Segmentación Multimodal para detección de Animales en Entornos Naturales

Jorge Urbón Burgos
777295@unizar.es

Supervisor: Rosario Aragües
raragues@unizar.es

Co-Supervisor: Jesús Bermúdez
bermudez@unizar.es

2 de enero de 2025

Resumen

Índice general

Capítulo 1

Introduction

La segmentación de imágenes es una tarea fundamental en el campo de la visión por computador y ha sido ojeto de investigación en *Deep Learning* debido a su importancia en aplicaciones como la detección y clasificación de objetos. En este trabajo, se proponen diferentes modelos de segmentación multimodal a partir de imágenes infrarojas `IR` y mapas de profundidad `D` con el fin de obtener una segmentación y clasificación precisas de animales en un entorno natural. Para ello, se empleará un *dataset* más detallado en la sección 2.1.1 y se analizarán diferentes modelos del estado del arte, así como los resultados obtenidos con diferentes técnicas de pre-procesado. La introducción de técnicas multimodales en la segmentación de imágenes puede suponer una gran mejora en las tareas de pastoreo y vigilancia de animales debido a un incremento en la información disponible, lo que permitiría una mejor segmentación y clasificación de los animales en la escena.

1.1. Trabajos Relacionados

A pesar de que la segmentación de imágenes ha sido profundamente estudiada en la última década, suponiendo un gran avance en la implementación de sistemas de detección y clasificación de objetos. Aquí encontramos proyectos tan relevantes como `YOLO` ([**?**]), `Mask R-CNN` ([**?**]) o `DeepLab` ([**?**]). Sin embargo, la mayoría de estos modelos se han centrado en la segmentación de imágenes `RGB` o de una única modalidad, siendo que proyectos como [**?**] han demostrado la utilidad de contar con información de diferentes modalidades para la segmentación de imágenes. Aquí, podemos encontrar modelos como TODO: MODELOS MULTIMODALES DE EJEMPLO.

1.2. Objetivos

El objetivo de este trabajo es, por tanto, la verificación de la utilidad de las técnicas multimodales en la segmentación de imágenes de animales en un entorno natural con el fin de demostrar su utilidad en tareas como el pastoreo y la vigilancia de animales. Para ello, se analizarán diferentes modelos del estado del arte y se propondrán diferentes técnicas de pre-procesado. TODO: METER MAS COSAS Q VAYA HACIENDO Y TAL

Capítulo 2

Metodología

2.1. Datos

2.1.1. Lindenthal Camera Traps Dataset

Dado el objetivo que tiene este trabajo de verificar la utilidad de las técnicas multimodales en la segmentación de imágenes de animales en un entorno natural, se requiere de un *dataset* que contenga parejas de imágenes multimodales sincronizadas y etiquetadas para el entrenamiento del modelo. De entre los *datasets* públicos disponibles, el más completo y adecuado para este propósito se trata de `Lindenthal Camera Traps` presentado por [1], por lo que durante este proyecto se empleará una parte[1] del mismo. Este *dataset* está constituido por imágenes de animales en el Zoo de Lindenthal, Colonia, obtenidas con una cámara estéreo `Intel RealSense D435` y en el que se incluyen imágenes infrarojas (`IR`) con una profundidad de 8 bits y mapas de profundidad (`D`) con una profundidad de 16 bits en los que el valor de cada pixel representa la distancia en milimetros[2]. Ambas modalidades se encuentran ya alineadas y sincronizadas, lo que permite un fácil acceso a la información de ambas modalidades. Además, el etiquetado de los animales se encuentra en formato `COCO`. En la figura 2.1 se muestra un ejemplo de una pareja de imágenes `IR` y `D` del *dataset*. La captura de estas imagenes se realiza desde el tejado de un establo, lo que permite una vista elevada de los animales en la escena. Además, los datos se capturan exclusivamente cuando se da algun tipo de movimiento, por lo que en el *setup* indicado se incluye un sensor infrarojo pasivo (`PIR`) que actúa como sensor de movimiento.

