Universidad de La Salle
Ciencia Unisalle

Ingeniería en Automatización | Facultad de Ingeniería

9-28-2020

Zonificación automática basada en imágenes de súper resolución

Elquin Mauricio Cascavita Ríos
Universidad de La Salle, Bogotá

Juan Andrés Baldion Góngora
Universidad de La Salle, Bogotá

Follow this and additional works at: https://ciencia.lasalle.edu.co/ing_automatizacion

Part of the Mechanical Engineering Commons, and the Other Engineering Commons

Citación recomendada

Cascavita Ríos, E. M., & Baldion Góngora, J. A. (2020). Zonificación automática basada en imágenes de súper resolución. Retrieved from https://ciencia.lasalle.edu.co/ing_automatizacion/784

This Trabajo de grado - Pregrado is brought to you for free and open access by the Facultad de Ingeniería at Ciencia Unisalle. It has been accepted for inclusion in Ingeniería en Automatización by an authorized administrator of Ciencia Unisalle. For more information, please contact ciencia@lasalle.edu.co.

ZONIFICACIÓN AUTOMÁTICA BASADA EN IMÁGENES DE SÚPER RESOLUCIÓN

Presentado por:

ELQUIN MAURICIO CASCAVITA RÍOS
JUAN ANDRÉS BALDION GÓNGORA

UNIVERSIDAD DE LA SALLE
FACULTAD DE INGENIERÍA
INGENIERÍA EN AUTOMATIZACIÓN
Bogotá D.C.
2020

ZONIFICACIÓN AUTOMÁTICA BASADA EN IMÁGENES DE SÚPER RESOLUCIÓN

Presentado por:
Elquin Mauricio Cascavita Ríos
Juan Andrés Baldion Góngora

Trabajo de grado para optar por el título
de Ingeniero en Automatización

PH.D. CESAR HERNÁN RODRÍGUEZ GARAVITO
DIRECTOR DEL PROYECTO

UNIVERSIDAD DE LA SALLE
FACULTAD DE INGENIERÍA
INGENIERÍA EN AUTOMATIZACIÓN
Bogotá D.C.
2020

Nota de aceptación:

_____
_____
_____
_____
_____

_____
Firma Director
Ing. Cesar Hernán Rodríguez Garavito

_____
Firma de Jurado
Ing. José Antonio Tumialan Borja

Bogotá D.C. 28 de septiembre de 2020

Agradecimientos

En primera instancia, quiero agradecer a Dios por haber tenido la oportunidad de que mis padres mantuvieran su fe en mí, dado que el camino de la universidad no fue fácil, pero tampoco fue imposible gracias por su apoyo incondicional, a un padre y una madre que sin experiencia en el tema intentaban impulsarme, y me permitieron realizarme como ingeniero.

Agradezco al destino cruzarme con mi compañero y haber generado tantos proyectos útiles que permitieron, además de crear una amistad, una relación de trabajo grupal que hizo posible este proyecto. Por otra parte, agradezco a los ingenieros y docentes que compartieron su conocimiento y su experiencia conmigo, de esta manera me generó una visión, una idea de lo que podría enfrentar en la vida laboral, además quiero agradecer a todos aquellos ingenieros que mantuvieron su confianza en mí y espero no haberlos defraudado. Un abrazo fuerte a todos los compañeros que conocí durante este camino, compañeros que me ayudaron cuando estaba atascado o de igual manera les ayudé en algún momento. Por último, a todas las personas que con el más mínimo apoyo me ayudaron para terminar este camino.

Espero que este proyecto sea de su agrado y que se pueda seguir investigando e impulsando esta área de Inteligencia Artificial dado que esto generará un gran avance a nivel tecnológico.

ELQUIN MAURICIO CASCAVITA RÍOS

En primer lugar, quiero agradecer a Dios por ser mi sustento y guía en todo momento, ya que sin él habría sido imposible llegar hasta este punto en el que me encuentro.

También le doy gracias a mis padres y mi hermano quienes me han apoyado en toda mi vida, en cada nueva etapa, siendo consejeros, guías y sobre todo, inculcándome valores para ser una persona que sea capaz de transformar la sociedad actual en la que vivimos.

Por último, le doy mi más sincero agradecimiento al ingeniero Cesar Hernán Rodríguez Garavito y al cuerpo de profesores de la universidad, pues siempre se han esmerado en trasmitir todo su conocimiento a los estudiantes de la mejor manera y se han preocupado porque cada persona pueda demostrar lo mejor de sí mismo. No sin olvidar a mis compañeros quienes fueron fundamentales para culminar con éxito una de las mejores etapas de mi vida.

JUAN ANDRÉS BALDION GÓNGORA

CONTENIDO pág.

INTRODUCCIÓN 3
1. MARCO TEÓRICO 4
1.1 SÚPER RESOLUCIÓN 4
1.1.1 SISR (SINGLE IMAGE SUPER RESOLUTION) 4
1.1.1.1 FILTROS ADAPTATIVOS DE WIENER 5
1.1.1.2 REGRESIÓN KERNEL ADAPTATIVA PARA ESPACIO TIEMPO 5
1.1.1.3 RECONSTRUCCIÓN ROBUSTA 5
1.1.2 MISR (MULTIPLE IMAGE SUPER RESOLUTION) 5
1.1.2.1 CARACTERÍSTICAS 6
1.1.2.2 POCS 7
1.1.2.3 MÉTODO BAYESIANO 7
1.1.2.4 ESTIMACIÓN PROBABILÍSTICA DEL MOVIMIENTO 7
1.1.2.5 REDES NEURONALES CONVOLUCIONALES 8
1.2 ZONIFICACIÓN 8
1.2.1 DETECCIÓN DE OBJETOS 9
1.2.2 SEGMENTACIÓN SEMÁNTICA 9
1.2.2.1 MÉTODO DEL VALOR UMBRAL 10
1.2.2.2 REDES NEURONALES CONVOLUCIONALES 10
1.2.3 SEGMENTACIÓN DE INSTANCIAS 11
2. GENERACIÓN DE IMÁGENES DE SÚPER RESOLUCIÓN 12
2.1. INTERPOLACIÓN LINEAL MODIFICADA 12
2.1.1. SIFT (SCALE INVARIANT FEATURE TRANSFORM) 16
2.1.2. RANSAC (RANDOM SAMPLE CONSENSUS) 16
2.1.3. AUMENTO DE TAMAÑO DE LA IMAGEN BASE 17
2.1.4. INTERPOLACIÓN LINEAL 18
2.1.5. UNIÓN DE IMÁGENES DE SÚPER RESOLUCIÓN 20
2.1.6. DESCONVOLUCIÓN CIEGA 20
2.1.7. RESULTADOS 22
2.2. ESRGAN 23
2.2.1. CARACTERÍSTICAS E IMPLEMENTACIÓN 24
2.2.2. RESULTADOS 26
2.3. PIX2PIX 27
2.3.1. ARQUITECTURAS GANs 27
2.3.1.1. U-NET 27
2.3.2. OBTENCIÓN DEL CONJUNTO DE DATOS DE ENTRADA 32
2.3.3. SEPARACIÓN DEL CONJUNTO DE ENTRENAMIENTO Y DE PRUEBA 32

2.3.4. AUMENTO Y LLAMADO DE DATOS ……32
2.3.5. DEFINICIÓN DE LA ARQUITECTURA.....33
2.3.6. ENTRENAMIENTO 33
2.3.7. RESULTADOS 35
3. ZONIFICACIÓN AUTOMÁTICA DE IMÁGENES.....36
3.1 OBTENCIÓN DE IMÁGENES DE ENTRADA ……36
3.2 DETECCIÓN DE OBJETOS ……..37
3.2.1 ARQUITECTURA FASTER R-CNN INCEPTION V2 ……....37
3.2.2 CARACTERÍSTICAS DE LA RED ……...38
3.2.3 OBTENCIÓN DEL CONJUNTO DE DATOS DE ENTRADA ……...39
3.2.4 ENTRENAMIENTO ……39
3.2.5 RESULTADOS ……40
3.3 SEGMENTACIÓN SEMÁNTICA ……...41
3.3.1 IMÁGENES DE SALIDA ……41
3.3.2 DEFINICIÓN DE PARÁMETROS ……43
3.3.3 ENTRENAMIENTO ……43
3.3.4 RESULTADOS ……45
4. RESULTADOS.....46
4.1 SÚPER RESOLUCIÓN ……...46
4.1.1 SIFT ……...47
4.1.2 ENTROPÍA ……...48
4.1.3 NITIDEZ ……...49
4.1.4 COMPARACIÓN FINAL ……50
4.2 ZONIFICACIÓN ……...52
4.2.1 ANÁLISIS DE RESULTADOS ……..54
4.2.1.1 EXACTITUD PIXEL A PIXEL ……54
4.2.1.2 MATRIZ DE CONFUSIÓN ……57
4.2.1.3 EXACTITUD, PRECISIÓN Y SENSIBILIDAD……59
5. IMPLEMENTACIÓN.....60
6. CONCLUSIONES 62
RECOMENDACIONES Y TRABAJOS FUTUROS ……64
REFERENCIAS 66
ANEXOS 70

LISTADO DE FIGURAS

Fig. 1-1. Diagrama de bloques de una imagen en alta resolución ……...6
Fig. 1-2. Estructura de una red neuronal convolucional ……...8
Fig. 1-3. Detección de objetos en una escena ……9
Fig. 1-4. Máscara de un conjunto de lagos en una imagen satelital ……..10
Fig. 1-5. Zonificación manual de una escena con zonas de interés ……..11
Fig. 1-6. Diferencias entre la detección de objetos, la segmentación semántica y la segmentación de instancias ……11
Fig. 2-1. Imagen base tomada en un cultivo de caña de azúcar ……15
Fig. 2-2. Sección de interés en la imagen original ……..15
Fig. 2-3. Algunos emparejamientos entre ambas imágenes ……..16
Fig. 2-4. Imagen base ampliada 3 veces con pixeles negros intermedios ……17
Fig. 2-5. Multiplicación de la imagen 1 por la matriz homográfica y obtención de la coordenada redondeada ……..18
Fig. 2-6. Representación de la interpolación lineal ……..19
Fig. 2-7. Zona de interés aumentada 3 veces ……..20
Fig. 2-8. Función de dispersión de punto ……..21
Fig. 2-9. Imagen de súper resolución de la zona de interés ……22
Fig. 2-10. Sección aumentada de la imagen original ……..……22
Fig. 2-11. Sección aumentada de la imagen de súper resolución por interpolación……23
Fig. 2-12. Arquitectura ESRGAN ……23
Fig. 2-13. Sección aumentada de la imagen original ……...26
Fig. 2-14. Sección aumentada de la imagen de súper resolución por ESRGAN ……26
Fig. 2-15. Arquitectura PIX2PIX ……27
Fig. 2-16. Arquitectura U-Net ……....28
Fig. 2-17. Comparación entre imagen reducida (izquierda) y original (derecha) ……32
Fig. 2-18. Imagen de entrada, imagen de salida e imagen generada durante el entrenamiento con sus respectivos tamaños ……34
Fig. 2-19. Pérdida total en las últimas épocas durante el aprendizaje de la red PIX2PIX para súper resolución ……...34
Fig. 2-20. Imagen de entrada, imagen de salida e imagen generada ampliadas sobre la zona de interés ……..35
Fig. 3-1. Imágenes aéreas seleccionadas para la zonificación ……..……..36
Fig. 3-2. Arquitectura Faster R-CNN Inception V2 ……..37
Fig. 3-3. Descripción de la red con alto costo computacional ……38
Fig. 3-4. Reducción dimensional de la capa de mayor tamaño ……..38
Fig. 3-5. Proceso de etiquetado en Labelimg ……...39
Fig. 3-6. Pérdida total en las últimas épocas durante el aprendizaje de la red Inception V2 ……40
Fig. 3-7. Imagen de entrada y objetos detectados por la red ……..40
Fig. 3-8. Segmentación manual utilizando Inkscape ……...42
Fig. 3-9. Imágenes aéreas seleccionadas segmentadas manualmente ……..42
Fig. 3-10. Imágenes de entrada, imágenes de salida e imágenes predichas en la primera época de la red de segmentación ……...43
Fig. 3-11. Imágenes de entrada, imágenes de salida e imágenes predichas en la época 150 ……...44

Fig. 3-12. Pérdida total en las últimas épocas durante el aprendizaje de la red PIX2PIX para zonificación ……...44
Fig. 3-13. Imágenes de entrada, imágenes de salida e imágenes predichas ……..45
Fig. 4-1. Comparación de la zona de interés en la imagen original, imagen obtenida por interpolación lineal modificada, ESRGAN y PIX2PIX ……..46
Fig. 4-2. Mejores emparejamientos entre la imagen original y las imágenes generadas por los distintos métodos ……..47
Fig. 4-3. Comparación de la zona de interés acercada en la imagen original, imagen obtenida por interpolación lineal modificada, ESRGAN y PIX2PIX con sus respectivos resultados ……...51
Fig. 4-4. Comparación de métricas de súper resolución ……51
Fig. 4-5. Imágenes de entrada e imágenes segmentadas por la red PIX2PIX ……53
Fig. 4-6. Diferencia entre la segmentación manual y la generada por la red PIX2PIX en el canal R de la primera imagen de prueba ……55
Fig. 5-1. Diferencia entre la imagen original y la generada por el método de interpolación lineal modificada ……60
Fig. 5-2. Zonificación de la imagen original y de la imagen de súper resolución …….60

LISTADO DE TABLAS

Tabla 3-1. Relación de la zona de interés con su respectivo color ……41
Tabla 4-1. Cantidad de puntos de emparejamiento de cada método con respecto a la imagen original ……..……..…...47
Tabla 4-2. Entropía de la imagen original y de la imagen obtenida con cada uno de los métodos ……..49
Tabla 4-3. Nitidez de la imagen original y de la imagen obtenida con cada uno de los métodos ……..50
Tabla 4-4. Porcentaje de acierto en cada categoría de la imagen 1 ……..55
Tabla 4-5. Porcentaje de acierto en cada categoría de la imagen 2 ……..56
Tabla 4-6. Porcentaje de acierto en cada categoría de la imagen 3 ……..56
Tabla 4-7. Porcentaje de acierto en cada categoría de la imagen 4 ……..56
Tabla 4-8. Porcentaje de acierto en cada categoría de la imagen 5 ……..56
Tabla 4-9. Matriz de confusión de la imagen 1 ……..57
Tabla 4-10. Matriz de confusión de la imagen 2 ……57
Tabla 4-11. Matriz de confusión de la imagen 3 ……58
Tabla 4-12. Matriz de confusión de la imagen 4 ……58
Tabla 4-13. Matriz de confusión de la imagen 5 ……58
Tabla 4-14. Exactitud, precisión y sensibilidad en las 5 imágenes de prueba ……..59
Tabla 5-1. Porcentaje de acierto en cada categoría de la imagen original ……61
Tabla 5-2. Porcentaje de acierto en cada categoría de la imagen de súper resolución .61

LISTADO DE ALGORITMOS

Alg. 2-1. Seudocódigo de la interpolación lineal modificada. ……..……12
Alg. 2-2. Seudocódigo de la implementación de ESRGAN. ...……..……24
Alg. 2-3. Seudocódigo de la red PIX2PIX para súper resolución……..……29
Alg. 4-1. Seudocódigo con las modificaciones realizadas al algoritmo 2-3……52

RESUMEN

En este trabajo se desarrolla una metodología de mejoramiento de imágenes y zonificación de áreas geográficas. En la primera parte, se utiliza un banco de imágenes tomadas por un dron de un determinado terreno o zona geográfica. A continuación, se implementan varias técnicas de súper resolución que permiten observar a detalle zonas específicas que antes no se podían detectar. Posterior a ello, se definen métricas de calidad que evalúan de forma cuantitativa el rendimiento de los algoritmos de súper resolución. Luego, se utiliza una estrategia de deep learning para identificar secciones específicas de zonas geográficas tales como zonas verdes, carreteras, zonas urbanas y cuerpos de agua de forma semántica. Todo esto con el fin de aprovechar al máximo una zona de cultivo, realizar planificación estratégica de pueblos o ciudades, observar daños en zonas protegidas, invasiones, conteo forestal, entre otros. Finalmente, se realiza un análisis cualitativo de los algoritmos planteados a partir de las métricas propuestas, concluyendo así que la mejor combinación de métodos es la interpolación lineal modificada para súper resolución, con una nitidez de 28.11, es decir, el doble que los demás métodos; y la red PIX2PIX para propósitos de segmentación de las categorías antes mencionadas en imágenes geográficas con un error de 26.31%, muy bueno en términos de segmentación.

INTRODUCCIÓN

En la actualidad, la generación de imágenes de súper resolución a partir de imágenes de baja calidad está en constante crecimiento dado que cada vez se le encuentra una mayor utilidad a este campo de aplicación. Existen muchas razones por las cuales es necesario este proceso, una de ellas es para extraer información que se encuentre en una imagen de baja calidad (Herrera, 2012).

Por otro lado, una de las aplicaciones que más fuerza está tomando es la de crear imágenes de súper resolución. Una de las preguntas es ¿porque es posible obtener imágenes de súper resolución? La respuesta a esta pregunta es porque se pueden adjuntar imágenes de baja resolución utilizando distintas técnicas para la agrupación de una misma escena a partir de muchas imágenes con pequeños desplazamientos entre ellas.

Existen diferentes enfoques de este desarrollo, pero en este caso específico se implementa sobre zonas rurales de las cuales se pueda obtener una optimización de recursos en grandes extensiones con el fin de analizar las diferencias entre áreas de cultivo, zonas que están desperdiciadas, regiones que podrán ser utilizadas con un mayor beneficio y el rendimiento en la planificación. Todo esto será a partir del análisis de los resultados con el propósito de obtener mayor rendimiento o de aprovechamiento en la precisión de agricultura. Por ejemplo, se sabe que el gobierno nacional posee un plan de desarrollo que está enfocado hacia cada uno de los departamentos en Colombia. El encargado de realizar esta gestión es el DNP (Departamento Nacional de Planeación). Sin embargo, al revisar el caso de Bogotá, se encuentra que la distribución de recursos para la planificación es la más baja con apenas 21.374 millones de pesos colombianos, es por ello que la planificación no ha hecho uso de herramientas tecnológicas puesto que no cuenta con los recursos suficientes. Por ello, se plantea el uso de drones en este campo. (Colombia, 2014)

Por estos motivos, se desarrolla este proyecto el cual consiste en implementar diversas técnicas para generar imágenes de súper resolución y para realizar la zonificación. En la primera sección se encuentran todas las definiciones relacionadas a estos dos conceptos. En la segunda sección se implementan algunos métodos para generar imágenes de súper resolución, destacando un método propio de los autores denominado como interpolación lineal modificada. En la tercera sección entrenan dos redes neuronales que detectan y segmentan objetos en una imagen con una base de datos general creada por los autores, lo cual permite generar un conocimiento global aplicado en un entorno local. En la cuarta sección se comparan los resultados de cada uno de los métodos implementados. En la quinta sección se prueban ambas herramientas en conjunto, pues se espera que la detección de zonas sea mejor, ya que la información artificial que se introdujo en la imagen de súper resolución se hace a partir de las texturas alrededor de un punto, lo cual provoca que la imagen tenga mejor información. La ventaja del trabajo propuesto es que es una herramienta modular, ya que brinda la posibilidad de utilizar únicamente la súper resolución o la zonificación, no necesariamente ambos, ya que depende del problema a resolver.

1. MARCO TEÓRICO

En este capítulo se mencionan los conceptos principales relacionados al tema de la súper resolución y la zonificación, es decir, las definiciones importantes y los métodos para llevar a cabo estos dos procedimientos.

1.1. SÚPER RESOLUCIÓN

Súper resolución es una técnica utilizada para aumentar la resolución de una imagen que está asociada con la cantidad de pixeles que una imagen posee, de tal manera que se puede decir que la súper resolución consiste en agregar pixeles intermedios entre los pixeles ya existentes.

También se define como la reconstrucción de mayor resolución de una imagen a partir de imágenes observadas, por lo tanto, se basa en la idea de que se puede utilizar una combinación de secuencia de imágenes de baja resolución (con ruidos) de una escena para generar una imagen o una secuencia de imágenes de alta resolución. De esta manera, se intenta reconstruir la imagen de una escena original con alta resolución. Todo esto tiene como propósito obtener imágenes que sean una reconstrucción más cercana de la realidad, mostrando detalles que en un principio no eran apreciables.

Para llevar a cabo la súper resolución se utilizan dos métodos; SISR (Single image Super Resolution) y MISR (Multiple Image Super Resolution). Estos dos a su vez, poseen varios submétodos, de hecho, existen miles de submétodos para llevar a cabo la súper resolución. Lo que difiere en ambos casos es que se toma un modelo base y a partir de ese se realizan modificaciones y/o adaptaciones para generar nuevos submétodos, es por esto que se van a mencionar los principales modelos base que se utilizan en súper resolución.

1.1.1. SISR (SINGLE IMAGE SUPER RESOLUTION)

Consiste en obtener una imagen de súper resolución a partir de una sola imagen. Es el método más común para generar esta clase de imágenes puesto que su costo computacional es menor, sin embargo, sus resultados no son los más precisos. SISR es desafiante porque el contenido de imagen de alta resolución no puede recuperarse normalmente de la imagen de baja resolución. Sin información de alta frecuencia, la calidad de la imagen de alta resolución es limitada. Además, SISR es un problema mal planteado porque una imagen de baja resolución puede producir varias imágenes de alta resolución posibles (MathWorks, 2019). Esto sucede porque solo se está tomando como referencia una sola imagen, por lo que no se está explorando el campo de visión a detalle, únicamente se tiene como referencia lo que hay en una sola imagen.

Para llevar a cabo SISR se utilizan distintos submétodos, entre ellos se encuentran:

1.1.1.1. FILTROS ADAPTATIVOS DE WIENER

En este método, para generar la imagen de salida de súper resolución, cada uno de los pixeles que componen la imagen se estiman a partir de un tipo de filtro adaptativo de Wiener. Cada pixel permite obtener una suma ponderada de los pixeles cercanos a él. El propósito de este filtro es que, utilizando métodos estadísticos, se reduzca el ruido presente en la señal de entrada (En este caso eliminar el ruido de la imagen de baja resolución) de tal modo que la señal de salida del filtro se aproxime lo más posible (minimizando el error cuadrático medio) a una señal deseada (Imagen de baja resolución con el menor ruido posible). Luego de obtener la imagen filtrada se hace la suma ponderada (Milanfar, 2014).

1.1.1.2. REGRESIÓN DE KERNEL ADAPTATIVA PARA ESPACIO TIEMPO

Esta aproximación se basa en la regresión multidimensional kernel, donde cada píxel en la imagen se aproxima a una serie local 3D (Taylor), capturando el comportamiento local esencial de su vecindad espacio-temporal. Los coeficientes de esta serie se estiman resolviendo un problema de mínimos cuadrados ponderados locales, donde los pesos son una función de la orientación tridimensional del espacio-tiempo en el vecindario. Como este marco se basa fundamentalmente en la comparación de píxeles vecinos tanto en el espacio como en el tiempo, implícitamente contiene información sobre el movimiento local de los píxeles a lo largo del tiempo (Takeda y Milanfar, 2017).

1.1.1.3. RECONSTRUCCIÓN ROBUSTA

El algoritmo propuesto reduce el número de estimaciones observadas de valores de píxeles de la imagen de baja resolución, utilizando un promedio de valores de píxeles en una región dividida. El efecto del algoritmo propuesto se demuestra con imágenes sintéticas e imágenes reales. Los resultados muestran que el algoritmo propuesto es aproximadamente 1.4 a 8.5 veces más rápido que los algoritmos convencionales (Masayuki y Okutomi, 2007).

