46,87 % al 63,08 %.

La última clase es *clutter*, en este caso, la precisión que se ha obtenido ha sido algo menor, pero en cambio, el aumento del recall ha sido mayor. La poca disminución de la precisión con este mayor aumento del recall hace que el valor de la métrica F1 sea mayor, pasando de un 0.9470 a un 0.9616. Este aumento también se refleja en la intersección de los píxeles de esta clase y es que se ha pasado de intersectar el 89,93 % de los píxeles a un 92,60 %, lo que supone un aumento de casi un 3 %. Por tanto se puede afirmar que esta clase de forma general ha obtenido unos mejores resultados que antes de aplicarle el *transfer learning*.

Finalmente, comparando los resultados de las clases de manera global, se observa una mejora en los valores de precisión, recall y F1, puesto que en este caso, dicho valor es de 0.9549 frente al 0.9341 obtenido anteriormente, de esta manera se observa como se obtiene claramente unos mejores resultados cuando se aplica este *transfer learning*. Además, si se observa el valor de IOU, éste ha pasado de intersectar el 87,64 % de los píxeles a intersectar ahora el 91,37 % de píxeles, reafirmando de esta manera como existe una mejora sustancial de los valores obtenidos en las métricas al aplicarle *transfer learning*.

- **Testeo dataset urbano**

Una vez realizada la comparación de los valores obtenidos de las métricas para el conjunto de testeo del dataset rural, ahora se analizarán los resultados obtenidos para los valores obtenidos para el conjunto de testeo del dataset urbano. En esta ocasión se han obtenido los valores que se muestran en la ilustración 7.34.

Por tanto, en este caso se van a comparar los valores obtenidos en la ilustración 7.34, con los valores que se habían obtenido en el entrenamiento sin haberle realizado *transfer learning*, es decir, con los valores que aparecen en la ilustración 7.22.

Se comenzará analizando la primera de las clases, es decir, la clase *forest*, esta clase corresponde como bien se ha mencionado anteriormente con la clase *tree*, y como se aprecia, en esta ocasión, han mejorado significativamente los valores de precisión y de recall. El aumento de los valores de esta métrica se reflejan automáticamente en el valor de F1, y es que se ha pasado de 0.8642 a 0.8853, suponiendo un aumento considerable. Además, la intersección para esta clase también ha aumentado, pasando del 76.09 % al 79.52 %, lo que demuestra una mejor segmentación semántica en las imágenes de testeo.

La siguiente de las clases es la clase *residential*, esta clase ha mejorado su precisión pero no así su recall, el cual este último ha disminuido levemente. Como resultado, el valor de la métrica

Nombre Clase	Precisión	Recall	Accuracy	F1	IOU
Forest	0.8770	0.8937	0.9423	0.8853	0.7942
Residential	0.9344	0.9523	0.9628	0.9432	0.8926
Land	0.8310	0.8398	0.9606	0.8354	0.7173
Road	0.8722	0.8830	0.9720	0.8776	0.7818
Person	0.3750	0.0008	0.9988	0.0015	0.0008
Car	0.7430	0.4365	0.9860	0.5499	0.3792
Clutter	0.8299	0.8093	0.9386	0.8195	0.6941
Global	0.8805	0.8805	0.9659	0.8805	0.7865

Ilustración 7.34: Testeo del entrenamiento del dataset urbano al realizarle un *transfer learning* partiendo del entrenamiento del dataset rural únicamente con las clases que aparecen en el dataset urbano.

F1 ha aumentado de 0.9417 a 0.9432, y es que el incremento de la precisión ha sido mayor a la disminución del recall. El IOU, también ha aumentado aunque en menor medida, puesto que se había obtenido un valor de 0.8898 en el primer entrenamiento, y en este caso dicho valor ha aumentado llegando hasta un 0.8926. De esta manera, se observa como los valores que se han obtenido en las dos clases analizadas han sido mayores al realizarle *transfer learning*.