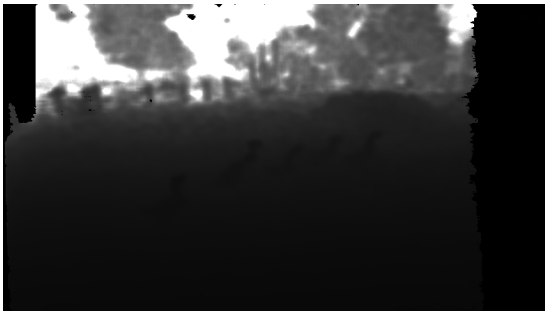
Desglose del *Dataset*

El *dataset* consiste en doce videos para los que se ha etiquetado cada décimo fotograma, lo que supone un total de 412 pares `IR-D` con 1038 instancias etiquetadas en formato `COCO` que incluyen la máscara, la caja delimitadora de la misma, la categoría y un identificador único para cada uno de los animales. Las categorías de animales presentes en

[1] A pesar de que este *dataset* cuenta con 775 secuencias de video que incluyen información `RGB` durante el día o `IR` durante la noche, así como mapas de profundidad en ambos casos, tan solo fueron etiquetados para su uso como *Ground Truth* 12 videos nocturnos, de los cuales se etiqueraría cada décimo fotograma.

[2] El sensor `Intel RealSense D435` cuenta con la capacidad de obtener imágenes estéreo de forma activa empleando ambas cámaras infrarojas mediante la proyección de un patrón infrarojo en la escena, sin embargo al haberse construido este dataset a partir de imagenes nocturnas en las que dicho patrón sería visible al sensor `IR`, se optó por generar los mapas de profundidad `D` mediante el uso de la cámara estéreo en modo pasivo.

(a) Imagen IR



(b) Imagen de profundidad

Figura 2.1: Ejemplo de una pareja IR-D del *dataset* Lindenthal Camera Traps

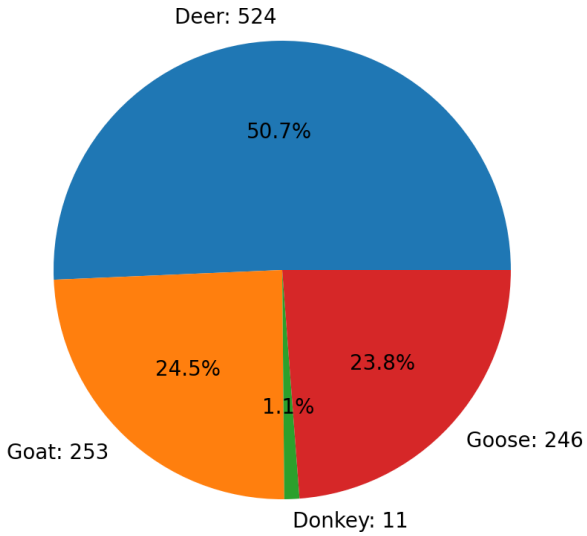


Figura 2.2: Desglose de la frecuencia de aparición de cada categoría en el dataset

el dataset son **Deer**, **Goat**, **Donkey** y **Goose**, siendo considerado todo lo demás descartable, es decir, parte de la categoría **Background**. En la figura 2.2 se muestra un desglose de la frecuencia de aparición de cada categoría en el dataset y se puede observar el amplio desbalanceo entre distintas categorías, por lo que cabe esperar que el modelo tenga ciertas dificultades a la hora de segmentar las categorías menos frecuentes como por ejemplo **Donkey**. Por otra parte, este *dataset* se mezclará aleatoriamente y se dividirá en dos partes: el 85 % de las imágenes se emplearán para entrenamiento y el 15 % restante para validación de los resultados con el fin de aplicar técnicas como el `EarlyStopping` o simplemente para una verificación manual del buen entrenamiento durante el proceso. En la figura 2.3 se muestra un ejemplo de un par IR-D con sus respectivas máscaras de segmentación extraidas del etiquetado COCO.