1.1.2. MISR (MULTIPLE IMAGE SUPER RESOLUTION)

En este método, las imágenes de súper resolución se crean a partir de varias imágenes de baja resolución. Este método está tomando mucha fuerza últimamente ya que las máquinas son cada vez más robustas. Otra de las razones por las que se ha optado por este método es porque sus resultados son mejores con respecto a SISR, pues la imagen de súper resolución resultante se ha construido con base en muchas imágenes que han abarcado casi la totalidad del espacio de visión, esto significa que al tomar muchas imágenes de un mismo escenario, se pueden obtener imágenes que están corridas entre sí, presentan ligeras rotaciones, inclinaciones y demás deformaciones que permiten explorar a profundidad el campo de visión.

Para obtener una imagen de súper resolución con este método es necesario conocer en primer lugar cómo se compone una imagen de súper resolución. Una imagen de súper resolución está formada por un conjunto de imágenes de baja resolución, cada una de ellas con transformaciones geométricas, difuminación y escala.

Se puede representar una imagen de baja resolución de la siguiente forma:

$$LR = D_k H_k F_k \cdot HR \qquad (1\text{-}1)$$

Donde LR es la imagen de baja resolución.

Dk es el factor de escala de la imagen de baja resolución.

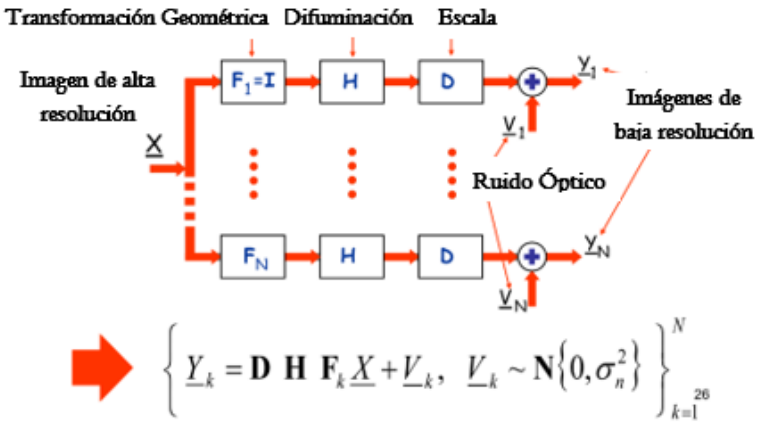
Hk es la difuminación de la imagen de baja resolución.

Fk es la transformación geométrica de la imagen de baja resolución.

HR es la imagen de súper resolución.

Así mismo, se puede expresar una imagen de baja resolución en diagrama de bloques.

Fig. *1-1.* Diagrama de bloques de una imagen en alta resolución.



Fuente: (Elad y Rubner, s.f.)

Donde Yn representa cada una de las imágenes que fueron tomadas en baja resolución. Xn representa la imagen de súper resolución. Entre mayor sea el número de imágenes de baja resolución, mayor definición tendrá la imagen de súper resolución.

1.1.2.1. CARACTERÍSTICAS

- TRANSFORMACIÓN GEOMÉTRICA

Son aquellas variaciones espaciales de una imagen con respecto a otra. Entre estas están la traslación y rotación en los 3 ejes posibles (Bedoya, s.f.). Se puede modelar con una matriz de transformación homogénea.

- RUIDO ÓPTICO

Es la variación aleatoria en los valores de píxel de la imagen. El ruido es uno de los problemas más difíciles de tratar a la hora de generar imágenes de súper resolución y para ello se deben utilizar combinaciones de filtros para eliminarlo en la mayor medida posible (FotoNostra, 2017).

- DIFUMINACIÓN

Disminución de la claridad y exactitud del contorno de una cosa, especialmente de un paisaje, una figura o un objeto. En este caso, los elementos de la imagen se ven borrosos (RAE, s.f.). La ventaja que existe es que, si la secuencia de imágenes de baja resolución es tomada en un lapso pequeño y con la misma cámara, la difuminación es prácticamente igual en todas las imágenes.

- ESCALA

Representa la relación que hay entre la imagen de súper resolución y la de baja resolución.

Al igual que en SISR existen muchos submétodos que se han ido adaptando de otros, los cuales se verán a continuación:

1.1.2.2. POCS

Se define como proyecciones sobre conjuntos convexos. La convexidad de una curva o una superficie, es la zona que se asemeja al exterior de una circunferencia o una superficie esférica, es decir, que tiene su parte sobresaliente dirigida al observador. El algoritmo POCS es una aproximación reducida e iterativa que ha sido utilizada para reconstruir imágenes de un espacio k truncado intentando extrapolar esa información truncada. Por lo tanto, POCS se convierte en uno de los métodos de restauración de súper resolución espacial más populares (Nakamura y Nakaura, 2016).

1.1.2.3. MÉTODO BAYESIANO

El objetivo es inferir la escena oculta a partir de la cual las imágenes de baja resolución son generadas, para ello, se utiliza el teorema de Bayes, en el que por medio de cálculos probabilísticos es posible estimar cuáles podrían ser los parámetros que generaron la imagen de baja resolución a partir de una imagen de alta resolución conociendo algunos factores como el gradiente. En otras palabras, se buscan obtener los parámetros que acompañan a la imagen de baja resolución (Tipping y Bishop, 2002).

1.1.2.4. ESTIMACIÓN PROBABILÍSTICA DEL MOVIMIENTO

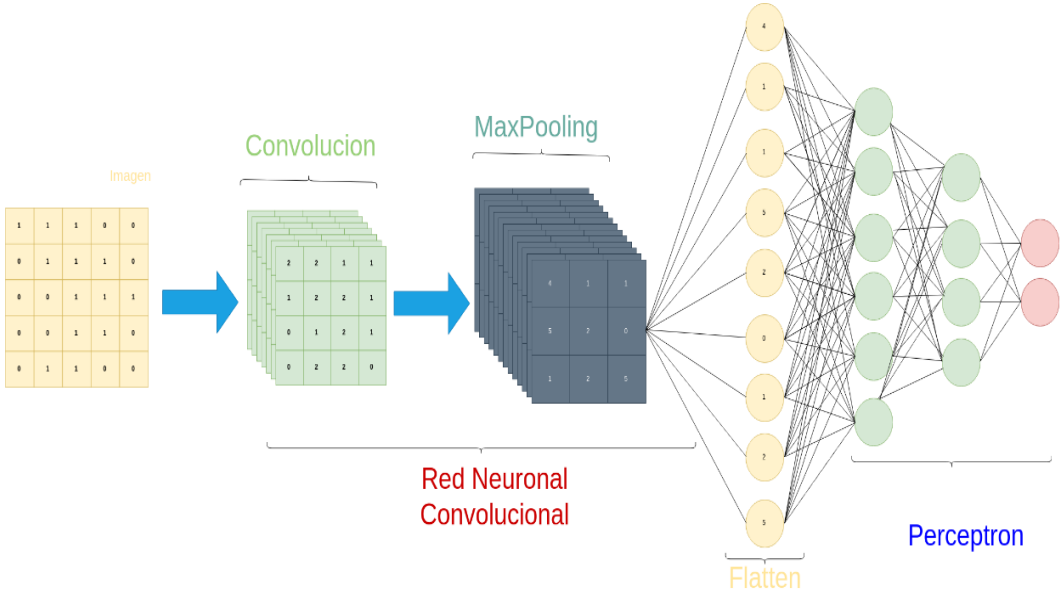
A pesar de su simplicidad, los algoritmos obtenidos son muy potentes en el manejo de las escenas más generales, con la estimación de movimiento

probabilística se permite el manejo de patrones de movimiento que son muy desafiantes, es decir, secuencias de imágenes con mucho movimiento. Las secuencias de imágenes resultantes son de alta calidad y contiene poca distorsión. Utilizando este método se puede llegar a estimaciones muy buenas de las matrices de transformación geométrica y difuminación (Elad y Protter, s.f.).

1.1.2.5. REDES NEURONALES CONVOLUCIONALES

Similar a como ocurre en SISR es necesario tener una imagen de referencia en alta resolución, sin embargo, esta imagen puede ser construida a partir de un mosaico de imágenes de baja resolución en las cuales se toman imágenes enfocando zonas más pequeñas, no todo el panorama completo. Al obtener esta imagen, se generan múltiples imágenes que tengan ruido, deformaciones geométricas y difuminaciones. Con ese banco de imágenes se entrena la red neuronal para que aprenda a identificar que es una difuminación, que es ruido y que es una transformación geométrica. Luego, al tener entrenada la red, se toman las imágenes de baja resolución y estas, al pasar por la red, van a ajustarse para obtener una imagen de súper resolución que se va a ir mejorando con cada imagen nueva que se vaya procesando (Kawulok y Benecki, 2019).

Fig. *1-2*. Estructura de una red neuronal convolucional.



Fuente: (Crispí, 2019)

1.2. ZONIFICACIÓN

Es la división de un área geográfica en sectores heterogéneos conforme a ciertos criterios (Anika, 2015). Para este caso específico, los criterios seleccionados son: agua, carretera o vías de transporte, zona urbana y zona verde.

En el estudio de visión por computador es importante tener en cuenta lo que se puede obtener visualmente a partir de una cámara, es decir, imágenes y videos. Con respecto a la detección de objetos se encuentra COCO (Common Objects

in Context) (Lin, 2015). También se encuentra Google, siendo uno de los más involucrados, permitiendo que las personas utilicen su motor más importante, es decir, el cerebro de todo lo que está basado Google en la plataforma digital, de esta manera permite que los investigadores tengan un estimulante para hacer sus estudios, por ejemplo, el uso gratuito de Google Colab o Tensor Flow.

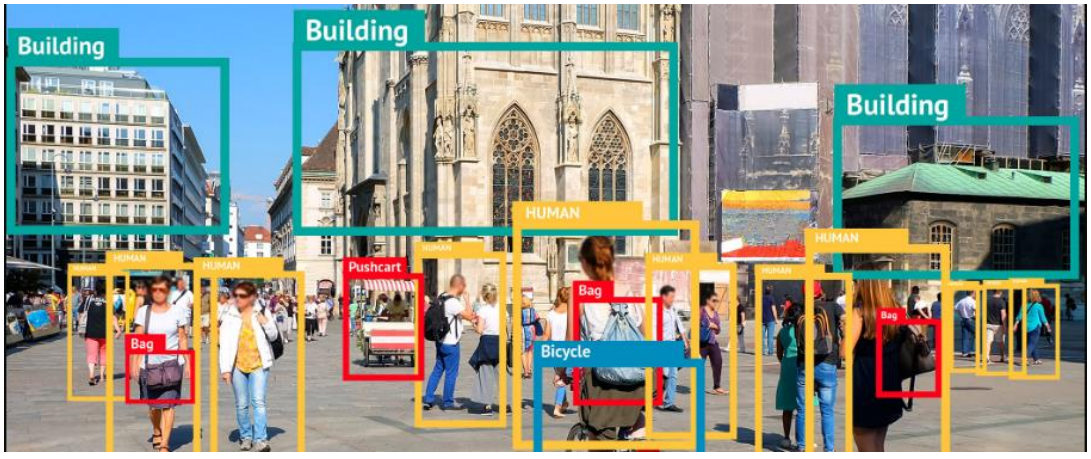
Estos dos principales estudios frecuentemente trabajan en conjunto. COCO posee dos características de detección de objetos de las cuales se encuentran: detección de objetos propiamente, el cual genera un “Bounding box” sobre cada objeto a identificar. Por otra parte, la segmentación de objetos conocida como la segmentación semántica, se encarga de separar, escoger y segmentar cada objeto dependiendo la categoría a la que pertenezca (Lin, 2015). Como fue mencionado anteriormente, Google está al pendiente de todo lo que pueda ser un avance sobre el estado del arte en la detección de objetos y por ello deja sus librerías con códigos abiertos para que cualquier persona pueda utilizarlos, manipularlos y realizar sus proyectos de investigación.

Con base a lo anteriormente mencionado, la zonificación puede ser llevada a cabo de 3 maneras las cuales difieren en la precisión en que son localizados y representados en la imagen los objetos de interés.

1.2.1. DETECCIÓN DE OBJETOS

Es una técnica de visión por computador que consiste en localizar instancias de objetos en imágenes y videos y colocar un recuadro o “Bounding box” con un color específico y una etiqueta sobre el objeto detectado.

Fig. *1-3*. Detección de objetos en una escena.



Fuente: (Bogado, 2018).

1.2.2. SEGMENTACIÓN SEMÁNTICA

Es un algoritmo de deep learning que asocia una etiqueta o categoría a cada píxel presente en una imagen. Se utiliza para reconocer un conjunto de píxeles que conforman distintas categorías (MathWorks, 2020). Por lo tanto, es una detección de objetos más robusta, pues además de detectarlos, se encarga de pintar todo el objeto con un color específico.

Para llevar a cabo la segmentación semántica, no existe un conjunto de métodos concretos, pues estos son productos de la experimentación de varios expertos, por tal motivo, los más destacados que se han encontrado son:

1.2.2.1. MÉTODO DEL VALOR UMBRAL

Son un grupo de algoritmos cuya finalidad es segmentar gráficos rasterizados, es decir, separar los objetos de una imagen que nos interesen del resto. Esto se realiza por medio de máscaras en donde la zona de interés es pintada en color blanco y el resto en color negro. Se le conoce también como “pixel-based” y trata básicamente de relacionar un rango de intensidad de pixeles con los objetos de interés.

Fig*. 1-4.* Máscara de un conjunto de lagos en una imagen satelital.

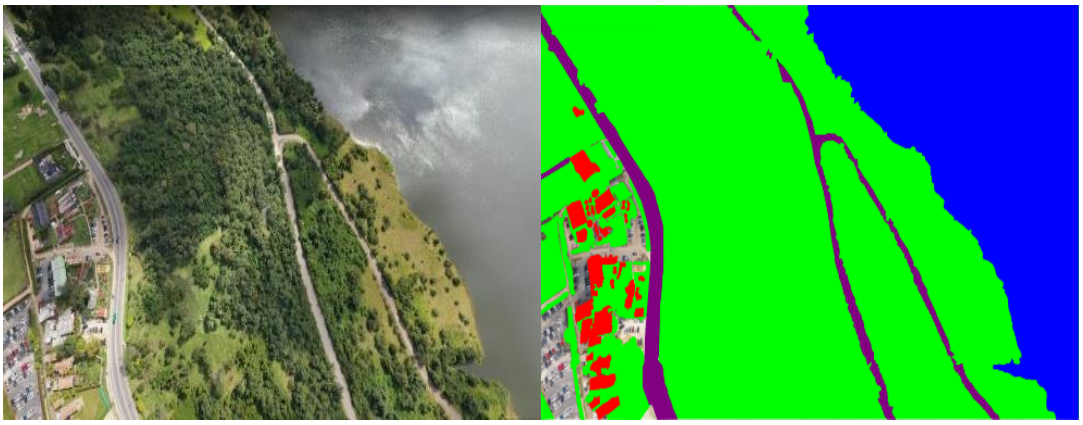
Fuente: (Gupta, 2020).

El problema de este método es que a pesar de ser rápido y brindar resultados muy buenos, solo se puede aplicar específicamente a una imagen, es decir, al tomar los valores umbrales de los pixeles de la imagen que corresponden, por ejemplo, al agua, como en este caso, es muy posible que estos mismos valores no apliquen en otra imagen que también posee agua, ya que a pesar de que las imágenes posean elementos en común entre sí, sus intensidades pueden variar mucho. Por ejemplo, existen ríos en los que el agua es de color azul turquesa mientras que existen otros en los que el agua es de color tierra. En estos casos en donde son el mismo objeto, pero sus intensidades varían, el método no es efectivo, pues es necesario realizar un procedimiento específico para cada imagen y no un método estándar que pueda realizar este proceso en una gran cantidad de imágenes de forma automática.

1.2.2.2. REDES NEURONALES CONVOLUCIONALES

El uso de redes neuronales es la mejor opción ya que ellas se encargan de aprender los patrones e intensidades asociados a cada objeto, por lo que le es más sencillo generar la máscara coloreada de cada uno de los objetos identificados. Sin embargo, es necesario tener en cuenta que no existe una arquitectura que sea 100% precisa; la arquitectura ResNet considerada la mejor en cuanto a segmentación, obtuvo un error cercano al 3.57% (Anwar, 2019), por lo que van a existir objetos que sean confundidos con otros.

Fig. *1-5*. Zonificación manual de una escena con zonas de interés.

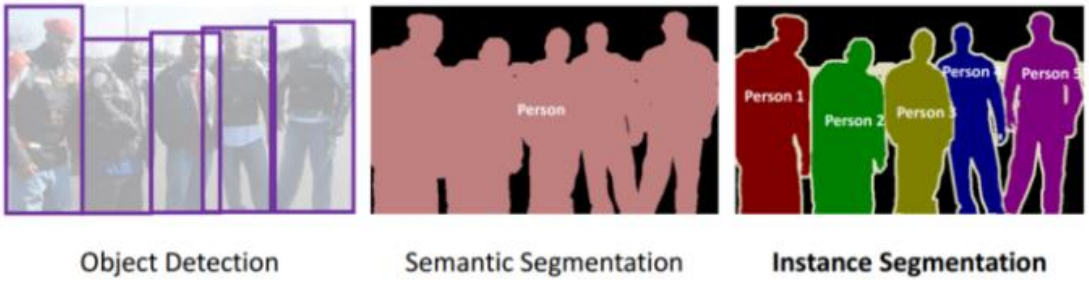


Fuente: Autores.

1.2.3. SEGMENTACIÓN DE INSTANCIAS

Es una combinación entre la detección de objetos y la segmentación semántica, pero la diferencia es que separa los datos que pertenecen a una misma clase.

Fig. *1-6*. Diferencias entre la detección de objetos, la segmentación semántica y la segmentación de instancias.



Fuente: (Liu, 2020)

En este proyecto se llevará a cabo la detección de objetos y la segmentación semántica, pues se requiere poder detectar las zonas de interés mencionadas y además pintarlas para poder conocer a detalle su ubicación y ocupación.

2. GENERACIÓN DE IMÁGENES DE SÚPER RESOLUCIÓN

En este capítulo se utilizan diversos métodos para generar imágenes de súper resolución. Se definen parámetros globales para ser utilizados en cada uno de ellos y se comprueban los tiempos de ejecución e imágenes resultantes de cada uno.

Para realizar el proyecto de forma eficiente, se realizaron tomas aéreas de un cultivo de caña de azúcar en una zona cercana a la ciudad de Cali, Colombia, ya que en este lugar están presenten las zonas de interés mencionadas.

Para obtener las imágenes de súper resolución, se implementaron 3 métodos: Interpolación lineal modificada, ESRGAN y PIX2PIX.

2.1. INTERPOLACION LINEAL MODIFICADA

Este método es una variante del método tradicional de interpolación lineal, la modificación es realizada por los autores.

A continuación, se muestra el seudocódigo correspondiente para implementar la interpolación lineal modificada:

Alg. *2-1.* Seudocódigo de la implementación de la interpolación lineal modificada.

```
Algoritmo Interpolación Lineal Modificada
        // Cargar Librerías.
        // Generar número aleatorio para seleccionar una imagen aleatoria.
        Definir Num_Imagen Como Entero
        Num_Imagen <- Azar()

        // Cargar imagen.
        Definir Ruta Como Entero
        ImgBase = CargarImagen(Concatenar(Ruta, Num_Imagen, Formato))

        // Recortar imagen sobre una zona de interés.
        ImgRecortada = Recortar(ImgBase, Coordenadas)

        // Convertir a escala de grises.
        ImgRecortadaGris = GrayScale(ImagenRecortada)

        // Procesamiento entre la imagen base y cada una de las imágenes
        restantes.
        Repetir
                // Cargar nueva imagen
                ImgBase2 = CargarImagen(Concatenar(Ruta, k, Formato))
                // Recortar imagen sobre una zona de interés.
                ImgRecortada2 = Recortar(ImgBase2, Coordenadas)
                // Convertir a escala de grises.
                ImgRecortadaGris2 = GrayScale(ImgRecortada2)
```

```
// Implementar algoritmo SIFT.
CoordImg1, CoordImg2 = SIFT(ImgRecortadaGris, ImgRecortadaGris2)
// Implementar algoritmo RANSAC para obtener matriz homográfica.
MH = RANSAC(CoordImg1, CoordImg2)
// Crear imagen base en negro del tamaño original por un factor.
BaseAmpliada1, BaseAmpliada2 = ZEROS(ImgRecortadaAncho*Amplitud,
ImgRecortadaAlto*Amplitud, 3)
// Aumentar tamaño imagen original con pixeles negros intermedios
Repetir
	Repetir
		BaseAmpliada1(i,j,:) =
		ImgRecortada(i/Amplitud, j/Amplitud,:)
	Cada Amplitud Hasta Que j = Longitud(BaseAmpliadaAncho)
Cada Amplitud Hasta Que i = Longitud(BaseAmpliadaAlto)
// Aumentar tamaño de la nueva imagen cargada
Repetir
	Repetir
		BaseAmpliada2(i,j,:) =
		ImgRecortada2(i/Amplitud, j/Amplitud,:)
	Cada Amplitud Hasta Que j = Longitud(BaseAmpliada2Ancho)
Cada Amplitud Hasta Que i = Longitud(BaseAmpliada2Alto)
// Hallar todas las coordenadas (X,Y) de la imagen ampliada 1 en
La imagen ampliada 2 por medio de la matriz homográfica.
CoordenadasX, CoordenadasY = HallarCoordenadas(BaseAmpliada, MH)
// Realizar la interpolación lineal entre ambas imágenes.
Repetir
	Repetir
		// Definir vectores que almacenaran las intensidades
		de los pixeles aledaños a cada pixel en negro.
		Si BaseAmpliada1(i,j) = 0
		Definir IntR, IntG, IntB Como Arreglos
		Repetir
			Repetir
			   IntR = [BaseAmpliada2(XX,YY,1)
				   Norma([i,j]-[XX,YY])]
			   IntG = [BaseAmpliada2(XX,YY,2)
				   Norma([i,j]-[XX,YY])]
			   IntB = [BaseAmpliada2(XX,YY,3)
				   Norma([i,j]-[XX,YY])]
			Desde i-Amplitud+1 Hasta Que XX=i+Amplitud-1
		Desde j-Amplitud+1 Hasta Que YY=j+Amplitud-1
		// Calcular la suma ponderada. IR, IG, IB son la
		intensidad final de cada canal.
		Definir IR=0,IG=0,IB=0,OpR=0,OpG=0,OpB=0 Como Real
		SumaPonderada <- Minimo(Suma(IntB))
		Repetir
			OpR = (IntR(t,2)/SumaPonderada)* IntR(t,1)
			OpG = (IntG(t,2)/SumaPonderada)* IntG(t,1)
			OpB = (IntB(t,2)/SumaPonderada)* IntB(t,1)
			IR = IR + OpR
			IG = IG + OpG
			IB = IB + OpB
		Hasta Que t = Longitud(IntB)
		// Aplicar la intensidad correspondiente al espacio
		En negro en la imagen ampliada
		BaseAmpliada1(i,j,1) = IR
		BaseAmpliada1(i,j,2) = IG
		BaseAmpliada1(i,j,3) = IB
	Hasta Que j = Longitud(BaseAmpliadaAlto)
Hasta Que i = Longitud(BaseAmpliadaAncho)
```

```
            // Guardar la imagen generada.
            GuardarImagen(BaseAmpliada1, Nombre=SRParcial)

            // Como se van a ir guardando varias imágenes, la idea es ir
            promediando cada nueva imagen generada.
            Definir PR, PG, PB Como Real
            Repetir
                  Repetir
                        PR = Redondeo(SRParcial(i,j,1)+BaseAmpliada1(i,j,1)/2)
                        SRParcial(i,j,1) = PR
                        PG = Redondeo(SRParcial(i,j,2)+BaseAmpliada1(i,j,2)/2)
                        SRParcial(i,j,2) = PG
                        PB = Redondeo(SRParcial(i,j,3)+BaseAmpliada1(i,j,3)/2)
                        SRParcial(i,j,3) = PB
                  Hasta Que j = Longitud(SRParcialAncho)
            Hasta Que i = Longitud(SRParcialAlto)

            Si k < Cantidad_Imagenes
                  // Guardar la imagen promediada.
                  GuardarImagen(SRParcial, Nombre=SRParcial)
            Si No
                  // Guardar la imagen final.
                  GuardarImagen(SRParcial, Nombre=SRFinal)

      Hasta Que k = Cantidad_Imagenes

      // Finalmente se obtiene una imagen que representa lo mejor de cada una.
      Se aplica el deblur para eliminar lo borroso de la imagen.
      Índice de dispersión de punto
      PSF = FiltroGaussiano(Amplitud*Amplitud)
      // Desconvolución ciega.
      SuperResolucionFinal = DesconvolucionCiega(SRFinal, PSF)

FinAlgoritmo
```

Para generar una imagen de súper resolución se debe tener una imagen base la cual se amplía. Para ello, se obtiene una imagen de referencia de un conjunto de imágenes de la misma escena; como este método hace parte del método MISR, se tomaron 196 imágenes de la misma escena y se seleccionó una de ellas de forma aleatoria.