La clase *land* ha sido una clase en la que se ha visto una mejoría notablemente alta, tanto en la precisión como en el recall. Estos cambios se reflejan directamente en la métrica F1, que ha aumentado de 0.7849 a 0.8354, representando un incremento de más de 0.05. Lo mismo ocurre con la intersección de esta clase, y es que se ha pasado de intersectar el 64,59 % a intersectar el 71,73 %, observándose de esta manera un incremento sustancial de más de un 6 %.

Seguidamente, la clase *road* se trata de una clase cuyos valores de precisión y de recall han aumentado, pero en menor medida que en las clases anteriores. Este aumento, aunque si bien ha sido mínimo, ha hecho que igualmente el valor de la métrica F1 haya aumentado pasando de un 0.8715 a un 0.8776. Lo mismo ocurre con la métrica IOU, y es que se ha obtenido un sutil aumento de dicho valor, llegando en esta ocasión el modelo ser capaz de intersectar el 78,18 % de los píxeles de esta clase.

Con respecto a la clase *person* ocurre algo diferente a lo visto hasta este momento para esta clase. Si se recuerda, todos los valores obtenidos eran de ceros, pero en esta ocasión, se han obtenido resultados diferentes. Cómo se observa, los valores obtenidos son muy bajos, pudiendo afirmar que no se está segmentando de forma correcta esta clase, pero hay que observar el porqué este incremento, aunque sea mínimo y siga sin segmentar de manera correcta esta clase. Al realizar el *transfer learning* con el dataset rural, el cual poseía también esta clase, y en el entrenamiento de este *transfer learning* volver a usar para entrenar a parte de estas imágenes urbanas, las rurales, se produce la identificación de algunos de los píxeles de la clase *person*.

La clase *car* se ve perjudicada en este *transfer learning*, y es que se recuerda que se ha

pasado a una unificación de dos clases, llamada *car*. En este caso aunque la precisión haya sido mayor que las de las clases anteriores por separado, el recall que se ha obtenido si ha sido bastante más bajo. Esto ha hecho que los valores de las métricas F1 e IOU hayan disminuido considerablemente, llegando este último valor a intersectar en tan solo un 37,92 %. Por lo tanto, se puede afirmar que la segmentación de esta clase además de no mejorar con respecto a los datos ya obtenidos, tampoco está realizando una buena segmentación semántica.

Por último, la clase *clutter* no ha mejorado sus resultados, se observa que la precisión ha disminuido más de los que ha aumentado el recall, esto ha hecho que el valor de la métrica F1 también haya disminuido sutilmente llegando hasta un valor de 0.8195. Finalmente, con la métrica IOU, se puede observar cuanto ha bajado la intersección de los píxeles para esta clase, y es que anteriormente se había obtenido una intersección de un 69,83 % y esta intersección ahora es de 69,41 %, se observa como a pesar de que no ha sido una gran disminución de los datos, ha sido una bajada, y esto indica que la segmentación para esta clase la hacía mejor sin el *transfer learning*.

A pesar de la disminución en la última clase, se ha observado que la mayoría de las clases han mejorado en los valores de las métricas obtenidas. Anteriormente se habían tenido unos valores de precisión, recall y de F1 globales de 0.8632, mientras que este valor para estas mismas métricas ahora con el *transfer learning* han aumentado hasta un valor de 0.8805, suponiendo así un incremento más que suficiente para deducir que con el *transfer learning* se han obtenido mejores resultados. Además, si se observa el valor obtenido para la intersección de manera global, el valor ha aumentado llegando hasta un porcentaje del 78,65 % de intersección de los píxeles para el conjunto de testeo del dataset urbano, ya que anteriormente dicho valor era de un 75,94 %. De esta manera, se puede afirmar que para este conjunto se obtienen unos mejores resultados una vez que se ha realizado el *transfer learning* descrito.