2.2. Modelo

El modelo a implementar, `CrossModalFusion` (CMX), fue presentado por [2] y propone una arquitectura de fusión de características de forma interactiva implementando rectificación de características inter-modal y bidireccional, así como atención cruzada se-

(a) Imagen IR etiquetada

(b) Imagen D etiquetada

Figura 2.3: Ejemplo de una pareja IR-D del *dataset* Lindenthal Camera Traps

cuencia a secuencia, lo que permite interacciones inter-modales más ricas y efectivas. La arquitectura propuesta por [2] se muestra en la figura 2.4, y su principal característica se encuentra en la rectificación inter-modal de las características a la salida de cada una de las capas del encoder, así como en la fusión de estas características rectificadas para su uso como características de menor nivel en la entrada al decoder. Además, este modelo incluye un *backbone* con pesos pre-entrenados en ImageNet que permiten un aprendizaje más efectivo de las características de la escena gracias a la gran cantidad de datos de entrenamiento con los que cuenta ImageNet. Sin embargo, dado que la versión de ImageNet empleada es puramente RBG, se requiere de un entrenamiento algo más intensivo para adaptar dichos pesos a las nuevas modalidades de entrada IR y D.

2.2.1. Entrenamiento

El entrenamiento del modelo es realizado en un portátil con una tarjeta gráfica NVIDIA GeForce GTX 1650 con 4GB de memoria VRAM y un procesador Intel Core i5-10300H con 8 núcleos y 16 hilos, así como 16GB de memoria RAM. Debido a los bajos resursos de los que se dispone, se optará por un entrenamiento en una única GPU y un tamaño de *batch* de 2. Además, como se ha mencionado en el anterior apartado, se empleará un *backbone* pre-entrenado en ImageNet y se aplicará la técnica de *transfer learning* para adaptar el modelo a las características de los datos de entrada. Se empleará la función de pérdida CrossEntropyLoss y el optimizador Adam con una tasa de aprendizaje de 10^{-6} dinámica. El modelo se entrenará durante 500 épocas con un tamaño de *batch* de 2. Por último, para fomentar el aprendizaje de las características de forma independiente en ambas modalidades, se realizará un *modality dropout* con una probabilidad de 0.5 decreciente con el número de épocas con el fin de que el modelo aprenda a desenvolverse con una sola modalidad en el caso de que, por el motivo que sea, la otra no esté disponible.

Además, como se precisa en la subsección 2.3, se analizarán diferentes técnicas de pre-procesado con el fin de determinar cuál puede aumentar la información disponible para el modelo y, por tanto, mejorar la segmentación de las imágenes.

2.3. Técnicas de Pre-Procesado

Cuando vamos a usar una imagen como input para un modelo de segmentación, se debe tener en cuenta el formato en el que se desea representar la información que se le proporciona al modelo. En este caso, se cuenta con dos modalidades de entrada: IR y