Fig*. 2-1*. Imagen base tomada en un cultivo de caña de azúcar.

Fuente: Autores.

Esta imagen tiene dimensiones de 3648 x 5472 x 3 pixeles, sin embargo, procesarla toda de una sola vez es un proceso que requiere mucho tiempo y recursos y en muchos casos no es necesario tener información detallada de toda la imagen, solo de una sección específica, es por ello que se recorta la imagen en una sección de interés de tamaño 912 x 1513 x 3 pixeles.

Fig. *2-2.* Sección de interés en la imagen original.

Fuente: Autores.

Tras definir la imagen base, se van a ir cargando una a una las siguientes imágenes que fueron tomadas. Este proceso es similar que el anterior, la única diferencia es que ahora se cargará otra imagen, pero recortada exactamente en la misma posición que la imagen anterior, esto va a generar que la imagen tenga algún movimiento, rotación, desplazamiento o inclinación.

2.1.1. SIFT (SCALE INVARIANT FEATURE TRANSFORM)

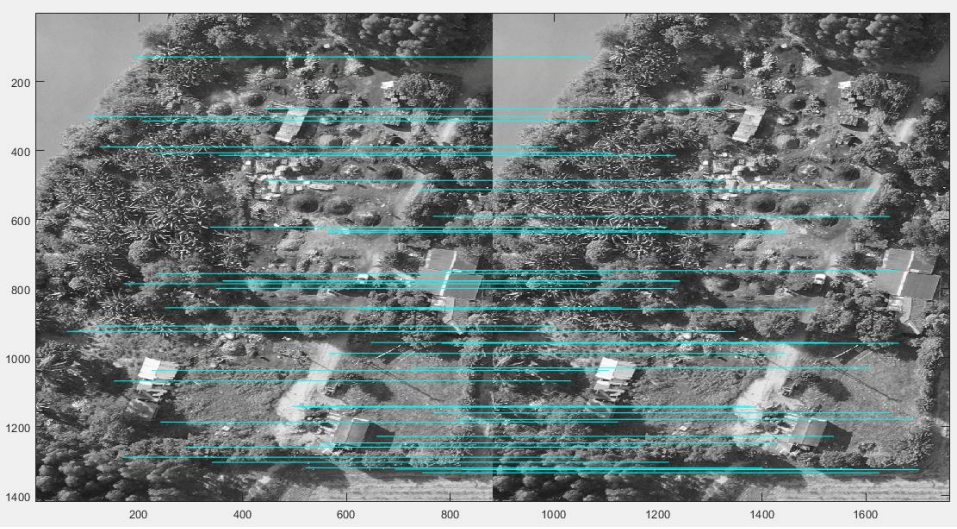
Es un algoritmo utilizado para extraer características relevantes de las imágenes. Se da a la tarea de localizar en una imagen puntos tanto relevantes en cuanto a la cantidad de información de su entorno como estables frente a las perturbaciones y transformaciones locales y globales que puede sufrir la imagen. Esto último garantiza que dichos puntos se puedan detectar de forma confiable y con mucha exactitud al realizar esta tarea de forma repetitiva. Por tal motivo, se pueden hallar las coordenadas de esos puntos en las dos imágenes (Lowe, 2004).

2.1.2. RANSAC (RANDOM SAMPLE CONSENSUS)

Es un método iterativo para calcular los parámetros de un modelo matemático de un conjunto de datos observados que contiene valores atípicos (MathWorks, 2019). El objetivo es poder hallar lo que se conoce como la matriz homográfica la cual permite capturar el cambio de perspectiva de un objeto en 3D que sufre un movimiento en el espacio a partir de sus proyecciones sobre un plano de imagen.

El algoritmo SIFT va a comparar las dos imágenes y va a encontrar los emparejamientos entre ambas imágenes. Como resultado de esto, se van a devolver las coordenadas que hicieron “match” junto con la cantidad de emparejamientos entre las dos imágenes.
Para implementar SIFT las imágenes deben estar en escala de grises.

Fig. *2-3*. Algunos emparejamientos entre ambas imágenes.



Fuente: Autores.

Con las coordenadas obtenidas se implementa el algoritmo RANSAC para obtener la matriz homográfica.

Con el fin de garantizar una matriz con los valores más cercanos a los esperados, se decide dejar el número de puntos a tomar en 1000, ya que, en este tipo de

imágenes, la cantidad de características puntuales van más allá de 15000 puntos, de tal forma que este algoritmo al tener que revisar 1000 pares de coordenadas, garantiza una buena matriz homográfica.

A continuación, se presenta la primera matriz homográfica obtenida. El resto posee las mismas dimensiones y sus valores varían ligeramente, puesto que esta matriz se calcula con cada nueva imagen, por lo que cada imagen posee ligeras variaciones posicionales y rotacionales.

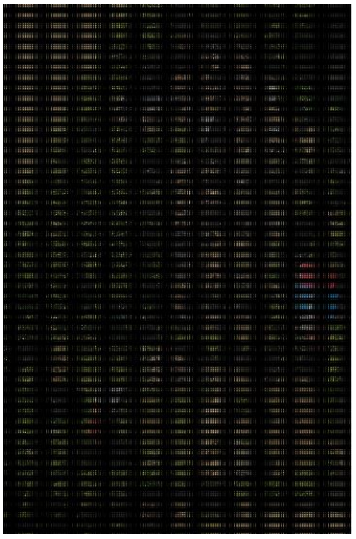
$$MH = \begin{bmatrix} 1.0001 & 0.0002 & -0.3074 \\ 0.0001 & 1.0001 & -0.1794 \\ 0 & 0 & 1 \end{bmatrix} \quad (2\text{-}1)$$

Al observar la matriz obtenida es claro que los valores son correctos, ya que los valores cercanos a 0 y 1 indican que las traslaciones y rotaciones son prácticamente nulas, lo cual es cierto, pues las imágenes fueron tomadas en un lapso de 100 ms, por lo que estas transformaciones son mínimas.

2.1.3. AUMENTO DE TAMAÑO DE LA IMAGEN BASE

El próximo paso a seguir es determinar el factor de amplitud, es decir, definir cuantas veces se quiere aumentar la imagen, pues esto va a involucrar obtener una imagen con mayor resolución, pero también va a requerir de un costo computacional mucho mayor. Para evitar que el programa pueda colapsar la memoria RAM y CPU del computador, se decide ampliar 3 veces la imagen de referencia y la imagen de comparación. Para ello, se van agregando 2 pixeles negros entre cada pixel que posee información, obteniendo lo siguiente:

Fig. *2-4*. Imagen base ampliada 3 veces con pixeles negros intermedios.

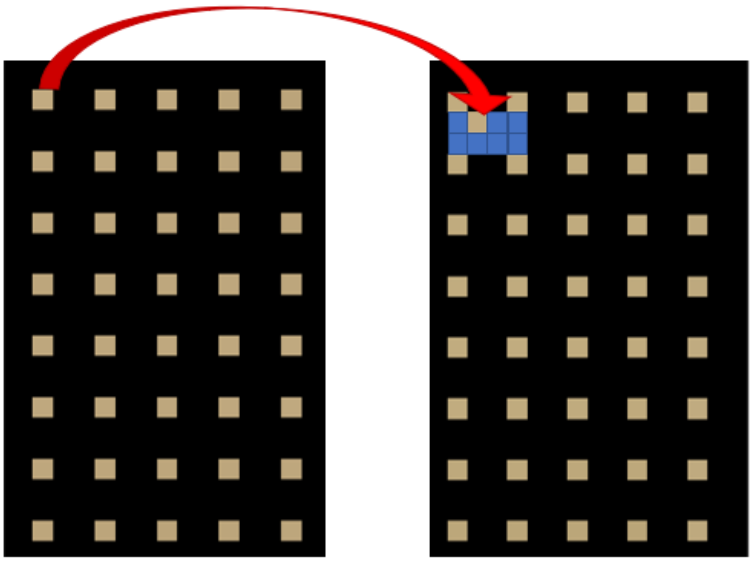


Fuente: Autores.

Al obtener la imagen ampliada se procede a multiplicar pixel a pixel la imagen base ampliada por la matriz homográfica correspondiente, recordando que la matriz sirve para este caso, puesto que tanto la imagen base como la de

comparación fueron ampliadas con el mismo factor. De esta forma se obtienen las coordenadas en (x, y) de la imagen 1 en la imagen 2. Sin embargo, este proceso puede resultar en valores decimales lo cual haría imposible obtener datos de la segunda imagen ya que no existen pixeles con índice decimal, este debe ser un valor entero, para ello se redondea el valor de las coordenadas obtenidas en la imagen 2. Estas coordenadas se dan en forma matricial.

Fig. *2-5*. Multiplicación de la imagen 1 por la matriz homográfica y obtención de la coordenada redondeada.



Fuente: Autores.

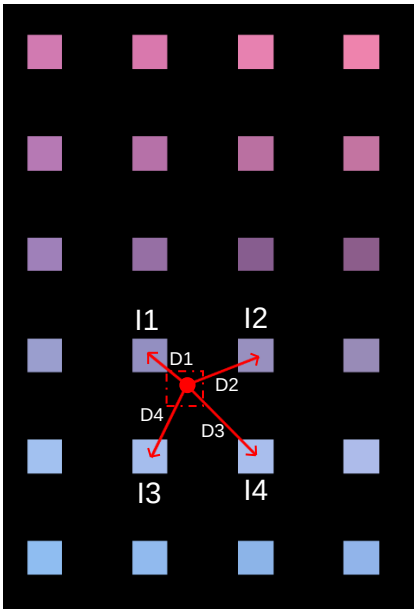
2.1.4. INTERPOLACIÓN LINEAL

En este punto ya se obtienen los pares de pixeles correspondientes en ambas imágenes, por lo que ya es posible realizar la interpolación lineal. Este paso consiste en rellenar con información cada uno de los pixeles en negro que se añadieron a la foto base. Para definir con qué información se va a llenar es necesario observar la posición en la que un determinado pixel se encuentra en las dos imágenes. Luego, alrededor de ese pixel se crea una matriz vecindad la cual es de tamaño 5x5 ya que es 2 pixeles por encima, uno inferior a la derecha y a la izquierda del pixel que se está analizando en la segunda imagen.

En este punto se presentan varios escenarios, el primero de ellos es cuando los pixeles de la segunda imagen estén en los extremos de la imagen, aquí la matriz vecindario se reduce del lado extremo en que se encuentra ese pixel.

Para obtener el valor de pixel negro de la primera imagen se observa cuantos pixeles están en la matriz vecindad de la segunda imagen, y según la cantidad de pixeles se normaliza la distancia a la que está ese pixel con respecto a la coordenada inicial del pixel. Luego, este valor se multiplica por la intensidad según el canal que se está analizando, de esta forma se tiene un valor promedio del pixel que va acorde a la suma ponderada de los pixeles vecinos y sus distancias correspondientes.

Fig. *2-6*. Representación de la interpolación lineal.



Fuente: Autores.

Con la siguiente ecuación se puede entender mejor el concepto:

$$IP(x,y) = \sum_{n=1}^{i} \frac{|(x,y)-(xi,yi)|}{\sum_{n=1}^{i}|(x,y)-(xi,yi)|} \cdot IC \qquad (2\text{-}2)$$

Donde IP es la intensidad del pixel por canal.

La sumatoria que va de 1 hasta i es según la cantidad de pixeles que están dentro de la matriz vecindad.

IC es la intensidad de cada pixel con información dentro de la matriz, por canal.

Esta operación se hace en todos los pixeles y en cada canal, es por este motivo que este método toma tanto tiempo puesto que se tienen que barrer todos los pixeles de la segunda imagen y en cada uno de ellos realizar la operación de la intensidad por canal.

Al realizar esto se obtiene una imagen de tamaño aumentada con información, así:

Fig*. 2-7.* Zona de interés aumentada 3 veces.

Fuente: Autores.

2.1.5. UNION DE IMÁGENES DE SÚPER RESOLUCIÓN

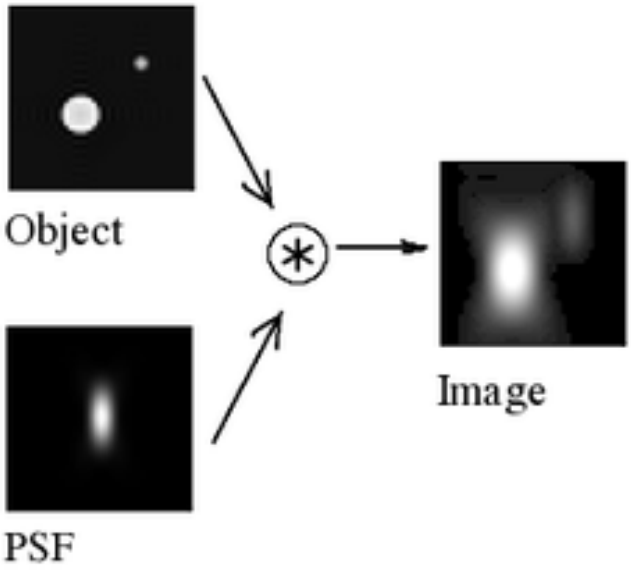
Como todo este procedimiento se realiza con cada imagen tomada, el objetivo es tener una imagen que irá almacenando cada nueva imagen que se va generando, pero el método para almacenarla es ir promediando la foto de súper resolución que había anteriormente con la nueva que se generó, de tal forma que al final lo que se hace es promediar cada pixel en cada canal con todas las imágenes de súper resolución obteniendo una imagen que va a representar lo mejor de todas las imágenes.

2.1.6. DESCONVOLUCIÓN CIEGA

A pesar de que, en este punto, ya se tiene una imagen de súper resolución, este procedimiento hace que ciertas zonas tomen patrones extraños o existan pixeles con intensidades que no corresponden con sus vecinos, incluso se presenta cierta borrosidad en algunos sectores, por ello, se implementa un método llamado desconvolución ciega, el cual es un proceso que elimina la distorsión de una imagen borrosa, utilizando el conocimiento de cómo el sistema óptico desenfoca un único punto de luz.

El operador de distorsión, también llamado PSF, describe el grado en que un sistema óptico desenfoca un punto de luz. El PSF es la transformación inversa de Fourier de la función de transferencia óptica. Para este proceso se hace una estimación inicial y el resultado es una estimación generada por la función.

Fig. *2-8*. Función de dispersión de punto.



Fuente: (Shaw, 1997).

Este proceso se realiza con una matriz predeterminada que proviene a partir de un filtro gaussiano de 5x5 (Del mismo tamaño de la matriz vecindad), el más común en este tipo de imágenes, pero sus valores exactos son muy difíciles de predecir pues proviene de múltiples imágenes que han sido procesadas.

La matriz estimada es de 5x5 con los siguientes valores:

$$PSF = \begin{bmatrix} 0.0075 & 0.0081 & 0.0091 & 0.0079 & 0.0072 \\ 0.0069 & 0.0075 & 0.0083 & 0.0072 & 0.0065 \\ 0.0058 & 0.0062 & 0.0135 & 0.0059 & 0.0053 \\ 0.0041 & 0.0044 & 0.0049 & 0.0042 & 0.0038 \\ 0.0031 & 0.0033 & 0.0036 & 0.0031 & 0.0028 \end{bmatrix} \quad (2\text{-}3)$$

Al observar los valores, se ve como estos disminuyen en los extremos y crecen en el centro, lo cual indica que el PSF estimado también posee forma gaussiana.

La imagen resultante al aplicar la desconvolución es la siguiente:

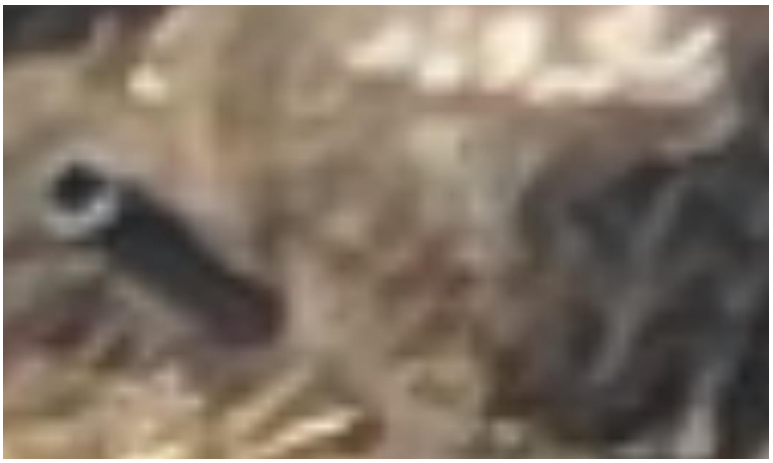
Fig. *2-9.* Imagen de súper resolución de la zona de interés.

Fuente: Autores.

2.1.7. RESULTADOS

La imagen obtenida es de 3219 x 4611 x 3 pixeles, es decir, 3 veces mayor que la imagen original recortada. A simple vista, la imagen parece tener mayor definición, sin embargo, para comprobar que efectivamente todo este procedimiento realizado, además de aumentar el tamaño de la imagen, permita ver con mayor claridad una sección específica, se decide hacer zoom sobre una zona que parece ser una tubería que proviene del suelo y se compara esa misma zona con la imagen original, por lo que se obtiene lo siguiente:

Fig. *2-10.* Sección aumentada de la imagen original.



Fuente: Autores.

Fig. *2-11.* Sección aumentada de la imagen de súper resolución.

Fuente: Autores.

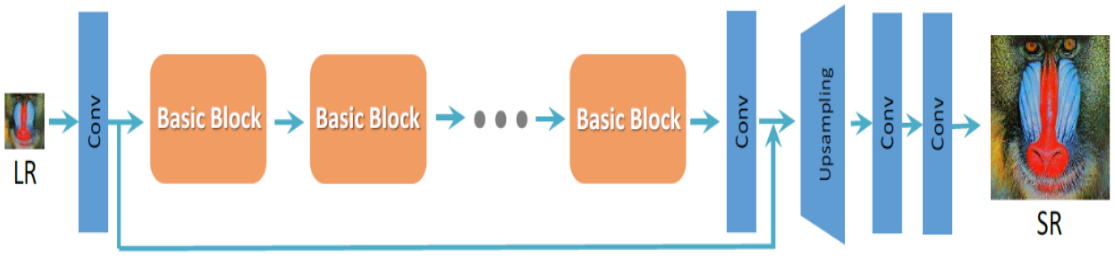
El programa tardó 4 horas y 34 minutos procesando 56 imágenes.

Como se puede ver, la imagen de súper resolución permite ver las texturas de los objetos con más detalle, a pesar de que aún se ve algo borroso, la mejora es clara en comparación a la imagen original. En el capítulo de resultados se analizará con mayor detenimiento la imagen obtenida.

2.2. ESRGAN (Enhanced Super-Resolution Generative Adversarial Networks)

Es una arquitectura del tipo GAN (Red Generativa Adversaria). Este tipo de arquitectura se compone de 2 redes neuronales convolucionales que resuelven 2 tareas diferentes. Una red generadora del tipo U-Net la cual se encarga de generar o producir el contenido que se desea. Está conformada por 16 bloques básicos que se componen de una capa convolucional y una capa de Batch Normalization. Por otro lado, se encuentra la red discriminadora conformada por 5 bloques que se componen de una capa convolucional y una capa de activación ReLU. Esta red sirve para saber si la imagen de salida es real o falsa, es decir, si se tomó una imagen propia del conjunto de imágenes o es generada a partir de la red generadora. Estas redes están en oposición. Una siempre busca engañar a la otra y de esta manera ambas redes van mejorando con el paso del tiempo.

Fig. *2-12.* Arquitectura ESRGAN.



Fuente: (Sinha, 2019).

2.2.1. CARACTERISTICAS E IMPLEMENTACIÓN

La particularidad de esta red es que, en primer lugar, se elimina la normalización de lotes (Batch Normalization). La eliminación de algunas capas BN ha demostrado aumentar el rendimiento y reducir la complejidad computacional en diferentes tareas orientadas a la súper resolución.

En segundo lugar, el bloque denso, originado por la red DenseNet, se usa para reemplazar el bloque residual para mejorar la red.

Estos cambios se hacen con el fin de mejorar otras redes existentes, pues estas redes tienden a generar resultados borrosos.

Los métodos anteriores basados en GAN, tienden a producir texturas que no son naturales y contienen ruido desagradable. ESRGAN se deshace de estos elementos y produce resultados naturales (Wang & Yu, 2018).

Este programa se encuentra disponible en el repositorio de GITHUB, sin embargo, para este caso específico se realizaron ciertas modificaciones para ser utilizado de forma correcta.