7.7.3. Conclusiones del *transfer learning*

Habiendo analizado este último *transfer learning*, se va a analizar cuales de los *transfer learning* realizados aportan mejores resultados para el dataset urbano. Anteriormente se había afirmado que el mejor *transfer learning* para segmentar semánticamente las imágenes del dataset urbano había sido el *transfer learning* que se partió del modelo entrenado con el dataset rural y todas sus clases, y este modelo entrenado luego con las imágenes del dataset urbano, en este caso los valores que se obtuvieron para el conjunto de testeo del dataset urbano fueron los ya mostrados en la ilustración 7.26. En este sentido, ahora se van a comparar los datos obtenidos de forma global en ese *transfer learning*, con este último que se acaba de analizar, es decir, con los resultados obtenidos en la ilustración 7.34. Cómo se observa, para el primer caso, se han obtenido valores de precisión, recall y F1 de 0.8664, siendo estos valores en este último *transfer learning* de 0.8805, lo que supone un aumento de valores para estas métricas. Esto mismo, ocurre con la métrica IOU, y es que se ha pasado de intersectar el 76,43 % de las imágenes predichas con las máscaras reales al 78,65 % de los píxeles del conjunto de testeo, esto ha supuesto un aumento del 2 % de intersección. De esta manera, se puede concluir afirmando que se obtienen unos mejores resultados en las métricas y por tanto de segmentación semántica con el último *transfer learning* que se acaba

de analizar.

De esta manera, se llega a una conclusión muy clara y concisa de los resultados obtenidos en los cuatro *transfers learnings* que se han llevado a cabo. Los resultados muestran una mejoría tanto para el conjunto de testeo del dataset rural como para el conjunto de testeo del dataset urbano con el último *transfer learning* realizado, que ha consistido en entrenar el modelo a partir del modelo entrenado con las imágenes del dataset rural con las clases que no se encuentran en el dataset urbano puestas como la clase *clutter*, y entrenarlo nuevamente con las imágenes del dataset rural (con las clases que no se encuentran en el dataset urbano puesto como la clase *clutter*) y el dataset urbano. De esta manera, a parte de poseer unos mejores resultados tal y como se han observado, se posee un modelo que es capaz de identificar la intersección de las clases que se encuentran en ambos datasets. Recordando que para el conjunto de testeo del dataset rural se ha obtenido un valor de intersección para sus clases de más de un 91 %, mientras que este mismo valor para el conjunto de testeo del dataset urbano ha sido de casi un 79 %. Considerando de esta manera, que se trata de un modelo que es capaz de segmentar semánticamente las imágenes aéreas de una manera bastante correcta.

A continuación, se observan algunas predicciones realizadas con el *transfer learning* que mejores valores de métricas ha proporcionado para ambos datasets. Por ello, en la ilustración 7.35 se observa dos predicciones realizadas sobre dos imágenes del conjunto de testeo del dataset rural, mientras que en la ilustración 7.36 se observa la predicción para dos imágenes del conjunto de testeo urbano.