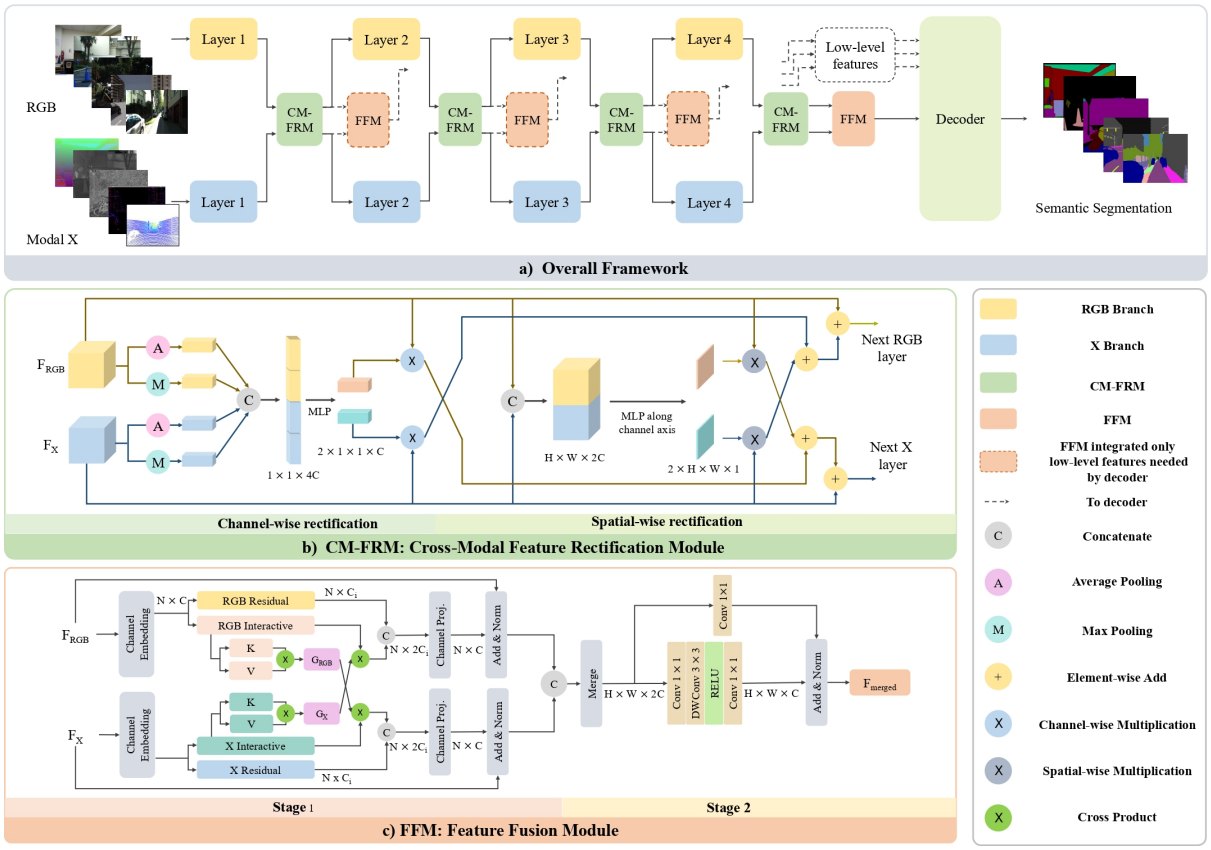


Figura 2.4: Arquitectura propuesta por [2]

`D`, por lo que se deben considerar las distintas técnicas de preprocesado que se pueden aplicar a cada modalidad para obtener un mejor resultado en la segmentación.

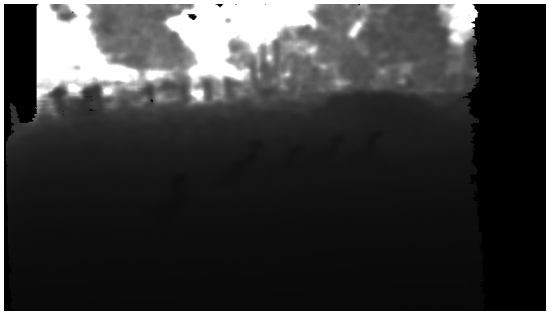
Así como aumentar la información que otorga el canal `IR` es una tarea complicada, ya que la imagen ya cuenta con una calidad aceptable, el canal `D` es el que más ruido presenta y, quizá, el que el modelo tenga más dificultades de entender debido a su escasa similaridad con los datos de pre-entrenamiento en `ImageNet`. Por tanto, se propone aplicar las técnicas de pre-procesado exclusivamente al canal `D` con el fin de aumentar la información disponible para el modelo e, idealmente, que este aprenda a emplearlos con el fin de mejorar la segmentación de las imágenes.