A continuación, se muestra el seudocódigo correspondiente para implementar la red ESRGAN:

Alg. *2-2.* Seudocódigo de la implementación de la red ESRGAN.

```
Algoritmo RedESRGAN
       // Cargar Librerías y red pre-entrenada.
       Leer IMAGEN_BAJA_RESOLUCION

       // Recortar imagen original en 8 secciones iguales.
       Definir AJUSTE_ALTURA,AJUSTE_ANCHO, SECCION_GUARDAR Como Real
       AJUSTE_ALTURA <- ALTO_IMAGEN_BAJA_RESOLUCION/4
       AJUSTE_ANCHO <- ANCHO_IMAGEN_BAJA_RESOLUCION/2
       Repetir
              Repetir
                     SECCION(i,j) = [(i-1)*AJUSTE_ALTURA:(i)*AJUSTE_ALTURA,
                     (j-1)*AJUSTE_ANCHO:(j-1)*AJUSTE_ANCHO
                     +(j)*AJUSTE_ANCHO,:]
                     GUARDAR_SECCION(SECCION(i,j))
              Hasta Que j = 2
       Hasta Que i = 4
```

```
        // Cargar el modelo pre-entrenado y aplicarlos sobre cada una de
        las 8 secciones.
        Leer MODELO_ESRGAN
        MODELO <- MODELO_ESRGAN

        Repetir
                Leer SECCION_i
                IMAGEN_FLOTANTE <- SECCION/255
                IMAGEN_SUPER_RESOLUCION = MODELO(IMAGEN_FLOTANTE)
                IMAGEN_SUPER_RESOLUCION_FINAL(i) =
                REDONDEO(IMAGEN_SUPER_RESOLUCION * 255)
                Escribir GUARDAR_SECCION(IMAGEN_SUPER_RESOLUCION_FINAL(i))
        Hasta Que i = 8

FinAlgoritmo
```

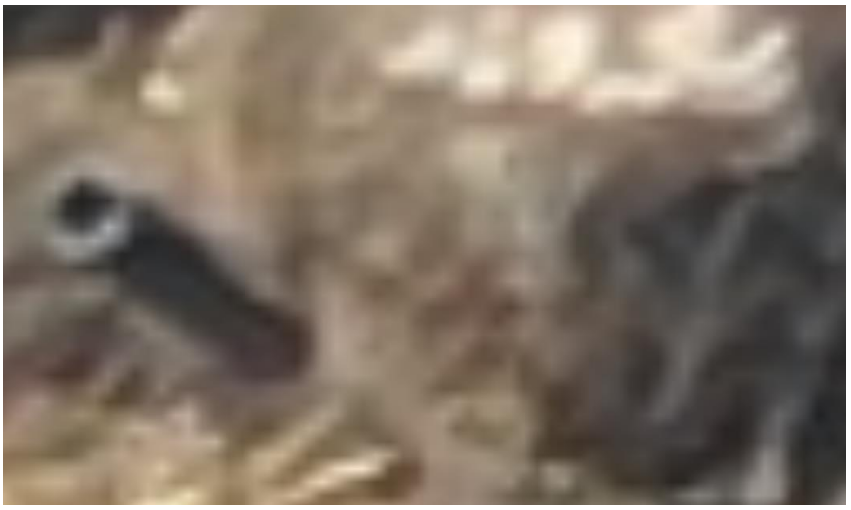
En primer lugar, si bien ya la red se encuentra pre-entrenada, es necesario reentrenar con algunas imágenes de zonas naturales, ya que en estos escenarios se presentan grandes cambios en las texturas y en las intensidades de los pixeles y es necesario que la red aprenda a identificar estos cambios.

En segundo lugar, se modifica el tamaño de las imágenes de entrada puesto que la red permite imágenes de máximo 1000 x 1000 pixeles, pero al hacer esto se pierde resolución, por lo que se decide partir la imagen en 8 secciones iguales de la imagen base recortada (ver figura 2-2) y a cada una de ellas implementar el algoritmo de súper resolución, luego se unen todas ellas lo cual es similar a como si se procesara toda la imagen de completa.

Posterior a ello, se carga el nuevo modelo pre-entrenado y se implementa sobre cada una de las 8 imágenes, luego se unen y se obtiene el siguiente resultado.

2.2.2. RESULTADOS

Fig*. 2-13.* Sección aumentada de la imagen original.



Fuente: Autores.

Fig*. 2-14.* Sección aumentada de la imagen de súper resolución.

Fuente: Autores.

El reentrenamiento tardó 1 hora y 16 minutos. La generación de la imagen tardó 2 minutos y 28 segundos.

La imagen resultante es de 3648 x 6052 pixeles, es decir, 4 veces más grande que la imagen base recortada original. Como se puede ver, si bien la claridad no aumenta mucho, hay secciones qué sí presentan texturas más definidas en comparación a la original, lo cual permite observar un poco mejor los bordes de cada elemento presente en esa escena, por lo que, efectivamente, se pudo obtener una imagen de súper resolución que permite ver con mejor claridad algunos objetos.

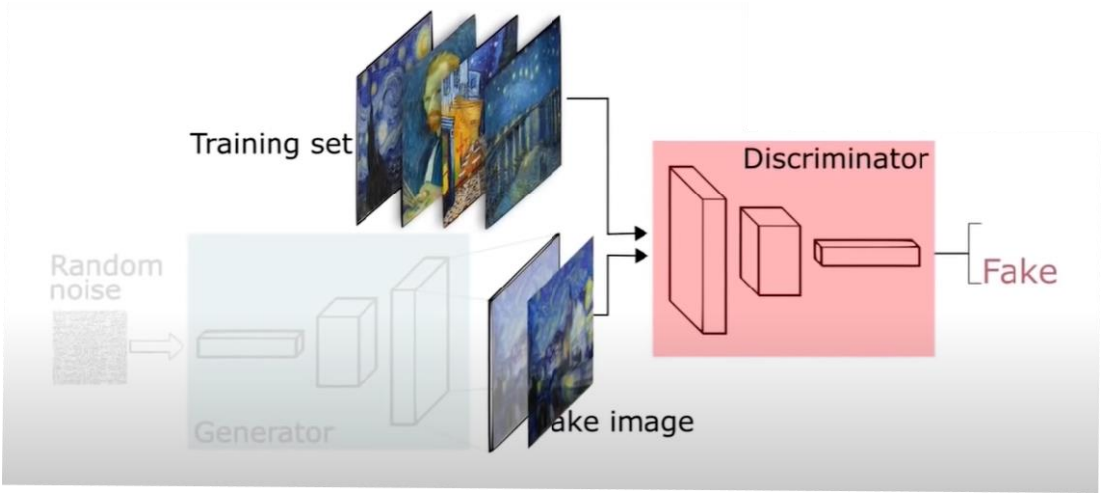
2.3. PIX2PIX

Al igual que en ESRGAN, esta también es una arquitectura basada en GAN, la diferencia es que es condicionada, lo cual se conoce como cGAN.

Esta arquitectura significa “picture to picture” es decir, imagen a imagen. Lo que significa que esta red se va a encargar de crear imágenes, no modificar las que ya existen (Isola, 2018).

La arquitectura se encarga de traducir imágenes de un dominio a otro dominio.

Fig*. 2-15.* Arquitectura PIX2PIX.



Fuente: (Santana, 2019).

2.3.1. ARQUITECTURAS GANs

Son dos redes neuronales conectadas. Una red convolucional que va a extraer las características de la imagen y una red deconvolucional que va a realizar la reconstrucción de la nueva imagen. Esta arquitectura se conoce como reloj de arena. En la primera parte de la red se encuentra el Encoder que se encarga de extraer a detalle todas las características de la imagen. La segunda parte es el decoder o decodificador de la información, el cual, a partir de la extracción de características de la imagen de entrada, las asocia con la imagen de salida, por lo que esta sección es la que genera las imágenes con sus respectivas intensidades por pixel.

Existen muchos tipos de arquitectura del tipo reloj de arena, sin embargo, una de las más comunes es la U-Net.

2.3.1.1. U-NET

U-net se inventó originalmente y se usó por primera vez para la segmentación de imágenes biomédicas. Su arquitectura puede considerarse ampliamente como una red de codificador seguida de una red de decodificador. A diferencia de la clasificación donde el resultado final de la red profunda es lo único

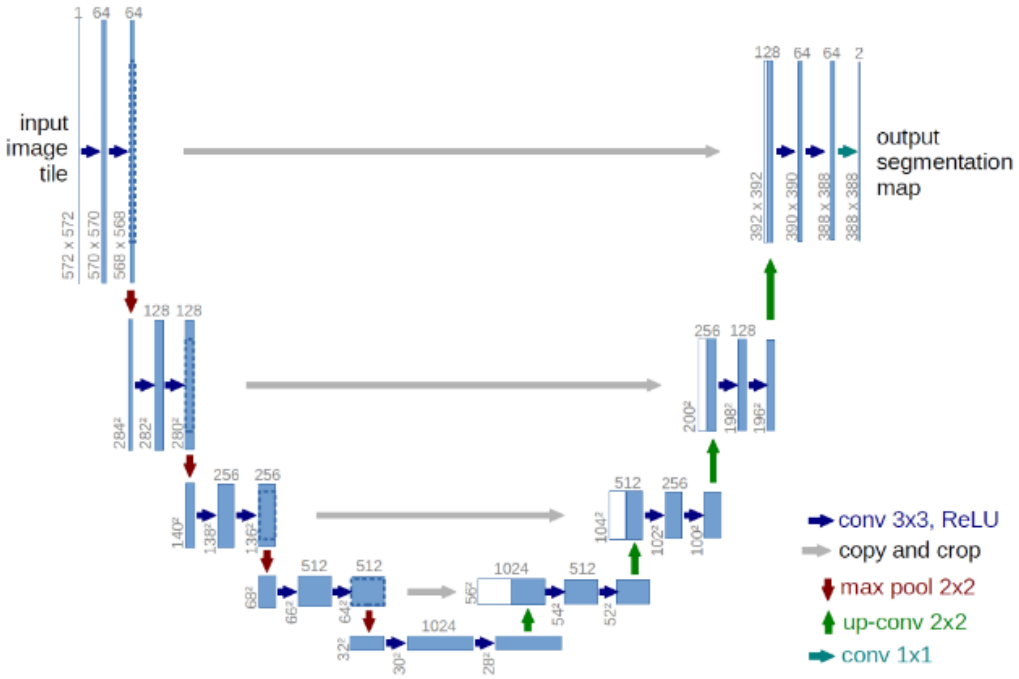
importante, la segmentación semántica no solo requiere discriminación a nivel de píxel sino también un mecanismo para proyectar las características discriminatorias aprendidas en diferentes etapas del codificador en el espacio de píxel (ArcGIS, 2019).

Una modificación importante en U-Net es que hay una gran cantidad de canales de características en la parte de muestreo ascendente, que permiten que la red propague información de contexto a capas de mayor resolución. Como consecuencia, el camino del decoder es más o menos simétrico a la parte del encoder, y produce una arquitectura en forma de U.

Algo que caracteriza la U-Net son los saltos de conexión. Si la información se comprime mucho se pueden hacer saltos entre las capas de la red, es decir, no llegar a la capa más profunda para evitar los cuellos de botella.

Para generar un contenido especifico, se debe ingresar la clase a la que pertenece, es decir, la etiqueta del vector de entrada condicionándola a la forma de la salida. Por lo tanto, la U-Net es un modelo en el que la imagen de entrada intenta asemejarse a la imagen de salida copiando sus características puntuales.

Fig. *2-16.* Arquitectura U-Net.



Fuente: (Ronneberger, 2015).

Sin embargo, para el caso puntual de la generación de imágenes de súper resolución, se modificó la arquitectura U-Net agregando 3 capas más, ya que la red original produce imágenes de 256 x 256 x 3 pixeles, pero este tamaño es inferior que las imágenes de entrada, por lo que no sería súper resolución. Al agregar 3 capas más, las imágenes de salida son de 2048 x 2048 x 3 que sí son de mayor resolución que las imágenes de entrada.

La arquitectura modificada se puede ver en el anexo 1 y 2.

Para realizar el entrenamiento de este modelo, se tomaron 196 imágenes, todas muestran la misma escena (Ver figura 2-1).

A continuación, se lista el procedimiento para obtener las imágenes de súper resolución a partir del modelo PIX2PIX junto con el seudocódigo correspondiente.

Alg. *2-3.* Seudocódigo de la implementación de la red PIX2PIX para súper resolución.

```
Algoritmo PIX2PIXSR

	// Importar librerías de TensorFlow.

	// Estas variables contendrán la ruta de las imágenes de entrada,
	salida y la ruta de los checkpoints correspondientes.
	Definir INPATH, OUTPATH, CHPATH Como Cadena

	// Cantidad de imágenes para entrenar.
	NUM_IMAGENES <- Longitud(INPATH)
	// Se utiliza el 80% de las imágenes para entrenamiento.
	CAN_ENTRENAMIENTO <- NUM_IMAGENES*0.8
	// Separar conjunto de entrenamiento y de prueba.
	ENTRENAMIENTO <- INPATH(0:CAN_ENTRENAMIENTO)
	PRUEBA <- INPATH(CAN_ENTRENAMIENTO:NUM_IMAGENES)
	target = Nombres(ENTRENAMIENTO)
	// Especificar tamaño de salida de las imágenes.
	IMG_WIDTH = 2048
	IMG_HEIGHT = 2048

	// Función para aumentar los datos. Aplicar transformaciones
	aleatorias para generar más imágenes. Las transformaciones son
	desplazamientos y rotaciones aleatorias
	CONJUNTO_ENTRENAMIENTO, CONJUNTO_PRUEBA =
	CARGAR_IMAGEN(RUTA_ENTRENAMIENTO, RUTA_PRUEBA)
		// Ajustar al tamaño especificado y aplicar rotaciones.
		CONJUNTO_ENTRENAMIENTO,CONJUNTO_PRUEBA =
		RANDOM_JITTER(RUTA_ENTRENAMIENTO, RUTA_PRUEBA)

	// Distribuir en lotes los datos de entrenamiento y prueba.
	CONJUNTO_ENTRENAMIENTO = BATCHES(CONJUNTO_ENTRENAMIENTO)
	CONJUNTO_PRUEBA = BATCHES(CONJUNTO_PRUEBA)

	// Construir la arquitectura.
	// Encoder.
	encoder(filtros)
		// Capa convolucional.
		Conv2D(filtros, padding, inicializador)
		// Capa de normalización.
		BatchNormalization()
		// Capa de activación.
		LeakyReLU()
```

```
        // Decoder.
        decoder(filtros)
                // Capa deconvolucional.
                Conv2DTranspose(filtros, padding, inicializador)
                // Capa de normalización.
                BatchNormalization()
                // Regularizador de la red.
                Dropout()
                // Capa de activación.
                LeakyReLU()

        // Red Generadora.
        Generador <- generator()
                // Son 11 bloques que conforman el encoder.
                downsample = [
                        encoder(64)
                        .
                        .
                        .
                        encoder(512)]

                // Son 11 bloques que conforman el encoder.
                upsample = [
                        decoder(512)
                        .
                        .
                        .
                        decoder(64)]

                // Conectar todo y realizar las skip connections de la U-Net.
                Definir Estructura Como Arreglo
                Repetir
                        Estructura <- AGREGAR(downsample(i))
                Hasta Que i = 11
                Repetir
                        Estructura <- AGREGAR(upsample(i))
                        Concatenar(Estructura(12-i),Estructura(11+i))
                Hasta Que i = 11
                // Construir el modelo.
                Generador <- MODELO(ESTRUCTURA)

        // Red discriminadora.
        Discriminador <- discriminator()
                // Son 4 bloques. Únicamente se tiene el encoder.
                Definir Down Como Arreglo
                Down = [
                        encoder(64)
                        .
                        .
                        .
                        encoder(512)]

                Discriminador <- MODELO(DOWN)

        // Funciones de coste del generador y del discriminador.
        perdida_generador(target, img_generada)
                // Error medio absoluto entre la imagen destino y la imagen
generada.
                perdida_l1 = media(abs(target, img_generada))
```

```
      perdida_discriminador(target, img_generada, eval_generador)
            // Diferencia entre los pixeles que la red considera falsos y
verdaderos.
            perdida_total = diff(target<>img_generada, eval_generador)

      // Parámetros de optimización. Se utiliza el optimizador Adam.
      optimizador_generador = optimizar(Adam, 0.00004, 0.5)
      optimizador_discriminador = optimizar(Adam, 0.00004, 0.5)

      // Establecer ruta de guardado de checkpoints.
  o   GuardarCheckpoint = Checkpoint(CHPATH, generador, discriminador,
      optimizador_generador, optimizador_discriminador)

      // Definir el paso de entrenamiento.
      paso_entrenamiento(img_generada, target, Generador, Discriminador)

            // Evaluar la imagen generada.
            eval_gen = discriminador(target, img_generada)

            // Llamar a las funciones de coste.
            gen_loss = perdida_generador(target, img_generada)
            disc_loss = perdida_discriminador(target, img_generada,
            eval_generador)

            // Calcular los gradientes para realizar la optimización.
            Backpropagation.
            gen_grad = gradiente(gen_loss)
            disc_grad = gradiente(disc_loss)

            // Optimizar.
            AplicarGradiente(optimizador_generador, Generador)
            AplicarGradiente(optimizador_discriminador, Discriminador)

      // Entrenar.
      Entrenar(CONJUNTO_ENTRENAMIENTO, Épocas)
            Repetir
                  Repetir
                        img_generada = EvaluarModelo(Generador,
                        CONJUNTO_ENTRENAMIENTO 𝑖)
                        paso_entrenamiento(img_generada, target 𝑖)
                  Hasta Que i = Longitud(CONJUNTO_ENTRENAMIENTO)

                  // Guardar un Checkpoint cada 10 épocas.
                  Si (Épocas + 1) MOD 10 = 0 Entonces
                        GuardarCheckpoint()
                  Si No
                        Pasar
                  Fin Si
            Hasta Que Épocas = 400

FinAlgoritmo
```

Ahora, se explicará con un poco mas de detalle en que consiste el seudocódigo mostrado anteriormente.

2.3.2. OBTENCION DEL CONJUNTO DE DATOS DE ENTRADA

En primer lugar, se hizo un reescalado de la imagen original de 5472 x 3648 pixeles, a uno de 1094 x 729 pixeles, es decir, 5 veces de menor tamaño que la imagen original, pues la idea es que esta imagen reducida (entrada) se asemeje a la imagen original (salida). Al intentar asemejarse, va a copiar los patrones que posee la imagen de mayor resolución, por lo que esta red va a generar imágenes de súper resolución relacionando una imagen de baja resolución con una de alta resolución.

Fig*. 2-17.* Comparación entre imagen reducida (izquierda) y original (derecha).

Fuente: Autores.

2.3.3. SEPARACION DEL CONJUNTO DE ENTRENAMIENTO Y DE PRUEBA

Para cada imagen en baja resolución existe su par en alta resolución. Por ello, existen 196 imágenes reducidas, sin embargo, a la hora de hacer el entrenamiento, este no se hace con el 100% de las imágenes, sino que se hace con el 80%, por ello se toman 157 imágenes para el entrenamiento y 39 para realizar las pruebas.

2.3.4. AUMENTO Y LLAMADO DE DATOS

Uno de los mayores problemas en deep learning es que se requiere una gran cantidad de datos (imágenes, en este caso) para realizar un entrenamiento correcto. Para enfrentar este problema se implementa una técnica conocida como “Data Augmentation” o aumento de datos.

Existen múltiples maneras de realizar aumento de datos, pero una que se ha vuelto muy popular últimamente ha sido la de ampliar la imagen de entrada unos cuantos pixeles en cada extremo y luego hacer un recorte de esa imagen ampliada con las dimensiones de la entrada de la red, moviéndose en ese espacio. También se aplica el modo espejo para rotar la imagen.

Por último, se llama una por una cada imagen, se aplica el aumento de datos y se distribuyen en diferentes lotes.

2.3.5. DEFINICIÓN DE LA ARQUITECTURA

En este punto se definen todas las capas de ambas redes, tanto la generadora como la discriminadora. En el caso de la generadora, se aplican las capas adicionales y las conexiones de salto. Los bloques que conforman el encoder se componen de una capa convolucional, una capa de Batch Normalization y una capa de activación ReLU. En el decoder existen los mismos bloques, pero solo los 3 primeros poseen una capa adicional que se encarga de regularizar la red.

El discriminador solo posee el encoder de 4 bloques cada uno con las capas mencionadas anteriormente.

Luego de definir cada modelo, se implementan las funciones de coste de cada red y los parámetros de optimización. La optimización se hace por medio de backpropagation, en el que se calculan los gradientes para optimizarse, es decir, para actualizar los valores de los parámetros.

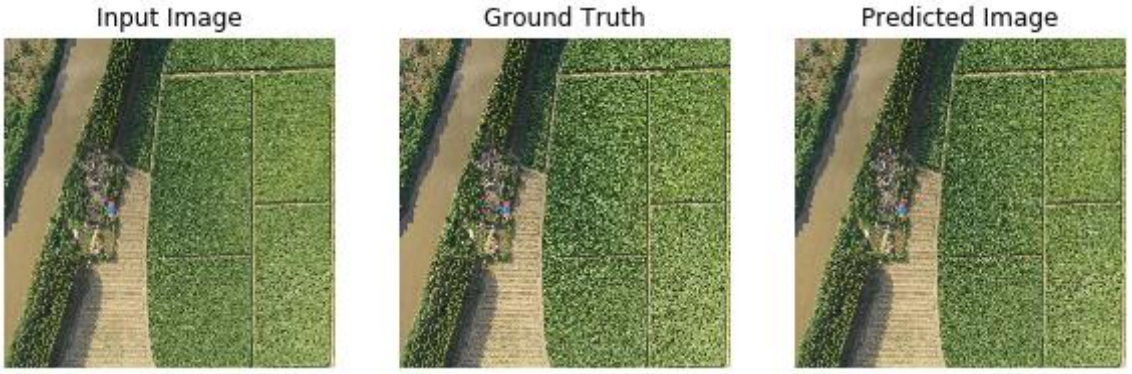
2.3.6. ENTRENAMIENTO

Se define la cantidad de épocas que va a tomar la red, en cada época se procesan las 157 imágenes de entradas con sus respectivas variaciones. Al final de cada época se imprime la imagen de entrada, la imagen de salida y la imagen generada, además del tiempo de ejecución por época, la precisión y la pérdida. Finalmente, se definen puntos de guardado los cuales van a almacenar los valores de los parámetros cada cierta época, puesto que como este procedimiento toma bastante tiempo, puede ocurrir cualquier incidente durante el entrenamiento, por lo que se prefiere prevenir cualquier inconveniente.

Para este caso específico se tomaron 400 épocas, sin embargo, para evitar el overfitting, se determina que cuando la pérdida no varíe más de 0.01% durante 5 épocas consecutivas, el programa se detenga.

Durante el entrenamiento se observan los siguientes resultados.

Fig. *2-18.* Imagen de entrada, imagen de salida e imagen generada durante el entrenamiento con sus respectivos tamaños.



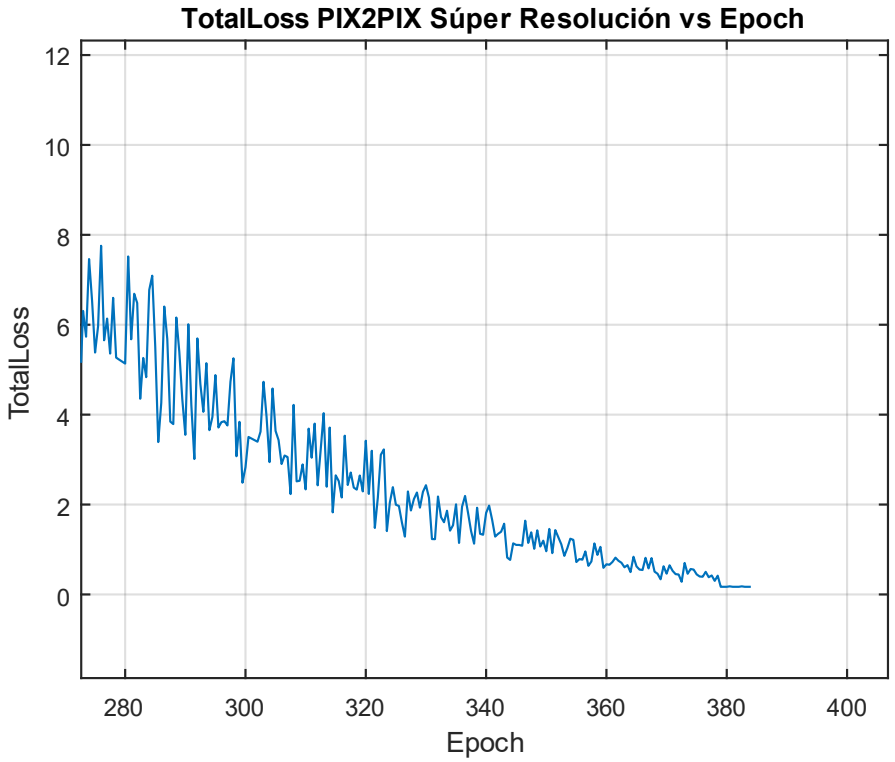
1094 x 729 Px 2048 x 2048 Px 5472 x 3648 Px

Fuente: Autores.