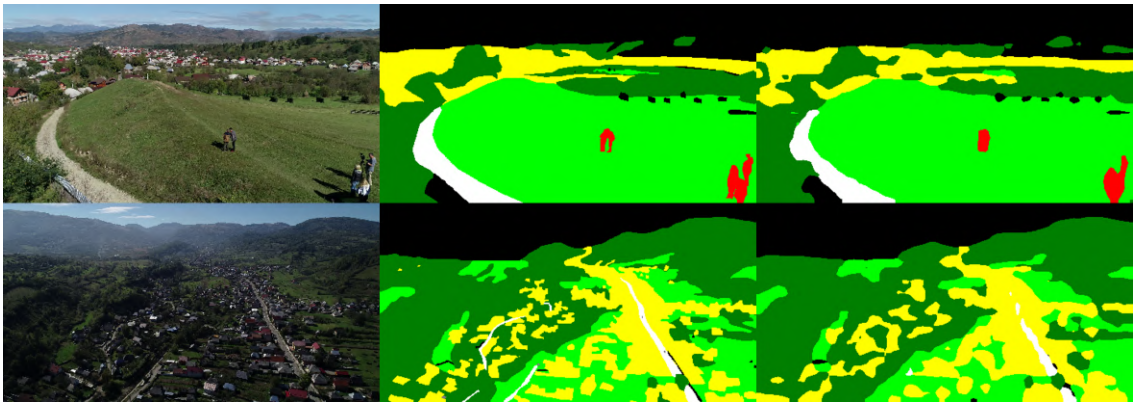

Ilustración 7.35: Algunas de las predicciones realizadas por el modelo con el mejor *transfer learning* con el conjunto de testeo del dataset rural. En el margen izquierdo se encuentra las imágenes reales, en la parte central las correspondientes máscaras y en el margen derecho se encuentran las predicciones.

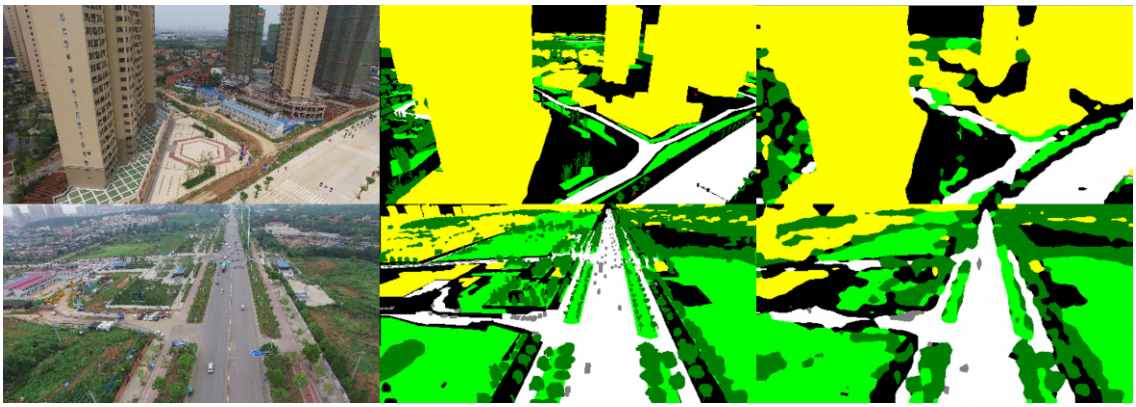

Ilustración 7.36: Algunas de las predicciones realizadas por el modelo con el mejor *transfer learning* con el conjunto de testeo del dataset urbano. En el margen izquierdo se encuentra las imágenes reales, en la parte central las correspondientes máscaras y en el margen derecho se encuentran las predicciones.

7.8. Validación con imágenes no etiquetadas

En esta sección se llevará a cabo la segmentación semántica de algunas de las imágenes que han sido proporcionadas por la empresa Aerolaser System S.L. Para ello, se me han proporcionado imágenes aéreas similares a las utilizadas para el entrenamiento y testeo. Estas imágenes han sido tomadas con una cámara de alta resolución, es por ello, que poseen una dimensión de 14204x10652 cada imagen. Pero para predecir sus máscaras, se han redimensionado al mismo tamaño que las imágenes con las que han sido entrenado el modelo, es decir, con un tamaño de 480x256.

Para las predicciones de las mismas, se ha utilizado el último modelo con *transfer learning* que se realizó. Se ha seleccionado este modelo debido a que, como se analizó anteriormente, ha demostrado ofrecer los mejores resultados en las métricas de los conjuntos de testeo tanto del dataset rural como del urbano.

A continuación, se observan algunas de las imágenes y sus predicciones con el modelo, para ello, hay que recordar la leyenda de los colores de las máscaras, en el cuadro 7.6 se observa cuales son las correspondencias entre las clases y los colores.

Una de las imágenes proporcionadas por Aerolaser System S.L. es la que se muestra en la ilustración 7.37. Esta imagen se ha pasado por el modelo de segmentación semántica que se ha entrenado y analizado, y se ha obtenido como máscara resultante la que se muestra en la misma ilustración al lado de su imagen. Para observar de una manera mejor la correspondencia entre la máscara y la imagen, en el margen derecho se observa la superposición de las dos anteriores.