Existe una variedad de técnicas de pre-procesado que aprovechan los tres canales de entrada del modelo para así lograr un *input* capaz de aportar mayor cantidad de información al modelo de segmentación. A continuación, se presentan las técnicas de pre-procesado que se proponen para este trabajo.

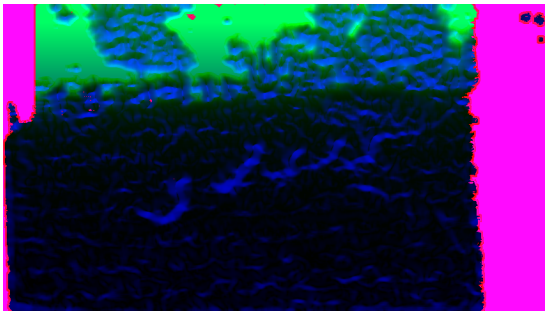
2.3.1. HHA Encoding

Esta técnica, propuesta por [3] usa los tres canales de la imagen de entrada para codificar las siguientes tres características:

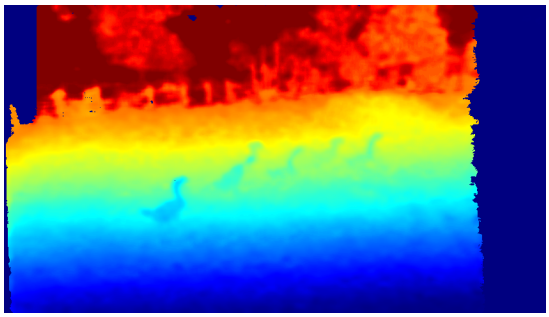
- Altura sobre el suelo
- Disparidad horizontal
- Ángulo con respecto a la gravedad



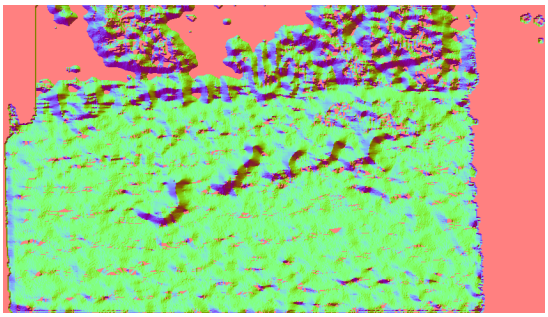
(a) Imagen de profundidad base



(b) Profundidad colorizada con HHA



(c) Profundidad colorizada con *Jet*



(d) Normales en la imagen de profundidad

Figura 2.5: Comparación entre las distintas técnicas de post-procesado

Estas colorización, además de implementar características que difícilmente serían aprendidas por el modelo si no se codificaran en la imagen de entrada, aprovecha los tres canales de entrada al *encoder*, por lo que los pesos ya preentrenados en *ImageNet* pueden ser empleados para una mejor comprensión de la escena. A pesar de que esta colorización permite calcular las características mencionadas, el tiempo de cómputo necesario para realizar esta operación puede ser elevado, por lo que la aplicación de esta técnica podría limitar los casos de uso en los que se emplee. Un ejemplo de esta técnica se muestra en la figura 2.5b.

2.3.2. Colorizacón Jet

Como se propone en [4], otra posible técnica es la colorización de la imagen de entrada usando el esquema de color *Jet* como se puede ver en la figura 2.5c, lo que implica la asignación de un valor *RGB* a cada píxel dependiendo de su valor de intensidad. Esto, al contrario de lo que sucede en 2.3.1, no amplía la información disponible para el modelo de ninguna forma, ya que en la imagen de entrada se sigue representando la distancia de los objetos a la cámara, pero podría resultar en mayor facilidad de aprendizaje para el modelo debido al uso de los pesos pre-entrenados en `ImageNet` y el mayor contraste entre los objetos en la escena. Este proceso, como se menciona en [4], puede ser más rápido que el de HHA, pero no necesariamente más efectivo.