La red se entrenó por 384 épocas ya que en este punto hubo 5 épocas consecutivas donde no existió variación de más del 0,01%, alcanzando un error mínimo de 0.38%.

La red tardó 416 segundos por época, para un total de 47 horas, 18 minutos y 24 segundos para 384 épocas.

Fig. *2-19.* Pérdida total en las últimas épocas durante el aprendizaje de la red PIX2PIX para súper resolución.



Fuente: Autores.

La forma en la que el error disminuye con el paso de las épocas es claramente exponencial donde la misma tiende hacia 0.35%. En las últimas épocas se estabilizo el error y los cambios ya no eran bruscos comparados a las épocas anteriores.

2.3.7. RESULTADOS

A simple vista no se pueden apreciar las diferencias, es por ello que se va a realizar un recorte sobre la zona de interés antes trabajada (Ver figura 2-9) en cada una de las tres imágenes.

Fig*. 2-20.* Imagen de entrada, imagen de salida e imagen generada ampliadas sobre la zona de interés.

Fuente: Autores.

Al observar los resultados obtenidos, se puede determinar que la red mejoró la imagen de entrada, ya que al aumentar la resolución se elimina lo pixelado que se ve la imagen, esto permite que las formas de los objetos se vean con mayor claridad a pesar de que la imagen generada se ve borrosa, pero más definida que la imagen de entrada. Por lo tanto, se puede determinar que hay una mejora a medida que se amplía la imagen.

3. ZONIFICACIÓN AUTOMÁTICA DE IMÁGENES

En este capítulo se utilizan dos métodos para localizar las zonas de interés antes mencionadas. Uno de ellos es la detección de objetos y el otro la segmentación semántica.

3.1. OBTENCIÓN DE IMÁGENES DE ENTRADA

En primer lugar, fue necesario conseguir las imágenes que se utilizarán para el entrenamiento. Para esto se hizo necesario consultar varias bases de datos de imágenes, pero estas imágenes se caracterizan por ser tomadas de forma aérea ya sea por un dron o un satélite y porqué en cada una de ellas se pueda observar una de las 4 zonas de interés (Agua, Vías, Zona Urbana o Zona Verde) o combinaciones de ellas. Para esto se consultaron bases de datos gratuitas como lo son ImageNet, Pixabay y Pexels.

A continuación, se muestran algunas de las imágenes recopiladas:

Fig*. 3-1.* Imágenes aéreas seleccionadas para la zonificación.

Fuente: ImageNet, Pixabay, Pexels.

En total se obtuvieron 1000 imágenes, las cuales son tratadas de diferentes maneras con los 2 métodos seleccionados.

A continuación, se explica cómo se llevó a cabo cada uno de ellos.

3.2. DETECCIÓN DE OBJETOS

El primer método que se implemento fue la detección de objetos haciendo uso de redes neuronales convolucionales. Esta red es veloz en cuanto al tiempo de entrenamiento y procesamiento de las imágenes, su estructura no es compleja lo cual permite una implementación sencilla y además recibe una gran cantidad de imágenes.

Para empezar, se descargan los modelos ya configurados junto con un pre entrenamiento de cada uno. Luego, se configura el conjunto de datos de entrada a partir del banco de imágenes obtenido teniendo en cuenta que las redes neuronales no admiten ciertos tipos de imágenes que tengan características tales como:

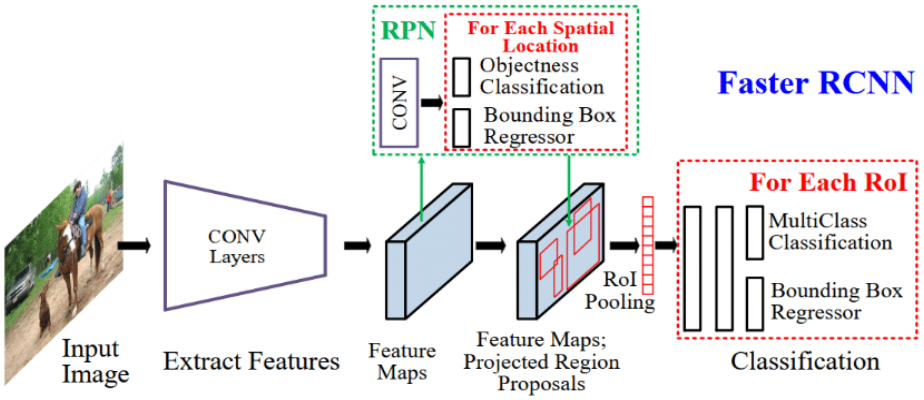
- Un peso mayor a 200 KB.
- El ancho mínimo de la imagen es de 600 pixeles y el alto máximo es de 720 pixeles.

Por tales motivos, se ajustan las imágenes de entrada para que cumplan con las especificaciones requeridas.

3.2.1. ARQUITECTURA FASTER R-CNN INCEPTION V2

En este punto se implementa un modelo basado en la arquitectura Faster R-CNN (Ren, 2015), específicamente el modelo Faster R-CNN Inception V2, sin embargo, solo se utilizará el tramo de detección de objetos.

Fig. *3-2.* Arquitectura Faster R-CNN Inception V2.



Fuente: (Durdu, 2019).

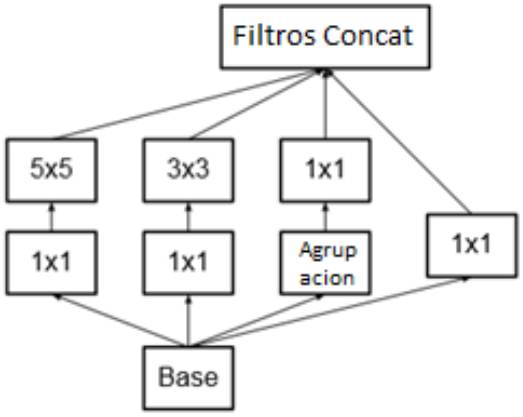
La red neuronal Faster R-CNN Inception V2 está diseñada para funcionar bajo ciertos parámetros de restricciones de memoria teniendo en cuenta el costo computacional del procesador en el cual se empleará, es por ello, que esta arquitectura tiene un costo computacional razonable, el cual, le permite implementarse en dispositivos móviles. Este diseño de red está diseñado para procesar una gran cantidad de datos mediante la optimización de la ejecución de operaciones.

3.2.2. CARACTERÍSTICAS DE LA RED

Al utilizar esta red neuronal primero se hace un estudio sobre su estructura interna y su comportamiento a medida que va aprendiendo. Se encuentra que esta red tiene una forma particular para disminuir el costo computacional, pues se sabe que cada entrada de la red debe multiplicarse por el peso, entonces la red tiene una factorización para hacer la reducción dimensional en la estructura de la capa al momento que hace su inicio y esto permite adaptarse rápidamente al entrenamiento.

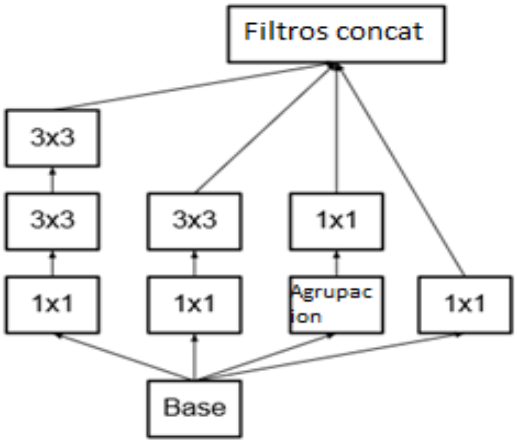
A continuación, se muestra cómo se descompone una capa de la red que tiene mayor efecto en el proceso de aprendizaje.

Fig. *3-3.* Descripción de la red con alto costo computacional.



Fuente: (Szegedy,2015).

Fig. *3-4.* Reducción dimensional de la capa de mayor tamaño.



Fuente: (Szegedy,2015).

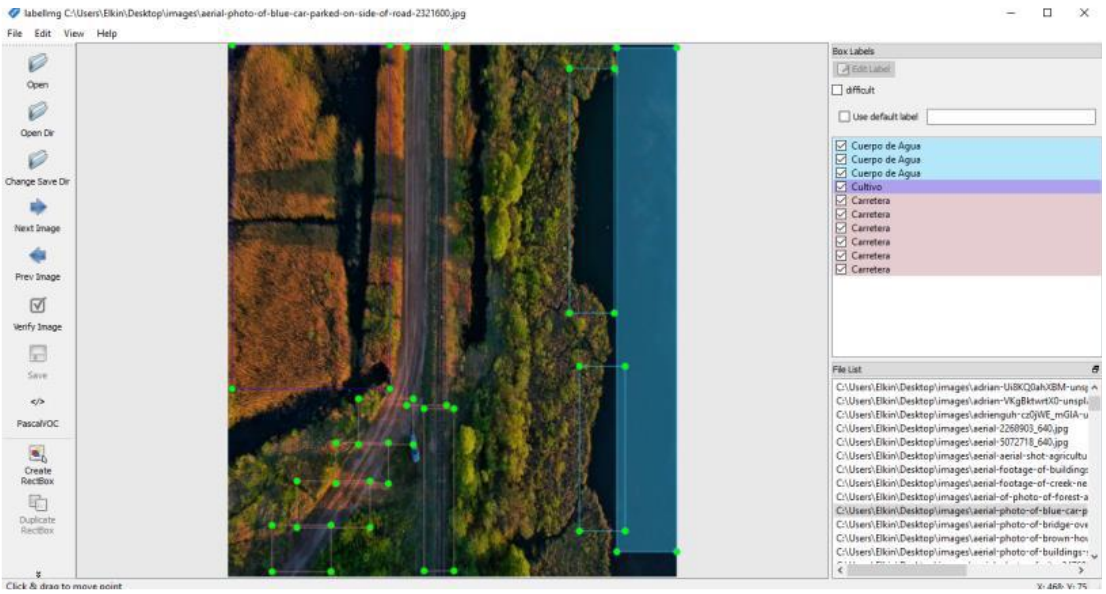
Cómo se puede observar, la matriz de 5x5 de la figura 3-3 se reduce a dos matrices de 3x3 en la figura 3-4, teniendo en cuenta que esto no generará cuellos de botella en el entrenamiento, lo cual aumentará en un 33% la eficiencia.

3.2.3. OBTENCIÓN DEL CONJUNTO DE DATOS DE ENTRADA

Luego de esto es necesario realizar un etiquetado de las imágenes, ya que en este momento es donde se le indica a la red en qué parte de la imagen están localizadas cada una de las zonas de interés.

Para el etiquetado de imágenes se utiliza Labelimg, un software libre que genera un archivo en formato XML donde se obtienen las coordenadas de cada uno de los objetos identificados. Este software permite crear un “Bounding box” donde se encuentra ubicado el objeto (Zona de interés) que será lo que la red neuronal entenderá para hacer su entrenamiento y generar el modelo predictivo.

Fig. *3-5.* Proceso de etiquetado en Labelimg.



Fuente: Labelimg.

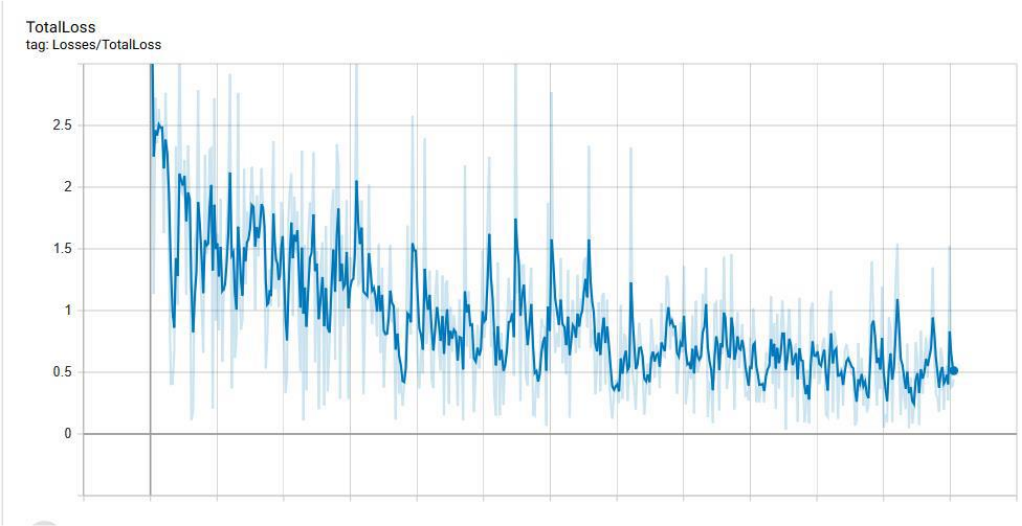
Este procedimiento se realiza con las 1000 imágenes de entrada, sin embargo, se usarán 500 imágenes para realizar el entrenamiento de la red y 200 para realizar la prueba después de generar el modelo.

3.2.4. ENTRENAMIENTO

Se procede a entrenar la red con el modelo previamente obtenido, obteniendo como resultado archivos en formato CVS, esto se hace con el fin de que la información de las imágenes se comprima en filas de información, por ejemplo, el nombre de la imagen o las coordenadas en cada punto. Posterior a ello, se convierte a TFRecords que es un tipo archivo que contiene todas las imágenes que estén dentro del proceso de aprendizaje y toda la información de los archivos CSV.

El entrenamiento se realiza con el tipo de archivo TFRecords.

Fig. *3-6.* Pérdida total en las últimas épocas durante el aprendizaje de la red Inception V2.



Fuente: Autores.

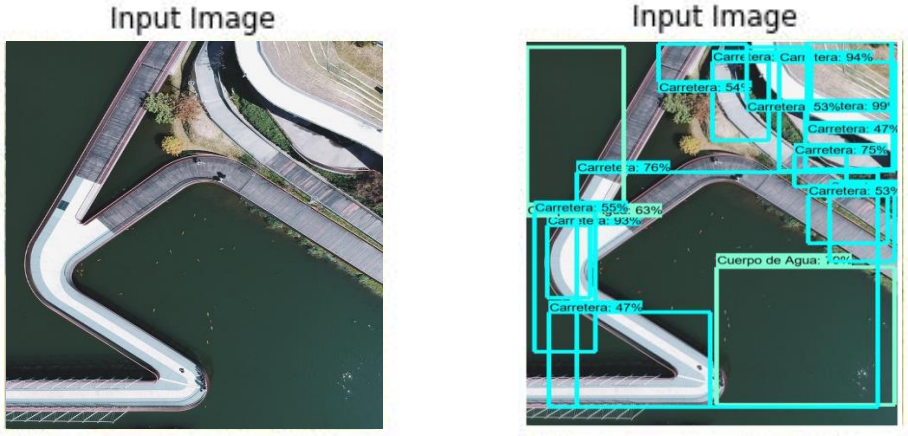
Al final del entrenamiento, la red tuvo un error cercano al 0.5% el cual es bueno considerando que se está trabajando con imágenes, uno de los campos más complejos de trabajar en el deep learning.

La red tardó un total de 27 horas, 38 minutos y 15 segundos para 16454 épocas. Detectando los objetos de una imagen nueva, la red tarda 57 segundos.

3.2.5. RESULTADOS

Al entrenar la red se ingresan algunas de las imágenes del conjunto de prueba para comprobar el funcionamiento de la red y determinar la precisión de la misma.

Fig. *3-7.* Imagen de entrada y objetos detectados por la red.



Fuente: Autores.

Como se puede ver, la red fue capaz de identificar casi el 70% de la superficie del agua presente en la imagen y un 60% de la superficie de las carreteras; ambas con una certeza promedio de 75%.

3.3. SEGMENTACIÓN SEMÁNTICA

Para llevar a cabo este procedimiento se utilizó la misma arquitectura PIX2PIX de súper resolución (Ver Alg. 2-3), realizando modificaciones en el generador (U-Net). Estas modificaciones son el cambio del tamaño de las imágenes de salida, pasando de 2024 x 2024 a 256 x 256. Esto significa que el número de bloques del encoder y del decoder se reduce, pasando de 11 bloques a 8 bloques. De resto, todo se deja igual, sin embargo, el conjunto de datos de entrada y de salida sí cambió.

Las imágenes de entrada son aquellas mencionadas anteriormente y las de salida son las mismas imágenes, pero con la segmentación realizada manualmente.

3.3.1. IMÁGENES DE SALIDA

Para realizar la segmentación se decidió asociar un color a cada categoría.

Tabla 3-1. Relación de la zona de interés con su respectivo color.

Zona de interés	Color RGB	Nombre del color	Apariencia
Agua	[0,255,0]	Azul	
Vía o Carretera	[128,0,128]	Púrpura	
Zona Urbana	[255,0,0]	Rojo	
Zona Verde	[0,0,255]	Lima	

Fuente: Autores.

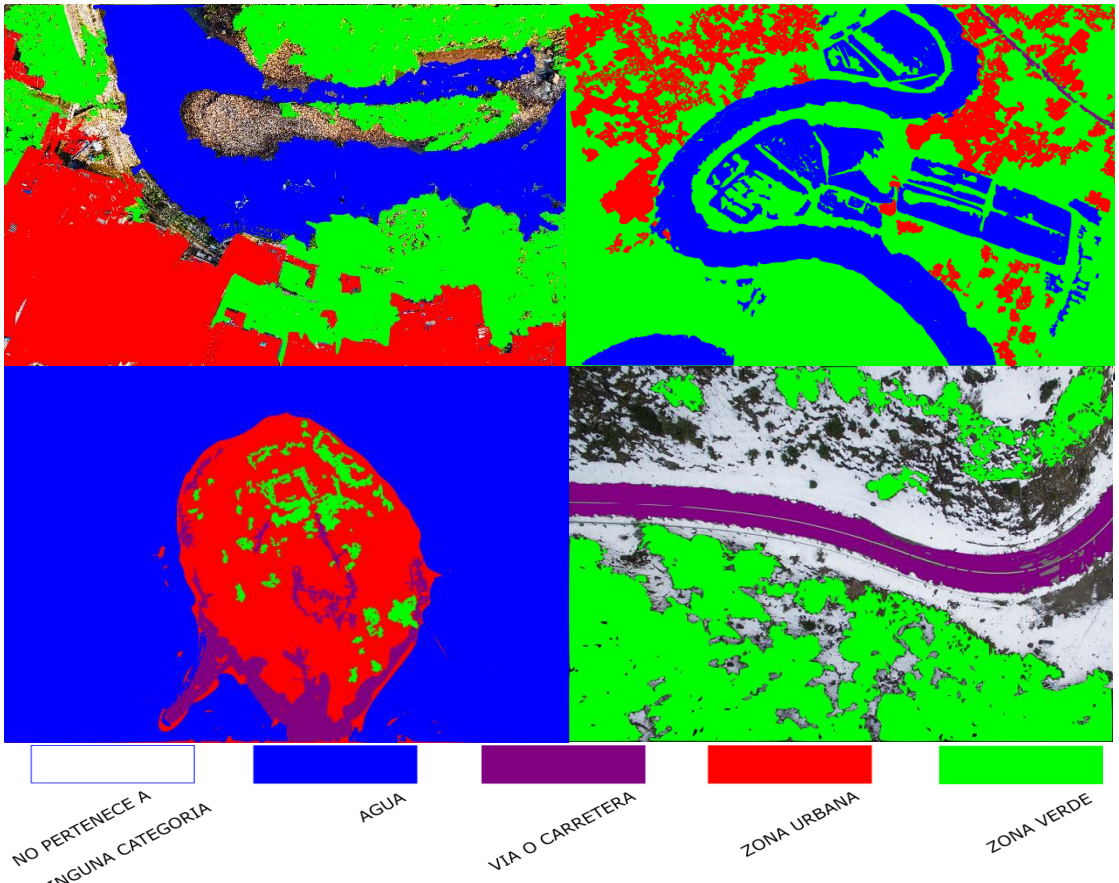
Luego de seleccionar los colores, se pintan cada una de las secciones con el color correspondiente. Este proceso es sin dudas el que toma más tiempo ya que hay zonas de las imágenes donde se entrecruzan las secciones por lo que se dificulta el proceso. Para simplificar un poco esto, se utilizó la herramienta Inkscape que posee una funcionalidad para pintar un pixel y sus vecinos por medio de un umbral, de tal forma que, si se pinta una zona que tiene, por ejemplo, agua, el programa pinta todos los pixeles que tienen un color similar, lo cual reduce significativamente el tiempo de pintado de todas las imágenes.

Fig. *3-8.* Segmentación manual utilizando Inkscape.

Fuente: Inkscape.

A continuación, se muestran las imágenes de la figura 3-1 segmentadas manualmente.

Fig. *3-9.* Imágenes aéreas seleccionadas segmentadas manualmente.



Fuente: Autores.

3.3.2. DEFINICIÓN DE PARÁMETROS

Como la arquitectura seleccionada es la misma PIX2PIX, no existen modificaciones como tal en cuanto al procedimiento a seguir, el alistamiento, aumento de datos y otros parámetros. El cambio se presenta a la hora de definir la arquitectura de la red generadora, es decir, la U-Net. Para este caso se agregó un bloque de más con una capa convolucional, una capa de normalización y una capa de activación, el resto se deja igual.

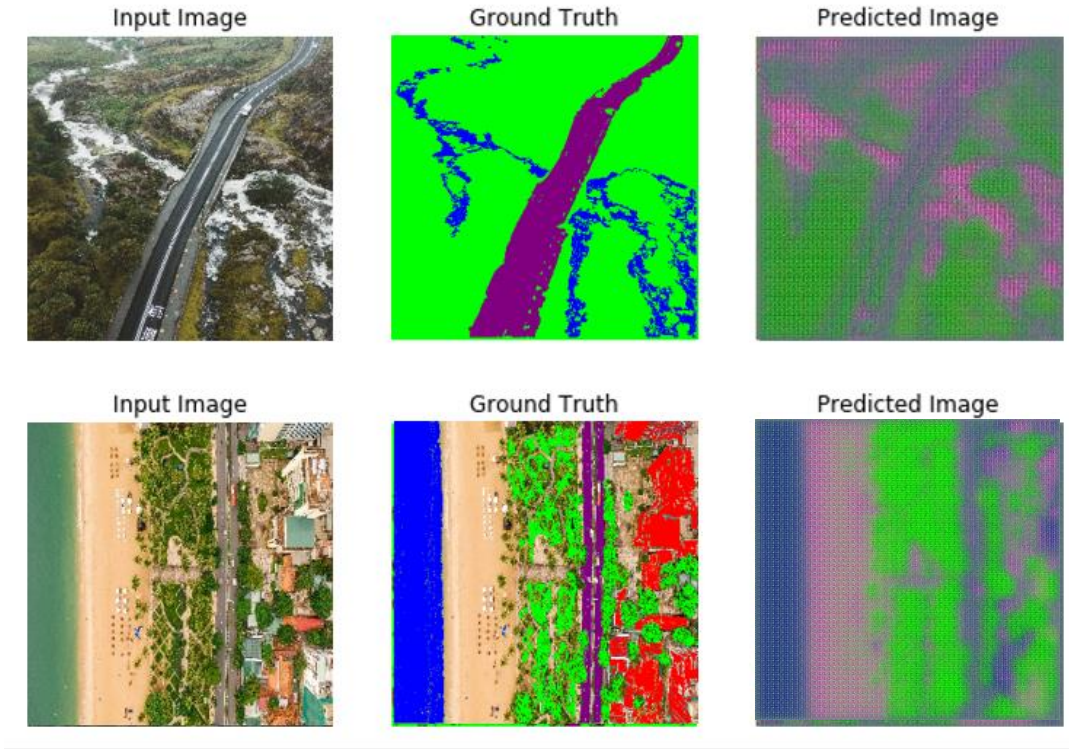
En el anexo 3 y 4 se presenta la arquitectura U-Net modificada para el generador de las imágenes segmentadas.