Se observa en esta última, como la carretera en mayor medida la está segmentando de manera correcta, si bien, hay algunas partes de la misma que no es capaz de detectar de manera correcta. Por otra parte, las edificaciones también las está segmentando de una manera correcta, puesto que las edificaciones que se ven en la imagen están representadas en color amarillo,

lo cual indica que han sido correctamente segmentadas como edificaciones. Sin embargo, se observa cómo para esta misma clase se ha segmentado otras pequeñas regiones en la imagen correspondiente principalmente a la clase *forest*, o incluso a la rotonda. Además, se observa como los coches no los está segmentando de forma correcta, de esta manera, esta visualización permite corroborar cualitativamente los valores previamente analizados de manera cuantitativa. A pesar de no ser una segmentación extremadamente buena, se puede afirmar que está realizando una segmentación de la imagen de manera correcta, pero bastante mejorable.

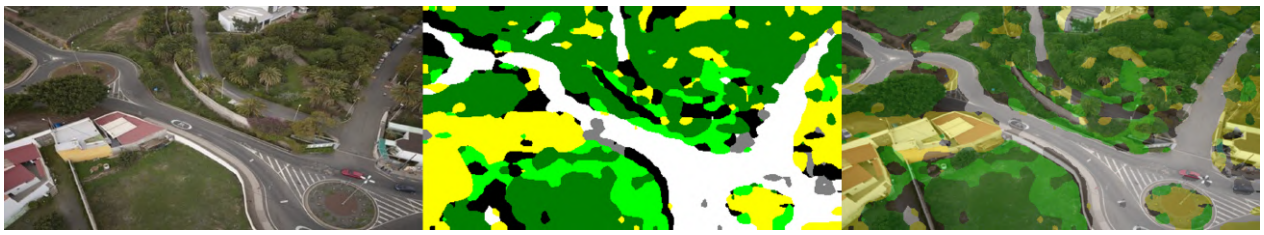

Ilustración 7.37: Primera de las imágenes proporcionadas por Aerolaser System S.L. con su máscara predicha por el modelo y la superposición de la máscara sobre la imagen. En el margen izquierdo la imagen de Aerolaser, en la parte central la máscara predicha por el modelo y en el margen derecho la superposición.

Otra de las imágenes que ha sido proporcionada por la empresa Aerolaser es la que se muestra en la ilustración 7.38. Al igual que se hizo para la imagen anterior, se ha superpuesto la imagen real y la predicción para facilitar el análisis visual. En este caso, se observa que está ocurriendo algo similar a lo visto anteriormente, y es que en la carretera en la parte más lejana, tiene dificultades para segmentar correctamente y la segmenta como *clutter*. Mientras que con respecto al resto de las clases, se observa como las clases *forest* y *land* se está segmentando casi que de manera perfecta, si no fuera porque hay pequeñas zonas en donde se está segmentando como la clase edificación, aunque hay que resaltar que la clase edificación si la está segmentado correctamente en las casas y edificaciones presentes en la imagen. En este caso, la segmentación realizada por el modelo sobre la imagen proporcionada por Aerolaser es relativamente fiel a lo que debería segmentar semánticamente.