2.3.3. Colorización por Distancia

Considerando los resultados obtenidos por Jet, se propone en una técnica de colorización que, si bien puede no resultar tan visualmente atractiva para el ojo humano como su alternativa, podría codificar la distancia de forma más efectiva para el modelo al establecer una relación lineal entre los valores de profundidad y el valor de los canales de la

imagen final. De tal forma, esta técnica también se basa en asignar un valor `RGB` a cada píxel dependiendo de su valor de intensidad, pero en este caso se determinará que los objetos más lejanos serán más dominados por el canal `R`, mientras que los objetos más cercanos serán más dominados por el canal `B` pasando por el canal `G` en el rango intermedio. La interpolación de los valores de distancia a estos valores podría ser aprendida por el modelo de forma algo más efectiva que en el caso de Jet a pesar de su menor contraste al ojo humano.

2.3.4. Normales

Siguiendo lo propuesto por [4], una mejor y más rápida solución de postprocesado consiste en la codificación de las normales en la imagen de entrada. Esta técnica consiste en calcular el vector normal de la superficie de cada pixel partiendo de la imagen de profundidad y codificarlo en los tres canales del *input* del modelo. Esto permite al modelo aprender características de la escena que de otra forma serían difíciles de aprender, como la orientación de los objetos en la escena. La imagen final, mostrada en la figura 2.5d, se obtiene calculando las derivadas parciales respecto a x e y de los valores de profundidad en la imagen y normalizando el vector resultante como se muestra en la ecuación 2.1.

$$\begin{aligned} \hat{n} &= \frac{(n_x, n_y, 1)}{\sqrt{n_x^2 + n_y^2 + 1}} \\ n_x &= \frac{\partial z}{\partial x}, n_y = \frac{\partial z}{\partial y} \end{aligned} \tag{2.1}$$

2.3.5. Comparación de Técnicas

Las distintas técnicas anteriores presentan diferentes resultados tras su evaluación, tal y como se puede ver en las matrices de confusión presentes en el anexo (TODO: referir al anexo!!!), en la tabla 2.1 se muestra la media de la precisión `IoU` obtenida por cada una de las técnicas de pre-procesado propuestas, donde se puede apreciar que la colorización `Jet` es la mejor opción cuando ambas modalidades están disponibles, mientras que no emplear ninguna técnica de pre-procesado es la mejor opción cuando solo se cuenta con la imagen `D` o `IR`. Esto se debe posiblemente a que TODO: expliar esta movida!!!

Además, como se puede ver en las matrices de confusión que se encuentran en el anexo (TODO: referir al anexo!!!) todas estas técnicas comparten una precisión extremadamente alta al diferenciar entre clases, siendo la confusión con el fondo la principal fuente de error en la segmentación de las imágenes. Un ejemplo del resultado de la segmentación de una imagen empleando colorización `Jet` se muestra en la figura **??**.

Técnica	**IR+D**	**Depth**	**InfraRed**
Sin Pre-Procesado	90.54 %	**74.56 %**	**90.51** %
HHA Encoding	90.60 %	41.83 %	89.78 %
Colorización Jet	**90.78 %**	73.54 %	90.23 %
Colorización por Distancia	89.92 %	73.58 %	88.70 %
Normales	90.76 %	72.49 %	88.22 %

Cuadro 2.1: Comparación de las técnicas de pre-procesado propuestas

Capítulo 3

Results

Bibliografía

[1] T. Haucke and V. Steinhage, "Exploiting depth information for wildlife monitoring," 2021. [Online]. Available: https://arxiv.org/abs/2102.05607

[2] J. Zhang, H. Liu, K. Yang, X. Hu, R. Liu, and R. Stiefelhagen, "Cmx: Cross-modal fusion for rgb-x semantic segmentation with transformers," 2023. [Online]. Available: https://arxiv.org/abs/2203.04838

[3] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," 2014. [Online]. Available: https://arxiv.org/abs/1407.5736

[4] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust rgb-d object recognition," 2015. [Online]. Available: https://arxiv.org/abs/1507.06821