3.3.3. ENTRENAMIENTO

En total, se realizó la segmentación manual de 250 imágenes, las cuales se utilizan para realizar el entrenamiento. Luego de esto, se define la cantidad de épocas y en cada una se procesan las 250 imágenes. Se determinó que el número de épocas fuera de 300, sin embargo, para evitar el overfitting, se determina que cuando la perdida no varíe más de 0.01% durante 5 épocas consecutivas, el programa se detenga.

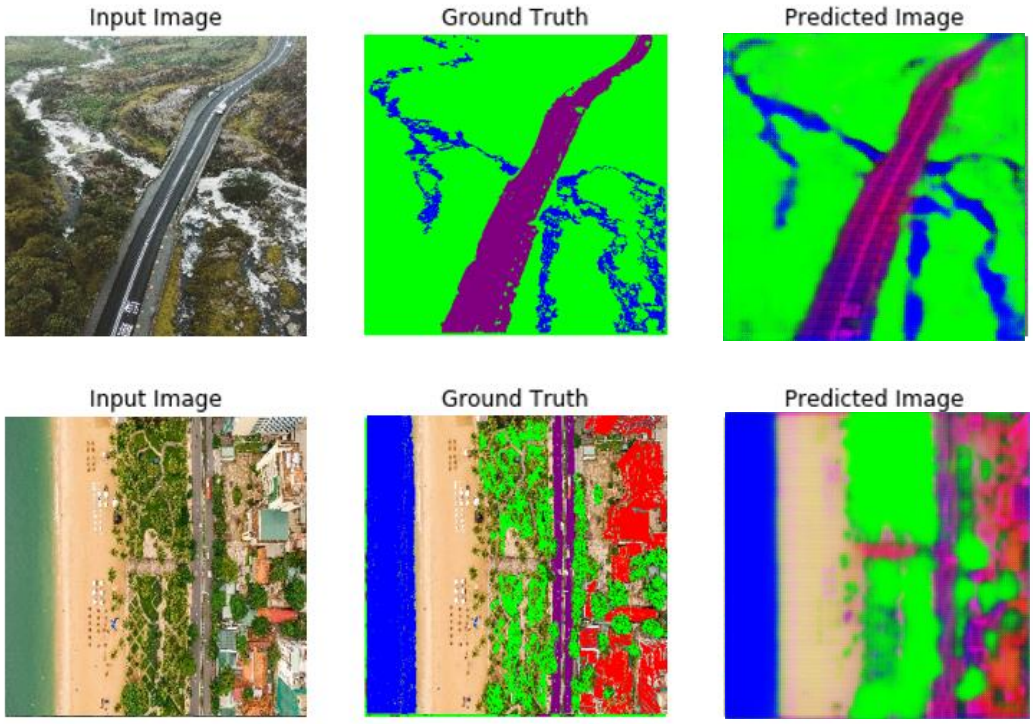
A continuación, se muestran algunas imágenes durante el proceso de entrenamiento de la red.

Fig. *3-10.* Imágenes de entrada, imágenes de salida e imágenes predichas en la primera época de la red de segmentación.



Fuente: Autores.

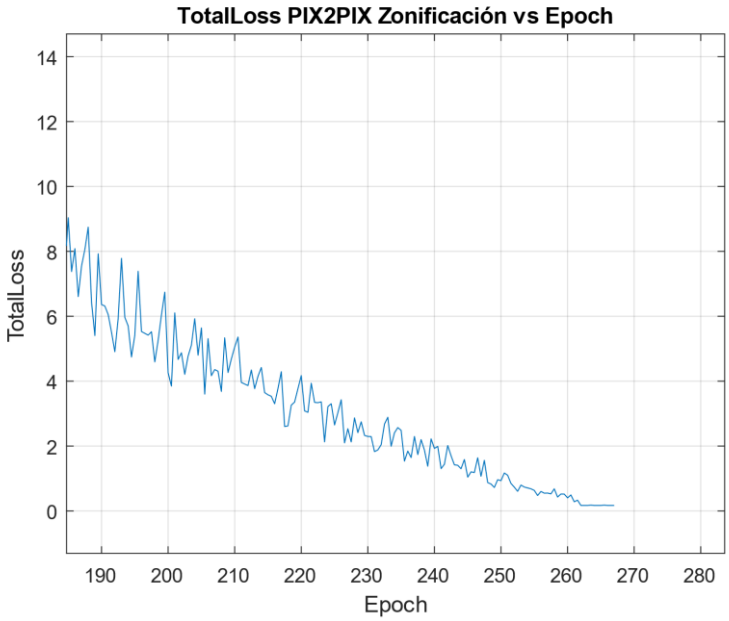
Fig. *3-11.* Imágenes de entrada, imágenes de salida e imágenes predichas en la época 150.



Fuente: Autores.

La red se entrenó por 267 épocas ya que en este punto hubo 5 épocas consecutivas donde no existió variación de más del 0,01%. Duró 35 horas, 6 minutos y 18 segundos.

Fig. *3-12.* Pérdida total en las últimas épocas durante el aprendizaje de la red PIX2PIX para zonificación.



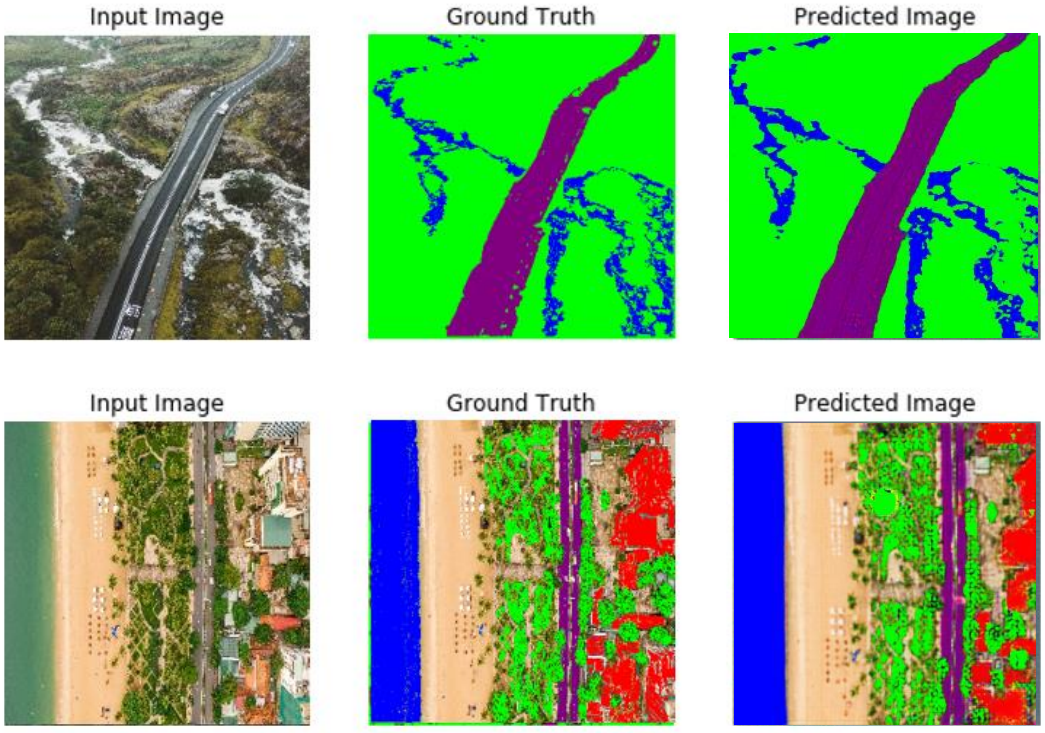
Fuente: Autores.

Se puede ver cómo el error decrece exponencialmente hasta acercarse a un valor de 0.17%, donde se mantuvo prácticamente estable desde la época 262 hasta la 267 que fue donde el programa se detuvo. Este error es mejor que el obtenido por la red detectora de objetos que fue de 0.5%.

Se prueba ahora con las mismas imágenes de entrenamiento y posteriormente con las de prueba.

3.2.4. RESULTADOS

Fig. *3-13.* Imágenes de entrada, imágenes de salida e imágenes predichas.



Fuente: Autores.

Al revisar los resultados, se puede ver como la red genera las imágenes ya vistas de forma eficiente. En realidad, esto es lo que debería suceder ya que la red vio esas imágenes. Para probar realmente el funcionamiento, se pondrán a prueba imágenes que la red no ha visto y observar si es capaz de pintar de forma eficiente las zonas de interés que haya en la imagen.

4. RESULTADOS

En este capítulo se muestran las imágenes obtenidas tanto en súper resolución como en zonificación comparándolas con diferentes métricas propias de cada tipo de imagen.

4.1. SÚPER RESOLUCION

Para comprobar cual método fue el que arrojó mejores resultados, se procede a comparar las 3 imágenes obtenidas con respecto a la imagen original (Ver figura 2-2).

Fig. *4-1.* Comparación de la zona de interés en la imagen original, imagen obtenida por interpolación lineal modificada, ESRGAN y PIX2PIX.

ORIGINAL INTERPOLACION ESRGAN PIX2PIX

Fuente: Autores

A simple vista las 4 imágenes se ven prácticamente iguales. Para determinar si en realidad mejoró la definición de las imágenes sería necesario acercarse a cada zona de la imagen, una por una y compararlas entre todas a la vez. Esto podría funcionar, pero es un proceso muy largo y es muy subjetivo de quien ve la imagen. Por ello, se establecen 3 métricas con fundamento matemático, las cuales permiten determinar con mayor precisión cual método arrojo la imagen de súper resolución más acertada y cercana a la realidad.

A estos procedimientos se les conoce como análisis de texturas. El análisis de texturas hace referencia a la caracterización de las regiones de una imagen por su contenido de textura. El análisis de texturas intenta cuantificar las cualidades de una imagen en función de la variación espacial en las intensidades de píxeles (MathWorks, 2020). El análisis de texturas es útil cuando los objetos de una imagen se caracterizan más por su textura que por la intensidad, y las técnicas de umbral tradicionales no se pueden utilizar de forma eficaz. Por tal motivo, el análisis de texturas es la mejor herramienta para determinar cuál imagen es mejor, ya que a simple vista todas parecen ser iguales.

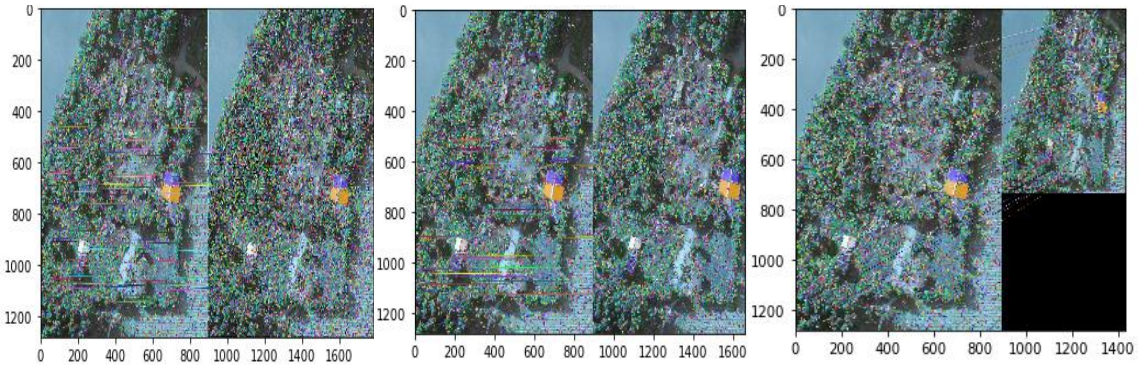
4.1.1. SIFT

Como ya se mencionó antes, este método permite relacionar características puntuales de dos imágenes. En este caso, se va a realizar la comparación de la imagen original con las 3 imágenes obtenidas y la que arroje mayor cantidad de puntos que concuerden en las dos imágenes será el mejor, puesto que la súper resolución mejora y detalla una imagen mas no cambia el color o los patrones de la imagen y, si esto sucede, serán zonas que no concordarán en las dos imágenes.

Se implementa el algoritmo SIFT entre la imagen original y cada una de las imágenes generadas asegurando que solo los mejores puntos de emparejamiento se tengan en cuenta, esto se hace filtrando la distancia, si está entre 0 y 0.7, es un emparejamiento valido.

Fig. *4-2.* 30 mejores emparejamientos entre la imagen original y las imágenes generadas por los distintos métodos.

INTERPOLACION ESRGAN PIX2PIX



Fuente: Autores.

Tabla 4-1. Puntos de emparejamiento de cada método con respecto a la imagen original.

Método De Súper Resolución	**Puntos de Emparejamiento**
Interpolación	1.626
ESRGAN	10.865
PIX2PIX	176

Fuente: Autores.

Como se puede observar en la tabla anterior, a partir de los datos que arroja el algoritmo SIFT, el mayor número de puntos de emparejamiento es el del método ESRGAN, pero esto no quiere decir que sea la mejor imagen en cuanto a súper resolución, por el contrario, lo que demuestra es que este método mejoró muy

poco la nitidez de la imagen, provocando que el algoritmo detecte gran cantidad de emparejamientos entendiendo como si las dos imágenes fueran la misma.

Como se están calculando los mejores puntos que no tengan una diferencia entre su posición original con respecto a su posición de la imagen resultante, esto quiere decir que los pixeles de la imagen donde generaron objetos más detallados, se les hizo una modificación a sus pixeles moviéndolos o aumentando el tamaño para generar un mayor número de pixeles dentro del mismo espacio, de esta manera, el lugar original fue directamente modificado.

Debido a esto, el método en el cual no se tuvo una mayor modificación dentro este espacio fue el ESRGAN. Por otra parte, el método PIX2PIX tampoco fue el mejor dado que se encontraron muy pocos emparejamientos y esto tiene sentido, ya que, al ver la imagen, se puede apreciar que ciertos detalles se pierden, por lo que el algoritmo es incapaz de relacionar puntos de una misma sección en las dos imágenes, pues en la obtenida por PIX2PIX se pierden detalles y estos ya no concordaran con la imagen original.

De esta manera, se puede concluir que el método que tuvo mejores resultados para una buena súper resolución teniendo como punto de medición el algoritmo SIFT es la interpolación lineal modificada, debido a que no tiene un alto número de puntos de emparejamiento, pues significaría que la imagen cambió muy poco, pero tampoco son lo suficientemente cercanos a cero, pues sería una imagen en la que se pierden los detalles y la nitidez.

4.1.2. ENTROPÍA

La entropía es una medida estadística de aleatoriedad que se puede utilizar para caracterizar la textura de una imagen. La entropía se define como los estados correspondientes del nivel de intensidad en que los píxeles individuales se pueden adaptar. Se utiliza en el análisis cuantitativo y en los detalles de la imagen de evaluación, por ello, el valor de entropía se utiliza, ya que proporciona una mejor comparación de los detalles de la imagen. De tal forma que un valor mayor de entropía significa información más clara, es decir, una imagen que ofrece mayor cantidad de detalles (Kopriva, 2016).

La entropía se define matemáticamente de la siguiente forma:

$$H(s_m) = -\sum_{n=1}^{256} p_n(s_m) \cdot \log_2\big(p_n(s_m)\big) \tag{4-1}$$

$H(s_m)$ es un valor numérico que representa la entropía de la imagen en escala de grises.

$p_n(s_m)$ es la densidad probabilística, la cual se calcula a partir del histograma de la imagen.

Tabla 4-2. Entropía de la imagen original y de la obtenida con cada uno de los métodos.

Método De Súper Resolución	Entropía
Original	7.4256
Interpolación	7.7345
ESRGAN	7.4188
PIX2PIX	7.4201

Fuente: Autores.

Según los datos obtenidos, se puede observar que la entropía fue menor en las imágenes obtenidas por el método ESRGAN y PIX2PIX con respecto a la imagen original, mientras que la imagen obtenida por el método de interpolación tuvo mayor entropía. Esto demuestra que la imagen original ofrece mayor cantidad de detalles con respecto a la imagen obtenida por ESRGAN y PIX2PIX, sin embargo, la imagen obtenida por interpolación lineal ofrece más detalles que la imagen original, por tal motivo, el método con mejores resultados teniendo como punto de medición la entropía es la interpolación lineal modificada.

4.1.3. NITIDEZ

Se define como la claridad del detalle en una fotografía. Una imagen nítida se ve de forma clara tanto en el enfoque como en el detalle (Zafra, 2020).

Este es uno de los indicadores más lógicos para determinar si una imagen se ve mejor que otra. Una persona podría decir con lo que ve si esto es cierto o no, pero este asunto se puede tratar desde un enfoque matemático el cual es objetivo.

Para calcular la nitidez de una imagen existen distintas técnicas, sin embargo, se va a hacer uso de la magnitud promedio del gradiente (Mlnsa, 2009). Esta técnica consiste en calcular la magnitud del gradiente de una imagen, es decir, la tasa máxima de cambio de intensidad en un pixel. Posterior a ello se obtiene el promedio de esta magnitud en todos los pixeles.

Esta técnica se puede representar mediante la siguiente ecuación:

$$SP = \overline{\sqrt{\nabla fx^2 + \nabla fy^2}} \tag{4-2}$$

SP (Sharpness) significa el nivel de nitidez de la imagen, donde un valor cercano a 0 significa que la imagen es completamente borrosa y un valor por encima de 20 indica que la imagen tiene buena nitidez (Atkins, 2006).

∇fx es el gradiente de la imagen en la dirección x.

∇fy es el gradiente de la imagen en la dirección y.

Utilizando la ecuación anterior se calcula la nitidez de la imagen original, la obtenida con la red PIX2PIX, ESRGAN e interpolación lineal modificada.

Tabla 4-3. Nitidez de la imagen original y de la obtenida con cada uno de los métodos.

Método De Súper Resolución	**Nitidez**
Original	13.58014
Interpolación	28.117
ESRGAN	14.5721
PIX2PIX	13.2368

Fuente: Autores.

4.1.4. COMPARACIÓN FINAL

Con base a los resultados obtenidos, se puede determinar que la mejor imagen es la que se obtuvo por el método de interpolación lineal, la cual, es la que brinda la mejor información, por ende, la que mejores detalles ofrece. Todo esto lo corrobora el haber sido la que mejores resultados obtuvo en las 3 métricas, es decir, SIFT, entropía y nitidez.

Seguido se encuentra la imagen generada por ESRGAN que mejora ligeramente la nitidez de la imagen original. Esto se puede corroborar visualmente, pues se pueden apreciar algunas mejoras en las texturas de los objetos con respecto a la imagen original. Además, obtuvo el segundo lugar en SIFT y nitidez y el tercero en entropía.

Finalmente, la imagen obtenida por PIX2PIX no fue buena, ya que incluso tuvo peor nitidez que la imagen original, esto lo corrobora su tercer puesto en SIFT y nitidez y su segundo puesto en entropía. Sin embargo, cabe recordar que las imágenes de entrada durante el entrenamiento de la red eran imágenes reducidas de tamaño (menor resolución) con respecto a la imagen original y las imágenes de salida eran las imágenes originales, por lo que es de esperar que la nitidez sea peor que la imagen original.

Para acompañar los resultados obtenidos anteriormente, se muestran las 4 imágenes con su valor obtenido en cada una de las métricas.

Fig. *4-3.* Comparación de la zona de interés acercada en la imagen original, imagen obtenida por interpolación lineal modificada, ESRGAN y PIX2PIX con sus respectivos resultados.

ORIGINAL

$NITIDEZ = 13.58014$
$ENTROPÍA = 7.4256$
$SIFT = N/A$

INTERPOLACIÓN

$NITIDEZ = 28.1117$
$ENTROPÍA = 7.7345$
$SIFT = 1626$

ESRGAN

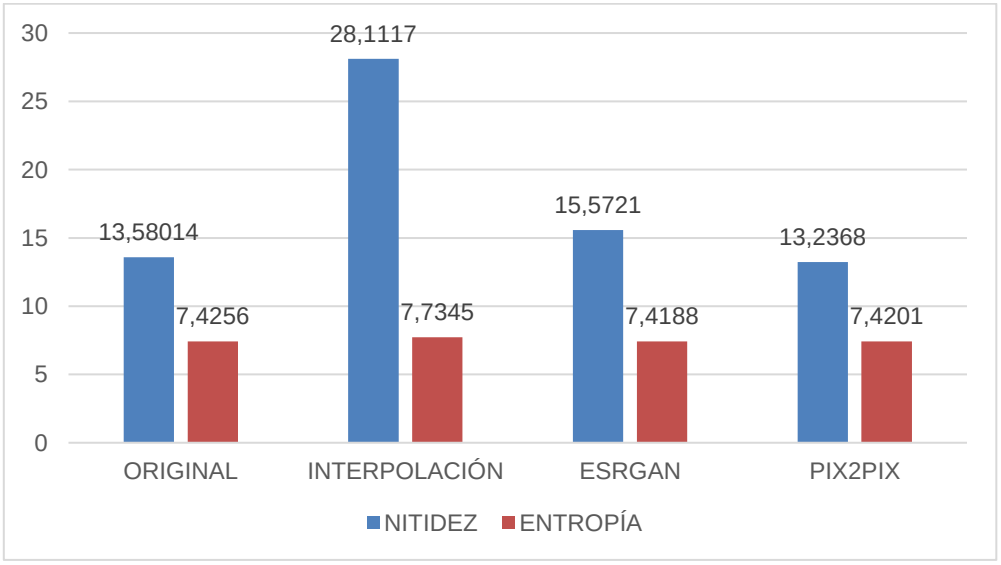
$NITIDEZ = 14.5721$
$ENTROPÍA = 7.4188$
$SIFT = 10865$

PIX2PIX

$NITIDEZ = 13.2368$
$ENTROPÍA = 7.4201$
$SIFT = 176$

Fuente: Autores.

Fig. *4-4.* Comparación de métricas de súper resolución.



Fuente: Autores.

4.2. ZONIFICACIÓN

Como la segmentación semántica se asemeja más a la zonificación, se hará la prueba solo con esta técnica.

Para determinar la eficiencia de la red, es necesario ingresar imágenes que la red no haya visto antes. Para ello, se utiliza el algoritmo 2-3, sin embargo, se realizan algunas modificaciones ya que este algoritmo se utiliza para entrenamiento. Es por ello que se realiza una modificación desde la sección del Checkpoint (Ver punto rojo en el algoritmo 2-3) hasta el final del seudocódigo. Esta modificación consiste en cargar los valores del modelo ya entrenados e implementarlos sobre la red generadora, ya que la red discriminadora no se utiliza, pues esta solo es necesaria para el entrenamiento.