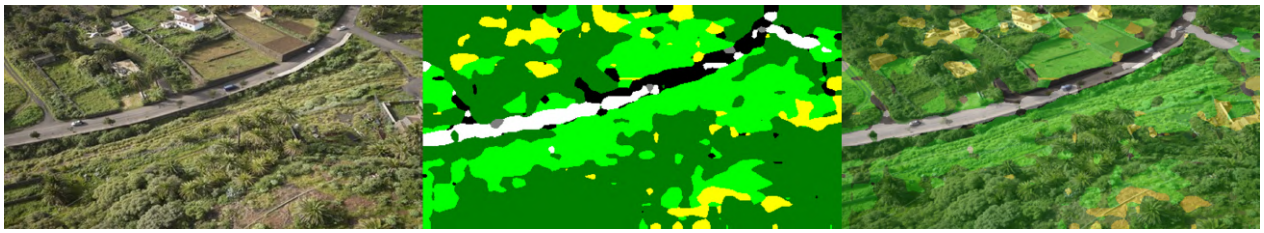

Ilustración 7.38: Segunda de las imágenes proporcionadas por Aerolaser System S.L. con su máscara predicha por el modelo y la superposición de la máscara sobre la imagen. En el margen izquierdo la imagen de Aerolaser, en la parte central la máscara predicha por el modelo y en el margen derecho la superposición.

Finalmente, se desea analizar una última segmentación semántica, ésta se observa en la ilustración 7.39. En este caso, se ha prestado especial atención a la segmentación de la clase

residential, ya que como se ha visto en las imágenes anteriores, se identificaron regiones erróneamente etiquetadas como esta clase en lugar de *forest* o *land*. Por ello mismo, se observa como se obtiene una predicción de la clase residencial más precisa, ya que como se observa se obtiene dicha clase en las propias casas y en parte de los terrenos de las mismas, y es que en este sentido se podría considerar como parte residencial a los terrenos de una casa debido que pertenece a la propia casa. Sin embargo, se observa como en el margen derecho de la imagen, se segmenta parte de un descampado con esta clase, cuando realmente no debiera de ser así. Finalmente, la clase *forest* se segmenta de una forma correcta, aunque el mayor problema lo obtiene en donde se encuentra la torreta de luz, y es que en esa zona no debería de predecir esta clase.

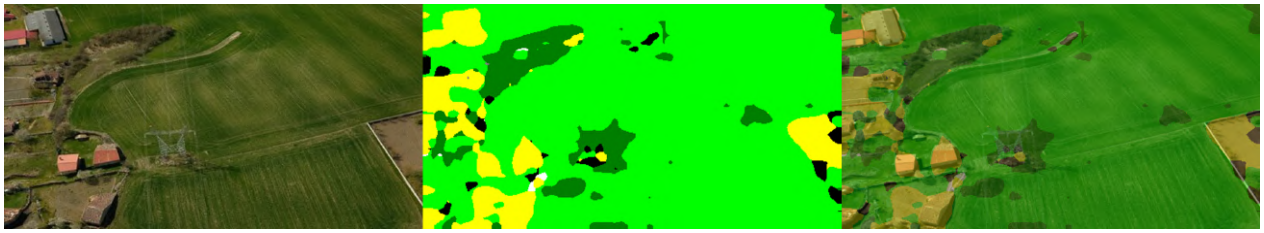

Ilustración 7.39: Tercera de las imágenes proporcionadas por Aerolaser System S.L. con su máscara predicha por el modelo y la superposición de la máscara sobre la imagen. En el margen izquierdo la imagen de Aerolaser, en la parte central la máscara predicha por el modelo y en el margen derecho la superposición.

En general, se ha observado que las predicciones realizadas por el modelo no son exhaustivamente precisas, aunque eso se sabía de antemano observando los valores que se han ido obteniendo a lo largo del entrenamiento. Por ello, de esta manera se concluye la validación de algunas de las imágenes proporcionadas por Aerolaser, confirmando los datos que se han obtenido en el entrenamiento, y es que aunque no se tratan de predicciones precisas, representa un buen punto de partida para buscar nuevas arquitecturas de segmentación semántica de imágenes aéreas.

Capítulo 8

Conclusiones y trabajo futuro

Como se ha ido observando a lo largo de este TFG, lo que se pretendía es poseer un modelo entrenado el cual sea capaz de segmentar semánticamente imágenes aéreas, en este caso tanto de imágenes rurales, como de imágenes urbanas. Para ello, se ha tenido que ampliar el conocimiento adquirido en distintas asignaturas de la carrera asociadas con la inteligencia artificial, y con algunas de las librerías usadas como lo es el caso de la librería Pytorch, de la cual, nunca hubiera trabajado con ella.