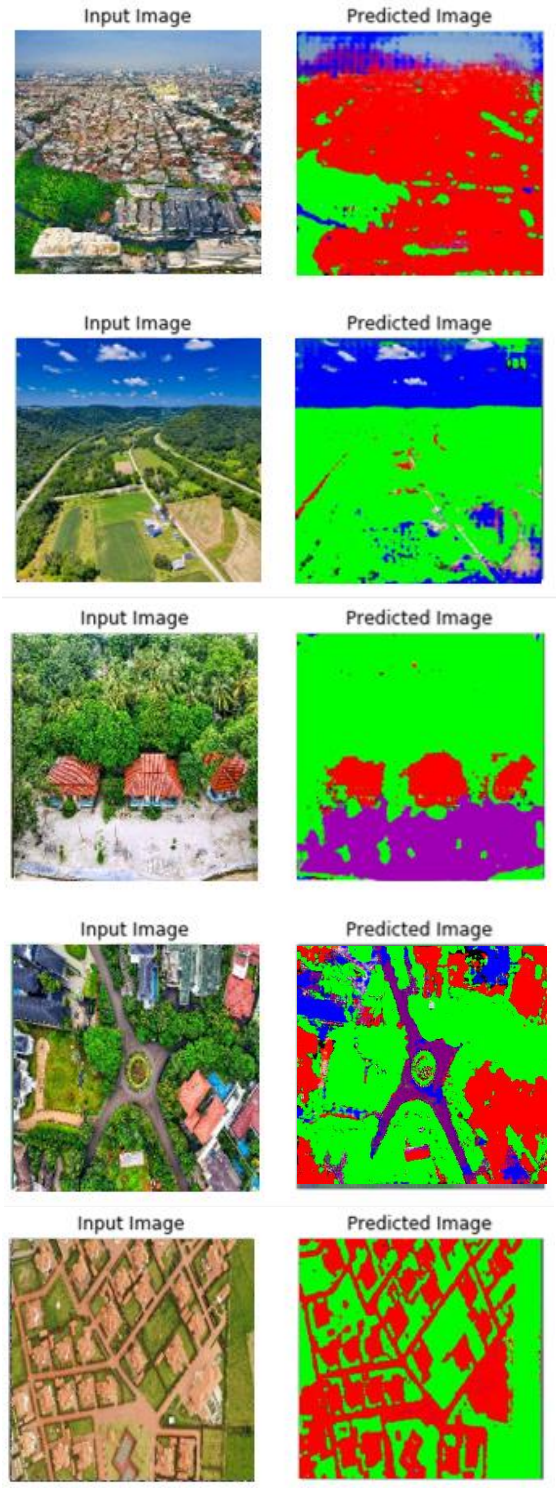
A continuación, se presentan las modificaciones realizadas al algoritmo 2-3 en forma de seudocódigo.

Alg. *4-1.* Seudocódigo con las modificaciones realizadas al algoritmo 2-3.

```
        // Recuperar los valores del entrenamiento del generador.
        RecuperarCheckpoint(GuardarCheckpoint, CHPATH)

        // Cargar la ruta de la imagen a zonificar.
        Definir RutaImagen Como Cadena

        // Abrir imagen a zonificar.
        Img_Prueba = AbrirImagen(RutaImagen)

        // Generar la zonificación de la imagen.
        Img_Zonificada = EvaluarModelo(Generador, Img_Prueba)

        // Cargar la ruta de la imagen a zonificar.
        Definir RutaImagenZonificada Como Cadena

        // Guardar imagen generada.
        GuardarImagen(Img_Zonificada, RutaImagenZonificada)

FinAlgoritmo
```

Después de implementar el algoritmo 2-3 con las modificaciones del algoritmo 4-1, se obtiene la zonificación de la imagen que se desee. Por lo tanto, se utilizan varias imágenes del conjunto de datos de prueba para observar los resultados. Según lo obtenido, se presentan 5 imágenes las cuales, a consideración de los autores y según las imágenes resultantes, representan los escenarios de la red, es decir; hay algunas imágenes resultantes de la red mal segmentadas; otras, en mayor medida, obtuvieron una segmentación aceptable y algunas otras fueron segmentadas de forma muy acertada.

Fig. *4-5.* Imágenes de entrada e imágenes segmentadas por la red PIX2PIX.



Fuente: Autores.

Al observar a simple vista todas las imágenes se puede observar que la segmentación es muy buena. En la primera imagen se pintó correctamente la zona urbana, la zona verde y el agua, solo hubo una confusión en el cielo ya que lo asumió como si fuera agua y carretera.

Con respecto a la segunda imagen, se puede ver que tuvo mayores dificultades puesto que las carreteras no las detectó y confundió todo el cielo con agua, lo cual no es un disparate puesto que el color del cielo es muy similar al color del agua y además ocupa gran proporción.

En la tercera imagen se presentó una segmentación casi perfecta, ya que se pintó adecuadamente la zona urbana, la zona verde y la vía frente a las casas, solo existieron algunas zonas muy pequeñas que se interpretaron como agua.

La cuarta imagen también tuvo una muy buena segmentación ya que la vía y la zona verde fueron pintadas con precisión. Con respecto a la zona urbana, la mayoría de las casas fueron pintadas adecuadamente y en algunas, los techos fueron confundidos con agua puesto que el color es muy similar.

Finalmente, la última imagen es la que mejor resultad tuvo puesto que la zona de las casas y la zona verde corresponde perfectamente a lo que es.

Este análisis se podría realizar con cada nueva imagen que se genere, pero su apreciación es subjetiva. Para determinar si de verdad las imágenes generadas son buenas o no se hace uso de algunas métricas.

4.2.1. ANALISIS DE RESULTADOS

4.2.1.1. EXACTITUD PIXEL A PIXEL

Es una métrica que consiste en determinar qué tanto varió la intensidad de un pixel con respecto a otro en la misma posición. Esto permite determinar si la imagen generada cambió tonos o patrones con respecto a la imagen original.

Para calcular esto, en primer lugar, se debe pintar manualmente la imagen original con las zonas correspondientes, esto para mirar las diferencias entre ambas imágenes. Después, se ajustan ambas imágenes a un mismo tamaño, en este caso, 1500 x 1500 x 3 pixeles. Luego se obtiene el histograma correspondiente por canal. Posteriormente, se resta pixel a pixel la intensidad, esto da como resultado una imagen en blanco y negro en donde los pixeles en negro significan que son zonas en las que ambas imágenes están pintadas de la misma manera, mientras que las zonas grises y blancas son aquellas en donde hubo diferencias. Por tal motivo, este proceso se hace para cada imagen y en cada uno de los canales, pues es necesario hallar la precisión en cada una de las categorías.

Fig. *4-6.* Diferencia entre la segmentación manual y la generada por la red PIX2PIX en el canal R de la primera imagen de prueba.

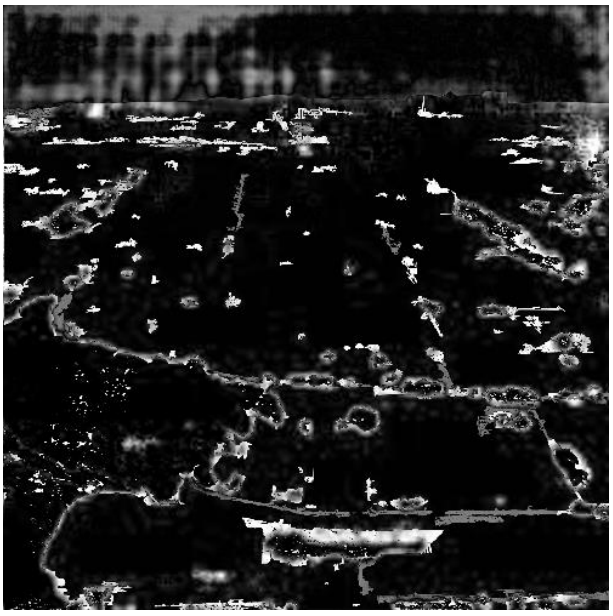


Figura: Autores.

Para que la segmentación generada por la red fuera perfecta, la imagen debería ser completamente negra al no existir diferencias. Es por ello que se calcula la cantidad de pixeles que no concordaron, es decir, aquellos en donde la intensidad es cercana a 255. Luego se implementa la siguiente ecuación:

$$PAc = 100\% - \frac{kc \cdot 100}{s} \quad (4\text{-}3)$$

PAc corresponde al porcentaje de acierto por canal.

kc es la cantidad de pixeles donde hubo diferencias entre las dos imágenes.

s es la cantidad de pixeles del color correspondiente en la imagen.

Al implementar esta ecuación en cada uno de los canales y en cada una de las imágenes se obtiene lo siguiente:

Tabla 4-4. Porcentaje de acierto en cada categoría de la imagen 1.

CATEGORÍA	PORCENTAJE DE ACIERTO
Agua	56.9450
Vía o Carretera	46.2345
Zona Urbana	89.0237
Zona Verde	86.2690
Promedio	69.1180

Fuente: Autores.

Tabla 4-5. Porcentaje de acierto en cada categoría de la imagen 2.

CATEGORÍA	PORCENTAJE DE ACIERTO
Agua	10.032
Vía o Carretera	11.3459
Zona Urbana	22.1398
Zona Verde	93.8044
Promedio	34.3305

Fuente: Autores.

Tabla 4-6. Porcentaje de acierto en cada categoría de la imagen 3.

CATEGORÍA	PORCENTAJE DE ACIERTO
Agua	98.2369
Vía o Carretera	88.2342
Zona Urbana	95.2456
Zona Verde	94.6621
Promedio	94.0947

Fuente: Autores.

Tabla 4-7. Porcentaje de acierto en cada categoría de la imagen 4.

CATEGORÍA	PORCENTAJE DE ACIERTO
Agua	32.1984
Vía o Carretera	85.4522
Zona Urbana	80.2496
Zona Verde	92.1689
Promedio	72.5173

Fuente: Autores.

Tabla 4-8. Porcentaje de acierto en cada categoría de la imagen 5.

CATEGORÍA	PORCENTAJE DE ACIERTO
Agua	100
Vía o Carretera	100
Zona Urbana	96.2864
Zona Verde	94.3568
Promedio	97.6613

Fuente: Autores.

Al observar el porcentaje de acierto final en cada imagen y promediarlo, se puede determinar que el programa tiene una exactitud del 73.544%.

4.2.1.2. MATRIZ DE CONFUSIÓN

Ahora bien, para complementar el porcentaje de acierto es necesario saber cuántas categorías fueron catalogadas correctamente y cuantas no. Para ello, se utiliza la matriz de confusión, la cual es una herramienta que muestra el desempeño de un algoritmo de clasificación, describiendo cómo se distribuyen los valores reales y los valores predichos por el algoritmo (Jalife, 2017).

Para crear esta tabla es necesario identificar en cada imagen cuantos pixeles fueron clasificados correctamente y cuantos no.

Tabla 4-9. Matriz de confusión de la imagen 1.

CLASE	AGUA	VIA	ZONA URBANA	ZONA VERDE	**TOTAL**
AGUA	51364	236	189	2105	53894
VIA	429	539	198	246	1412
ZONA URBANA	47219	3289	1169821	216945	1537274
ZONA VERDE	1513	1647	128784	408674	440618
TOTAL	100525	5711	1398992	527970	2033198

Fuente: Autores.

Tabla 4-10. Matriz de confusión de la imagen 2.

CLASE	AGUA	VIA	ZONA URBANA	ZONA VERDE	**TOTAL**
AGUA	0	3124	31245	47982	82351
VIA	0	726	416	1143	2285
ZONA URBANA	0	12475	21465	731	34671
ZONA VERDE	0	45127	25678	1214587	1285392
TOTAL	0	61452	78804	1264443	1404699

Fuente: Autores.

Tabla 4-11. Matriz de confusión de la imagen 3.

CLASE	AGUA	VIA	ZONA URBANA	ZONA VERDE	**TOTAL**
AGUA	946	623	216	312	2097
VIA	197	319432	2167	4765	326561
ZONA URBANA	96	3799	109418	4027	117340
ZONA VERDE	175	7415	10167	1674239	1692086
TOTAL	1414	331269	121968	1683343	2138084

Fuente: Autores.

Tabla 4-12. Matriz de confusión de la imagen 4.

CLASE	AGUA	VIA	ZONA URBANA	ZONA VERDE	**TOTAL**
AGUA	31577	9354	28662	5334	74927
VIA	293	395478	12347	2183	410301
ZONA URBANA	27465	4184	518859	10635	561143
ZONA VERDE	1783	2149	19572	1115371	1138875
TOTAL	61118	411165	579440	1133523	2185246

Fuente: Autores.

Tabla 4-13. Matriz de confusión de la imagen 5.

CLASE	AGUA	VIA	ZONA URBANA	ZONA VERDE	**TOTAL**
AGUA	0	0	0	0	0
VIA	0	0	0	0	0
ZONA URBANA	0	0	1329416	47894	1377310
ZONA VERDE	0	0	89175	706136	795311
TOTAL	0	0	1418591	754030	2172621

Fuente: Autores.

4.2.1.3. EXACTITUD, PRECISION Y SENSIBILIDAD

Para medir la eficacia de cada imagen generada se utilizan las métricas de exactitud, precisión y sensibilidad (Metz, 1978), las cuales son calculadas por medio de las siguientes fórmulas:

$$Ex = \frac{\sum_{i=1}^{M} TPi}{TOTAL} \tag{4-4}$$

$$Pr = \frac{\sum_{i=1}^{M} \frac{TPi}{TPi+FPi}}{M} \tag{4-5}$$

$$Se = \frac{\sum_{i=1}^{M} \frac{TPi}{TPi+FNi}}{M} \tag{4-6}$$

Donde Ex es exactitud, Pr es precisión y Se es sensibilidad.

M representa el número de clases.

Total representa el número de muestras presentadas a clasificar.

TP representan a los valores adecuadamente marcados como positivos

FP los valores erróneamente marcados como positivos

FN los valores marcados erróneamente como negativos.

Tabla 4-14. Exactitud, precisión y sensibilidad en las 5 imágenes de prueba.

MÉTRICA	IMAGEN 1	IMAGEN 2	IMAGEN 3	IMAGEN 4	IMAGEN 5
EXACTITUD	0.7411	0.5621	0.9563	0.7824	0.9252
PRECISIÓN	0.5391	0.3112	0.8812	0.7391	0.9684
SENSIBILIDAD	0.7261	0.4704	0.8378	0.7223	0.9632

Fuente: Autores.

Con base a la tabla obtenida se puede determinar que la red obtuvo un 78.342% de exactitud, 68.78% de precisión y 74.396% de sensibilidad, lo cual conlleva a un error cercano a 26.7533% por parte de la red. Al computar este valor con el porcentaje de acierto hallado anteriormente de 73.544%, se determina que la red tiene un error promedio de 26.3083%, lo cual, en el campo de la segmentación semántica es muy bueno, teniendo en cuenta que algunas redes importantes tales como ICCV, DeepLAB y FCN-8s Front End obtuvieron 30%, 26.1% y 28.7% respectivamente (Mwiti, 2019).

5. IMPLEMENTACION

Consiste en aplicar la súper resolución en la imagen original (Ver figura 2-1) y luego realizar la segmentación. Por ello, se va a utilizar el método de interpolación lineal modificada (Ver Alg. 2-1) para generar la imagen de súper resolución y la red PIX2PIX (Ver Alg. 2-3 y modificación presentada en Alg. 4-1) para realizar la segmentación.

Fig. *5-1.* Diferencia entre la imagen original y la generada por el método de interpolación lineal modificada.

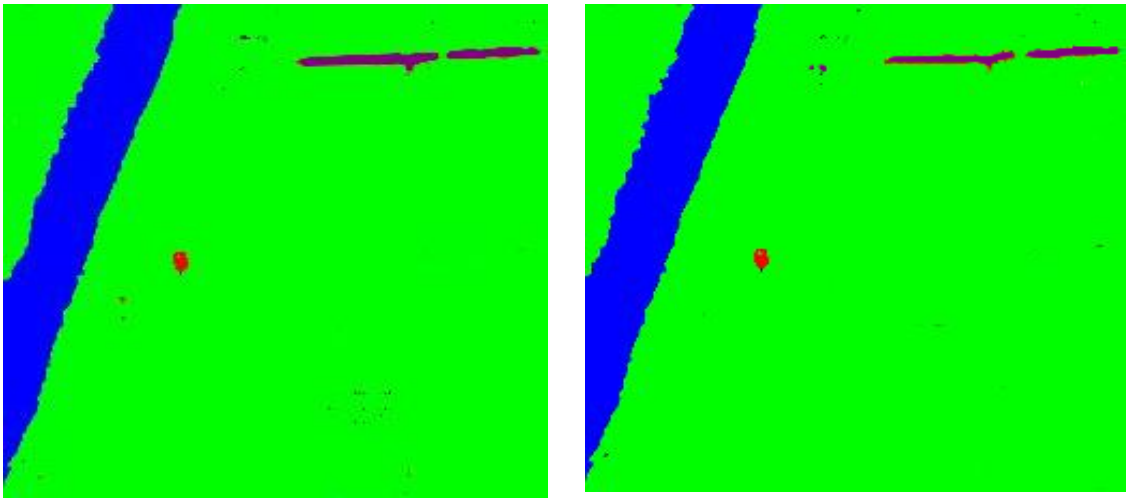
ORIGINAL INTERPOLACIÓN

Fuente: Autores.

Ambas imágenes pasan por la red PIX2PIX de zonificación obteniendo los siguientes resultados.

Fig. *5-2.* Zonificación de la imagen original y de la imagen de súper resolución.

ORIGINAL INTERPOLACIÓN



Fuente: Autores.

A simple vista no se aprecian grandes diferencias, sin embargo, al hacer un análisis detallado se puede notar que en la imagen de súper resolución existe menos ruido y se cubre una mayor superficie del agua.

El nivel de ruido se calcula con la entropía, la cual, para la imagen zonificada de baja resolución es de 0.70388 mientras que en la de súper resolución es de 0.70722, es decir, tiene mayor entropía, lo que significa que la imagen de súper resolución zonificada brinda más detalles que la de baja resolución.

Por otro lado, se comprueba la diferencia entre la zonificación real y las dos recién obtenidas, esto para determinar si, efectivamente, la imagen de súper resolución brinda una mejor zonificación. Para esto se utiliza la métrica de exactitud pixel a pixel, para determinar cuánto porcentaje de cada zona fue acertado.

Tabla 5-1. Porcentaje de acierto en cada categoría de la imagen original.

CATEGORÍA	PORCENTAJE DE ACIERTO
Agua	88.4995
Vía o Carretera	40.6218
Zona Urbana	44.4364
Zona Verde	96.2341
Promedio	67.4480

Fuente: Autores.

Tabla 5-2. Porcentaje de acierto en cada categoría de la imagen de súper resolución.

CATEGORÍA	PORCENTAJE DE ACIERTO
Agua	89.3674
Vía o Carretera	41.2365
Zona Urbana	45.2415
Zona Verde	96.6627
Promedio	68.1270

Fuente: Autores.

6. CONCLUSIONES

En este trabajo se han presentado diversas formas de resolver dos problemas, la súper resolución y la zonificación. En cuanto a la súper resolución, se obtuvo una mejora considerable al utilizar el método de interpolación lineal modificado, pero este método es el que más tiempo utiliza para su ejecución teniendo en cuenta que siempre se debe tener un conjunto amplio de imágenes de la misma escena con ligeras variaciones entre las imágenes.

La imagen obtenida por el método de interpolación lineal modificada es la que mejores resultados da puesto que la imagen final se obtuvo a partir de múltiples imágenes que están ligeramente rotadas o desplazadas, lo cual permite explorar la escena con mayor detalle. Es decir, la información de la imagen final se obtiene de la realidad, mientras que, al utilizar los métodos de redes neuronales, estas crean la información a partir de una base, la estiman (intensidad de los pixeles en cada canal) pero no es real, no se obtuvo de muchas muestras sino solo de una, es por esto que el método de interpolación lineal modificado propuesto por los autores, brinda los mejores resultados, pues se asemeja a una reconstrucción más cercana de la realidad.

Si bien la arquitectura PIX2PIX dio buenos resultados en la zonificación, no fue así en la súper resolución. Esto demuestra que, a pesar de que un modelo sea bueno y versátil, no necesariamente arroja buenos resultados para todos los casos, es por esto que existen tantos tipos de arquitecturas de redes neuronales, pues cada una de ellas se especializa en una tarea diferente y el utilizar una arquitectura en una tarea para la cual no fue propiamente diseñada puede provocar un desempeño poco eficiente.

Se puede comprobar que la segmentación semántica es más efectiva y además brinda mayor información de la localización y tipo de objeto en una escena en comparación a la detección de objetos. La arquitectura utilizada fue PIX2PIX que obtuvo un error medio de 28.104%, lo cual es bueno en términos de segmentación semántica.

Con base en los resultados obtenidos, se puede determinar que la categoría con mejor segmentación fue la zona verde, luego la zona urbana, seguido del agua y, por último, las vías y carreteras. Esto sucede porque los tonos de la zona verde son muy característicos, es difícil encontrar tonos de verde que no sean zona verde. La zona urbana se caracteriza por tonos rojizos y blancos. El agua toma principalmente tonos azules, pero existe agua que puede ser de color blanco si el brillo del sol le pega de frente, a tonos amarillentos si el agua está sucia. Por último, las vías y carreteras toman únicamente tonos grises y blancos en distintas escalas, lo cual se confunde con agua cuando hay sombras o con zona urbana cuando le da mucha luz.

Una de las ventajas de utilizar las redes neuronales convolucionales es que ellas aprenden en un contexto global y no local. Esto significa que, en este caso específico de la zonificación, la red no aprende a asociar una intensidad de un pixel con la intensidad de otro pixel, sino que la red analiza un pixel y sus vecinos y con base a ese análisis es que realiza la predicción.

Esta es una ventaja enorme, ya que, si no fuera así, la imagen predicha sería una mezcla de muchos colores por todas partes, pues existen muchas tonalidades de pixel que pueden corresponder a distintas categorías, como sucede con las distintas tonalidades del agua, de los árboles o de las edificaciones.

RECOMENDACIONES Y TRABAJOS FUTUROS

La mejor recomendación que se puede dar a la hora de trabajar con deep learning y técnicas robustas como las que se han manejado a lo largo de todo el trabajo es contar con mucha paciencia y con los implementos adecuados, ya que durante la obtención de los resultados fue necesario probar muchas veces cada algoritmo pues se presentaban errores y también era necesario comprobar la repetitividad de cada algoritmo utilizado. Pero más allá de estos errores que dependen de quien programe o realice esta labor, es necesario contar con un buen computador. En este caso, se utilizó un procesador Intel Core I7-9700K, una memoria RAM de 32 GB y un disco duro de estado sólido de 500 GB, pero faltó utilizar una tarjeta gráfica (GPU) la cual es más importante que la CPU a la hora de realizar los entrenamientos y procesamientos. Si se hubiera contado con una GPU de al menos 4 GB, los tiempos se habrían reducido considerablemente, pero para esto se requiere una inversión mayor. A pesar de todo, existen algunas herramientas que pueden facilitar este proceso, pero solo para realizar pruebas y entrenamientos que requieran poco tiempo. Una de estas herramientas es Google Colab donde se le permite a cualquier persona hacer uso de computadores propios de Google equipados con la última tecnología, pero el problema es que solo se pueden utilizar por un tiempo limitado y el almacenamiento es el que se tenga disponible en la cuenta de Google Drive, por eso es bueno para probar algo pequeño, ya que la mayoría de entrenamientos pueden alcanzar días o semanas y esta herramienta no lo permite, pero sin dudas brinda una ayuda muy buena en este tipo de trabajos que no requieren tiempos extremos de entrenamiento.