Para la obtención del modelo anterior, ha sido necesario la utilización de dos datasets distintos que recogieran las imágenes rurales y urbanas, además, se ha observado como en los datasets existían clases que se solapaban entre sí, este solapamiento ha permitido realizar un *transfer learning* que mejore la segmentación de las clases. No obstante, se ha observado que la distribución de las distintas clases en ambos datasets no se encontraban balanceadas, y esto ha favorecido a las clases que más píxeles poseían, y perjudicado a las que menos.

En este sentido, con el *transfer learning* realizado se ha podido mejorar en cierta medida algunos de los valores de las distintas métricas obtenidas, y es que con esta técnica el modelo ha sido capaz de obtener de manera global una mejor segmentación semántica.

Todos los resultados sobre los valores de las métricas mostrados en este trabajo, han sido como bien se ha indicado sobre el conjunto de testeo del dataset correspondiente. No se ha querido enseñar los valores de las métricas para los conjuntos de entrenamiento y validación, ya que se ha querido mostrar valores más cercanos a lo que se iba a obtener con la validación de imágenes no etiquetadas, y por ello, el conjunto de testeo ha sido el conjunto principal de estudio en cuanto a los valores de segmentación se refiere.

Como se ha podido observar, gracias a la validación que se ha realizado sobre imágenes no etiquetas, se ha comprobado como se ha obtenido una segmentación semántica de las imágenes proporcionadas por Aerolaser que se asemeja bastante a los resultados obtenidos sobre el conjunto de testeo. En las imágenes no etiquetadas no hay máscaras con que obtener métricas, de esta manera, se observado cualitativamente que en aquellas clases en las que se ha poseído menores valores en las métricas, han sufrido más en la segmentación semántica de las imágenes de validación.

Con este trabajo de fin de grado se ha obtenido un modelo interesante capaz de identificar de manera aproximadamente correcta ciertas clases, pero el modelo obtenido es muy mejorable, y en este sentido se abren dos frentes como posibles trabajos futuros. Por una parte, como trabajo futuro se encontraría la obtención de un tercer dataset que sea refuerzo de aquellas clases que menos píxeles poseen en los datasets que se tienen, de esta manera, se estaría tratando de balancear los datasets y por tanto se debería de obtener unos mejores resultados una vez realizados los correspondientes entrenamientos.

Por otra parte, se encuentra como posible mejora el cambio de arquitectura, ya que aunque se ha visto que la arquitectura DeepLabV3 proporciona unos mejores resultados con respecto a la arquitectura ResNetUNet, existen arquitecturas más modernas que harán que los valores de segmentación semántica aumenten, consiguiendo de esta manera una mejor segmentación semántica.

Con las dos posibilidades nombradas, se podría primero comenzar con la primera, de manera que se vuelva a entrenar con la misma arquitectura observando como se comporta ahora el modelo una vez que se encuentran los datasets balanceados. Mientras que sería más idóneo teniendo una vez los datos con las clases balanceadas, pasar a otra arquitectura y comprobar su desempeño.

Lo analizado, entrenado y validado en este trabajo de fin de grado, tan solo supone el comienzo de un trabajo mucho más amplio en el que se intentará realizar y llegar a una segmentación semántica de imágenes aéreas que se asemeje a la exactitud, ya que con las herramientas con las que se dispone hoy en día, se puede llegar a montar este sistema.

Bibliografía

[1] Accenture, FinlandEconomics (2018). Gráfica en la que se observa el crecimiento anual del pib esperado en 2036. https://www.expansion.com/blogs/sociedad-empresa-digital/2018/04/28/inteligencia-artificial-y-el-papel-del.html.

[2] ADE20K Team (2017). Ade20k. https://groups.csail.mit.edu/vision/datasets/ADE20K/.

[3] Adrian Rosebrock (2022). Intersection over union (iou) for object detection. https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/.

[4] ATRIA Innovation (2019). Qué son las redes neuronales y sus funciones. https://www.atriainnovation.com/que-son-las-redes-neuronales-y-sus-funciones/.

[5] Big Data (2021). Aprendizaje semi-supervisado con distribución de etiquetas. https://topbigdata.es/aprendizaje-semi-supervisado-con-distribucion-de-etiquetas/?utm_content=cmp-true.

[6] Cityscapes Team (2020). The cityscapes dataset. https://www.cityscapes-dataset.com/.

[7] COCO Team (2021). What is coco? https://cocodataset.org/#home.

[8] Data Science Team (2020). Redes neuronales residuales – lo que necesitas saber (resnet). https://datascience.eu/es/aprendizaje-automatico/una-vision-general-de-resnet-y-sus-variantes/.

[9] Escuela de Ingeniería Informática de la Universidad de Las Palmas de Gran Canaria (2019). Grado en ingeniería informática. https://eii.ulpgc.es/es/informacionacademica/tft.

[10] Estefanía Freire, Sarahí Silva (2019). Redes neuronales. https://bootcampai.medium.com/redes-neuronales-13349dd1a5bb.

[11] IBM (2023a). Aprendizaje no supervisado. https://www.ibm.com/es-es/topics/unsupervised-learning.

[12] IBM (2023b). Aprendizaje supervisado. https://www.ibm.com/es-es/topics/supervised-learning.

[13] ImageNet Team (2020). The imagenet dataset. https://www.image-net.org/download.php.

[14] Jose Martinez Heras (2020). Precision, recall, f1, accuracy en clasificación. https://www.iartificial.net/precision-recall-f1-accuracy-en-clasificacion/.

[15] Kabel (2020). Aprendizaje por refuerzos; ¿qué es? ¿cómo se usa? https://www.kabel.es/aprendizaje-refuerzos/.

[16] Laura Ruiz (2021). Neurona: qué es y cuáles son sus partes. https://www.psyciencia.com/neurona-que-es-y-cuales-son-sus-partes/.

[17] Lyu, Y., Vosselman, G., Xia, G.-S., Yilmaz, A., and Yang, M. Y. (2020). Uavid: A semantic segmentation dataset for uav imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 165:108 – 119.

[18] Madhuanand, L., Nex, F., and Yang, M. Y. (2021). Self-supervised monocular depth estimation from oblique uav videos. *ISPRS Journal of Photogrammetry and Remote Sensing*, 176:1–14.

[19] Marcu, A., Licaret, V., Costea, D., and Leordeanu, M. (2020). Semantics through time: Semi-supervised segmentation of aerial videos with iterative label propagation. *arXiv preprint arXiv:2010.01910*.

[20] Maziyar Panahi (2022). Scale vision transformers (vit) más allá de abrazar la cara. https://hackernoon.com/es/scale-vision-transformers-vit-beyond-hugging-face.

[21] Pascal VOC Team (2012). Pascal voc data sets. http://host.robots.ox.ac.uk/pascal/VOC/.

[22] Real Academia Española (2022). Inteligencia. https://dle.rae.es/inteligencia.

[23] Rubén Rodríguez Abril (2020). Redes residuales: Resnet. https://lamaquinaoraculo.com/computacion/redes-residuales/.

[24] The Segment Anything team (2023). Introducing segment anything: Working toward the first foundation model for image segmentation. https://ai.facebook.com/blog/segment-anything-foundation-model-image-segmentation/.

[25] Tractica, Statista (2017). Gráfica en la que se observa los ingresos globales de distintas aplicaciones de la inteligencia artificial entre los años 2016 a 2025. https://es.statista.com/grafico/9437/las-aplicaciones-mas-rentables-de-la-inteligencia-artificial/.

[26] Xeridia (2019a). Partes de una neurona artificial. https://www.xeridia.com/blog/redes-neuronales-artificiales-que-son-y-como-se-entrenan-parte-i.

[27] Xeridia (2019b). Partes de una neurona biológica. https://www.xeridia.com/blog/redes-neuronales-artificiales-que-son-y-como-se-entrenan-parte-i.