En el proyecto presentado se obtuvieron resultados muy buenos, pero siempre se puede mejorar el trabajo realizado. En primer lugar, a la hora de hablar acerca de la súper resolución, se podría implementar una nueva metodología que involucre los tres métodos utilizados en uno solo, es decir, mejorar la resolución de una imagen de entrada haciendo uso, en primer lugar de ESRGAN para obtener mejores texturas, luego implementar la interpolación lineal modificada para brindar muchos más detalles y aumentar la nitidez de la imagen y, finalmente, como esta imagen va a tener una calidad muy alta, utilizar esa imagen como la imagen objetivo con la cual entrenar la red PIX2PIX, de tal forma que la red aprenderá los patrones de una imagen con mucha mejor información, lo cual, podría brindar una herramienta útil y con tiempos de ejecución menores.

Otra forma de poder mejorar el trabajo realizado en cuanto a súper resolución es utilizar un banco de imágenes aun mayor, ya que, como se menciono a lo largo del trabajo, se utilizaron 196 imágenes de la misma escena, pero si se utilizaran mas imágenes, sin dudas la imagen resultante seria aun mejor. También se podría realizar la combinación con otros métodos en donde se implemente la interpolación lineal modificada y luego aplicar una red neuronal especializada en súper resolución distinta a ESRGAN, como lo son SRGAN, SRCNN o VDSR.

En cuanto a la zonificación los resultados también fueron buenos, pero en una aplicación que requiera la mejor precisión posible, es preferible utilizar una red mas precisa. Para ello se pueden tomar varios caminos: El primero es mejorar la red PIX2PIX de segmentación, ya que se utilizaron 250 imágenes para el entrenamiento y la cantidad de épocas fue de 384 utilizando el criterio de que el

error no variara durante 5 épocas consecutivas. Pero, si se utiliza un banco que contenga más de 1000 o 10000 imágenes y se utiliza un criterio de error en donde este se acerque lo mayor posible a 0%, sin dudas se obtendrían resultados mucho mejores teniendo en cuenta los tiempos reportados en el entrenamiento.

Otra vía para mejorar la zonificación es utilizar otras redes neuronales especializadas en segmentación como lo son AlexNet, GoogleNet o ResNet. Estas redes han sido las que menor error han obtenido en la segmentación, ya que son redes de mayor dimensión paramétrica que requieren un entrenamiento con mayor volumen de información y tiempo de procesamiento amplio, por tales motivos, son redes que no son dadas para todo público, pues son desarrolladas por las empresas más grandes del mundo en tecnología. Pero existen versiones "lite" abiertas al publico las cuales pueden ser utilizadas y modificadas por cualquier persona, por lo que, si se realiza el mismo procedimiento explicado en este trabajo sobre una de esas redes, es posible que se obtenga mejor precisión en la segmentación.

Por último, cabe destacar que todo el trabajo presentado se enfocó en agricultura de precisión, pero en realidad estas técnicas mostradas pueden ser implementadas en muchos campos de diversas áreas. Por ejemplo, el campo de la medicina esta acogiendo en gran manera las técnicas de deep learning para detectar y predecir enfermedades, utilizando técnicas muy similares a las presentadas en este trabajo, ya que las imágenes médicas que se obtienen de rayos X o resonancias magnéticas, son de bajas resolución y se hace obligatorio mejorar la resolución para ver con detalle la imagen. Pero la segmentación también toma un papel importante, pues cuando se trata de detectar anomalías en el cuerpo, estas técnicas lo pueden hacer rápidamente y con una precisión casi perfecta, lo cual puede ayudar a combatir ese problema de forma inmediata.

REFERENCIAS

Anika, L. (2015, 12). Zoning as Taxidermy: Neighborhood Conservation Districts and the Regulation of Aesthetics. Indiana Law Journal.

Anwar, A. (2019, 6). Difference between AlexNet, VGGNet, ResNet, and Inception. Recuperado de: https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96

ArcGIS. (2020). Semantic segmentation and how U-net works? Recuperado de: https://developers.arcgis.com/python/guide/how-unet-works/

Atkins, B. (2006). Sharpness: What is it and How it is Measured. Recuperado de: https://www.imatest.com/docs/sharpness/#:~:text=Image%20sharpness%20can%20be%20measured,distance%3B%20see%20Figure%203

Bedoya, O. (s.f.). Transformaciones geométricas. GeoGebra. Recuperado de: https://www.geogebra.org/m/gWSKktdT

Bogado, V. (2018, 8). Competencia en Kaggle — Google AI Open Images — Detección de objetos. Recuperado de: https://medium.com/@vanessabogado/competencia-en-kaggle-google-ai-open-images-detecci%C3%B3n-de-objetos-4ffaa389cb1c

Colombia, D. N. (2014, 10). Regionalización Presupuesto de Inversión en Bogotá D.C. 2015. Recuperado de: https://colaboracion.dnp.gov.co/CDT/Inversiones%20y%20finanzas%20pblicas/Bogot%C3%A1%2015-Ajustada.pdf

Crispí, J. (2019, 6). Introducción al deep learning parte 2: Redes Neuronales Convolucionales. [Imagen]. Recuperado de: https://medium.com/@jcrispis56/introducci%C3%B3n-al-deep-learning-parte-2-redes-neuronales-convolucionales-f743266d22a0

Diccionario de la Real Academia Española. (s.f.). Difuminación. Real Academia de la Lengua Española. Recuperado de: https://dle.rae.es/difuminar

Durdu, A. (2019, 4). Real-Time Diseases Detection of Grape and Grape Leaves using Faster R-CNN and SSD Mobile Net Architectures. Recuperado de: https://www.researchgate.net/figure/High-level-diagram-of-Faster-R-CNN-16-for-generic-object-detection-2-Inception-v2-The_fig3_334987612

Elad, M., Rubner Y. (s.f.). Super Resolution from Multiple Images. [Gráfico]. Recuperado de: http://toodle.cs.huji.ac.il/cs14/file.php/67735/SuperResolution.pdf

Elad, M., Protter, M. (s.f.). Super-Resolution with Probabilistic Motion Estimation. Semantics Scholar. Recuperado de: https://pdfs.semanticscholar.org/096d/2f557337e39eb250af8aea832bb31fbf519e.pdf

FotoNostra. (2017). El ruido en las fotografías. Recuperado de: https://www.fotonostra.com/fotografia/elruido.htm

Gupta, R. (2020, 6). Image Processing, Computer Vision and Deep Learning for Aerial and Satellite Images. Recuperado de: https://www.linkedin.com/smart-links/AQEXlbiNVCtvUw

Herrera, P. (2012). Construcción de imágenes de súper resolución usando un método probabilista. Instituto Nacional de Astronomía Óptica y Electrónica. Recuperado de: https://ccc.inaoep.mx/~esucar/Clases-mgp/Proyectos/finalPavelHerrera.pdf

Isola, P., Zhu, J., Zhou, T., Efros, A. (2018, 11). Image-to-Image Translation with Conditional Adversarial Networks. Recuperado de: https://arxiv.org/pdf/1611.07004.pdf

Jalife, A., Calderón, G., Fierro, A., Nakano, M (2017). Clasificación de Imágenes Urbanas Aéreas: Comparación entre Descriptores de Bajo Nivel y Aprendizaje Profundo. Recuperado de: https://scielo.conicyt.cl/scielo.php?script=sci_arttext&pid=S0718-07642017000300021

Kopriva, I., Ju, W., Zhang, B., Xiang, D. (2016). Single-channel Sparse Nonnegative Blind Source Separation Method for Automatic 3D Delineation of Lung Tumor in PET Images. Recuperado de: https://drive.google.com/file/d/0B2C7oTadbXbjdjdTaUI0S29VVVU/vie

Kawulok, M., Benecki, P. (2019). Deep Learning for Multiple-Image Super-Resolution. IEEE Explore. Recuperado de: https://ieeexplore.ieee.org/document/8884136

Lin, T., Patterson, G., Ronchi, M., Cui, Y., Belongie, S., Bourdev. (2015, 4). COCO (Common Objects in Context). Recuperado de: https://arxiv.org/pdf/1405.0312.pdf

Liu, P. (2020, 4). Single Stage Instance Segmentation — A Review. [Imagen]. Recuperado de: https://towardsdatascience.com/single-stage-instance-segmentation-a-review-1eeb66e0cc49

Lowe, D. (2004, 11). Distinctive Image Features from Scale-Invariant Keypoints. Recuperado de: https://link.springer.com/article/10.1023/B:VISI.0000029664.99615.94

Masayuki, T., Okutomi, M. (2007, 6). A fast algorithm for reconstruction-based super resolution and evaluation of its accuracy. Research Gate. Recuperado de: https://www.researchgate.net/publication/220463898_A_fast_algorithm_for_reconstruction-based_superresolution_and_evaluation_of_its_accuracy

MathWorks, Inc. (2018). Segmentación semántica y tres cosas que es necesario saber. MathWorks. Recuperado de: https://la.mathworks.com/solutions/image-video-processing/semantic-segmentation.html?s_tid=srchtitle

MathWorks, Inc. (2019). Súper resolución de una sola imagen mediante el aprendizaje profundo. MathWorks. Recuperado de: https://la.mathworks.com/help/images/single-image-super-resolution-using-deep-learning.html

MathWorks, Inc. (2019). Análisis de texturas. MathWorks. Recuperado de: https://la.mathworks.com/help/images/texture-analysis-1.html#:~:text=El%20an%C3%A1lisis%20de%20texturas%20intenta,intensidad%2C%20o%20niveles%20de%20gris.

MathWorks, Inc. (s.f.). Using RANSAC for estimating geometric transforms in computer vision. MathWorks. Recuperado de: https://la.mathworks.com/discovery/ransac.html

Metz, C. (1978). Basic principles of ROC analysis. In Seminars in nuclear medicine.

Milanfar, P. (2014). Super Resolution Imaging. Estados Unidos. Taylor and Francis Group.

Mlnsa, P., Rodríguez, J. (2009). The Essential Guide to Image Processing. Estados Unidos. Academic Press.

Mwiti, D. (2019. 7). A 2019 Guide to Semantic Segmentation. Recuperado de: https://heartbeat.fritz.ai/a-2019-guide-to-semantic-segmentation-ca8242f5a7fc

Nakamura, S., Nakaura, T. (2016, 8). Efficacy of the projection onto convex sets (POCS) algorithm at Gd-EOB-DTPA-enhanced hepatobiliary-phase hepatic MRI. National Center for Biotechnology Information. Recuperado de: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4978650/

Noor, M., Rosni, A. (2020). The evolution of UAVs applications in urban planning. Coordinates. Recuperado de https://www.researchgate.net/publication/324028974_The_evolution_of_UAVs_applications_in_urban_planning.pdf

Ren, S., He. K., Girshick, R., Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems.

Ronneberger, O., Fischer, P., Brox, T. (2015, 5). U-Net: Convolutional Networks for Biomedical Image Segmentation. [Imagen]. Recuperado de: https://arxiv.org/abs/1505.04597

Shaw, P., Rawlins, D. (1991, 8). The point-spread function of a confocal microscope: its measurement and use in deconvolution of 3-D data. [Imagen]. Recuperado de: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2818.1991.tb03168.x

Sinha, V. (2020, 5). ESRGAN: Enhanced Super Resolution GAN. [Imagen]. Recuperado de: https://medium.com/analytics-vidhya/esrgan-enhanced-super-resolution-gan-96a28821634

Santana, C. (2019, 9). Generar flores realistas con la arquitectura PIX2PIX. [Video]. Recuperado de: https://www.youtube.com/watch?v=YsrMGcgfETY

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z. (2015). Rethinking the Inception Architecture for Computer Vision. Cornell University.

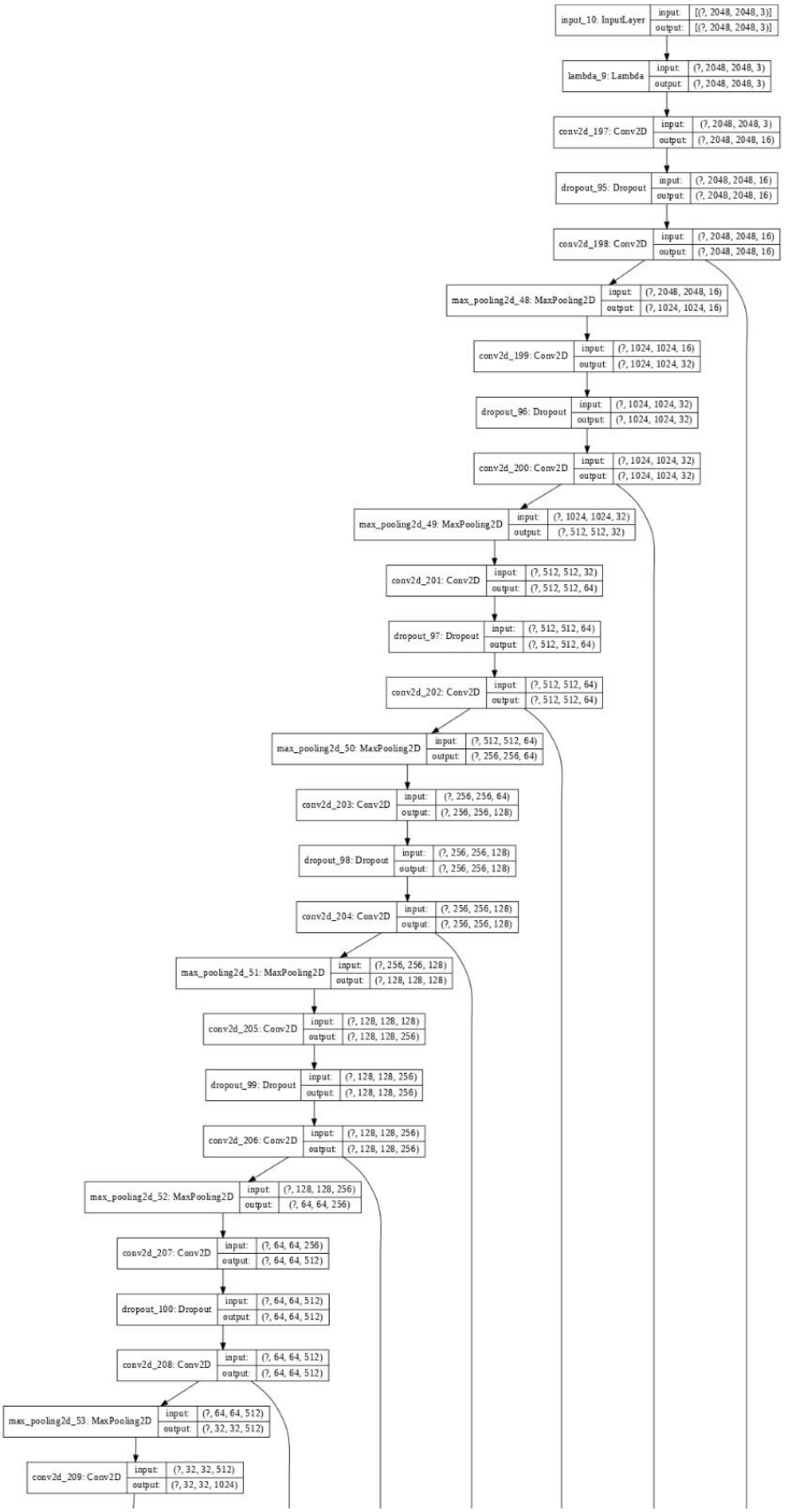
Tipping, M., Bishop, C. (2002). Bayesian Image Super-Resolution. Microsoft Research. Recuperado de: https://papers.nips.cc/paper/2315-bayesian-image-super-resolution.pdf

Wang, X., Yu, K. (2018, 9). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. Recuperado de: https://arxiv.org/abs/1809.00219

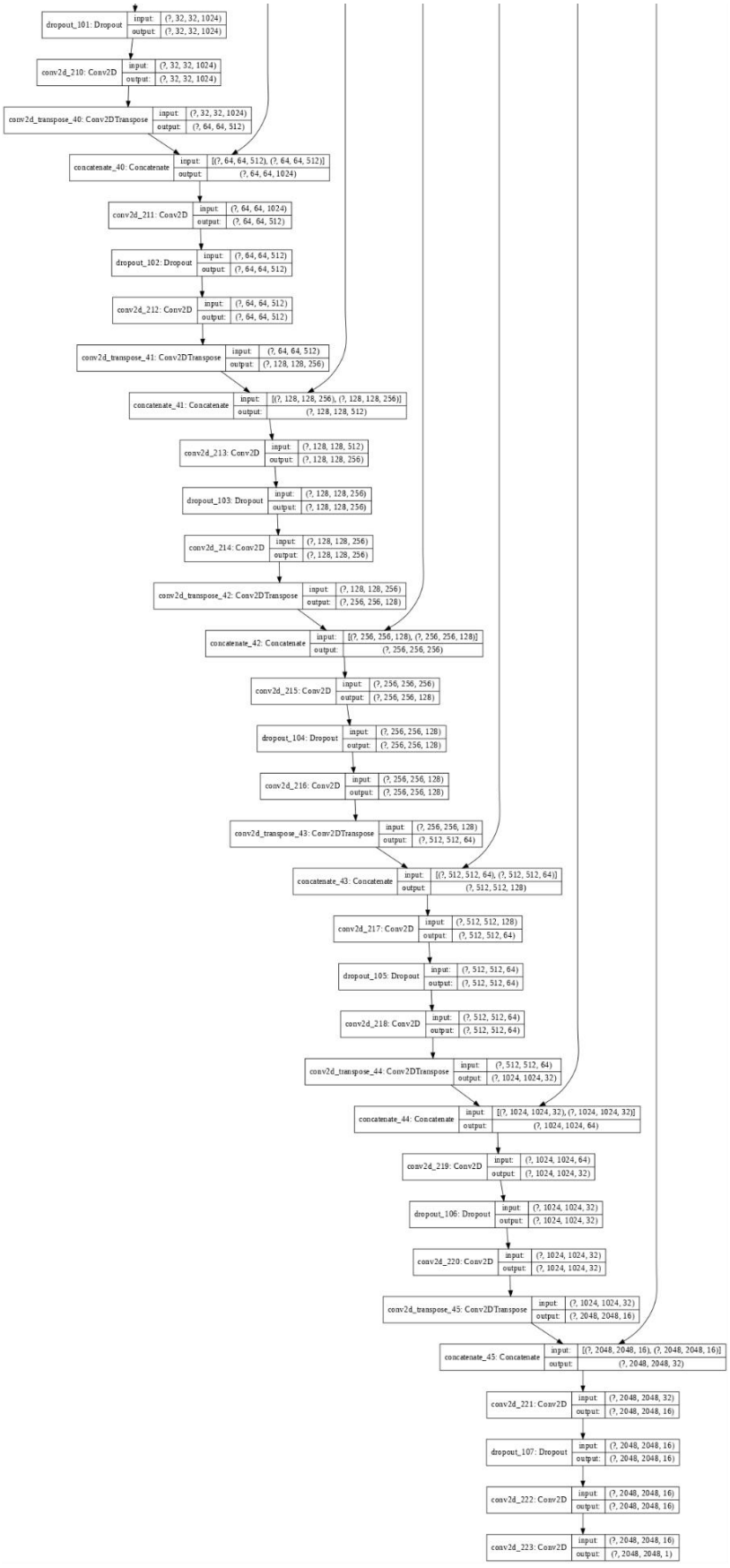
Zafra, D. (2020). La nitidez en fotografía – Qué afecta a la nitidez de una imagen. Recuperado de: https://capturetheatlas.com/es/nitidez-en-fotografia/

ANEXOS

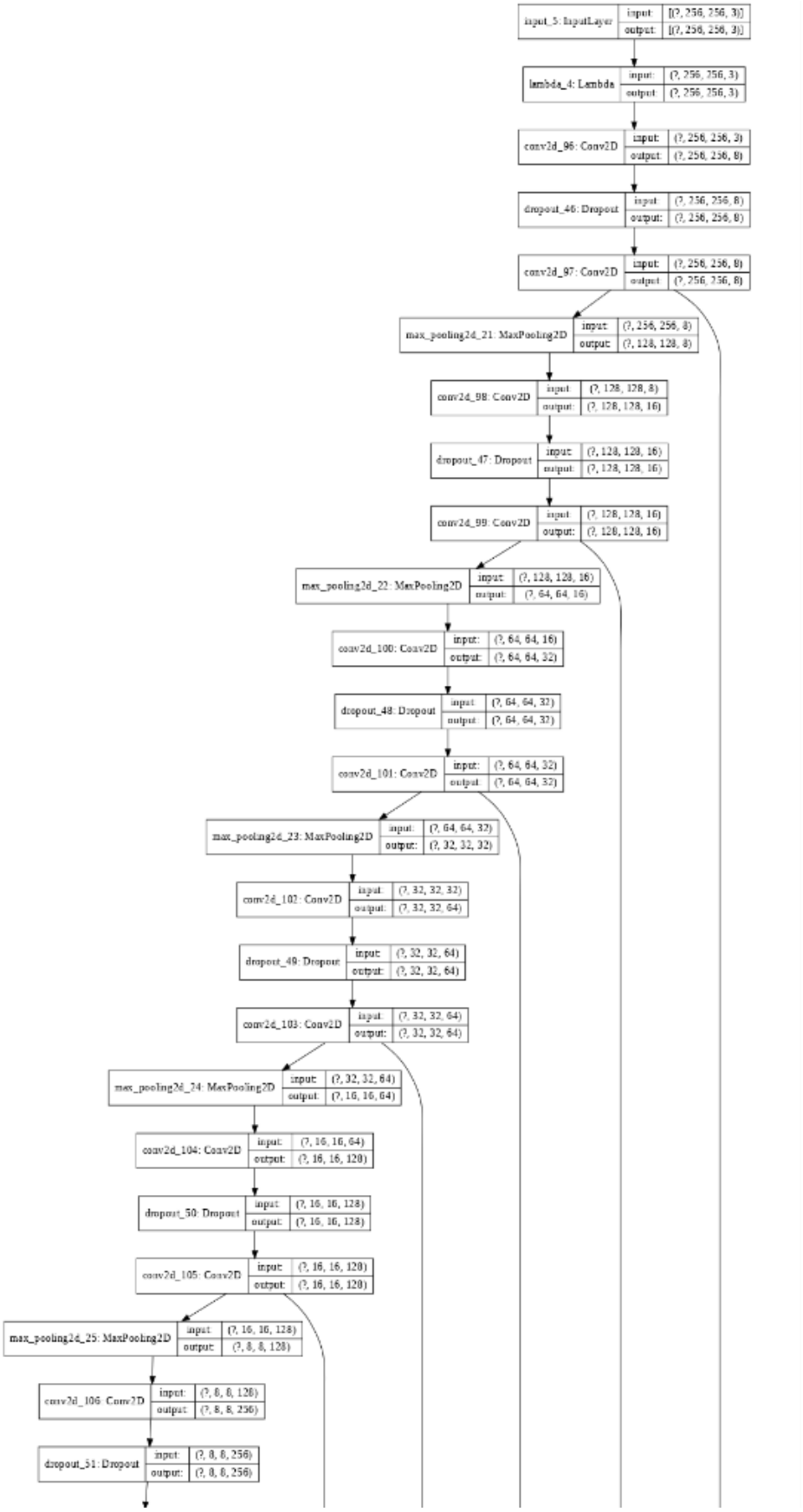
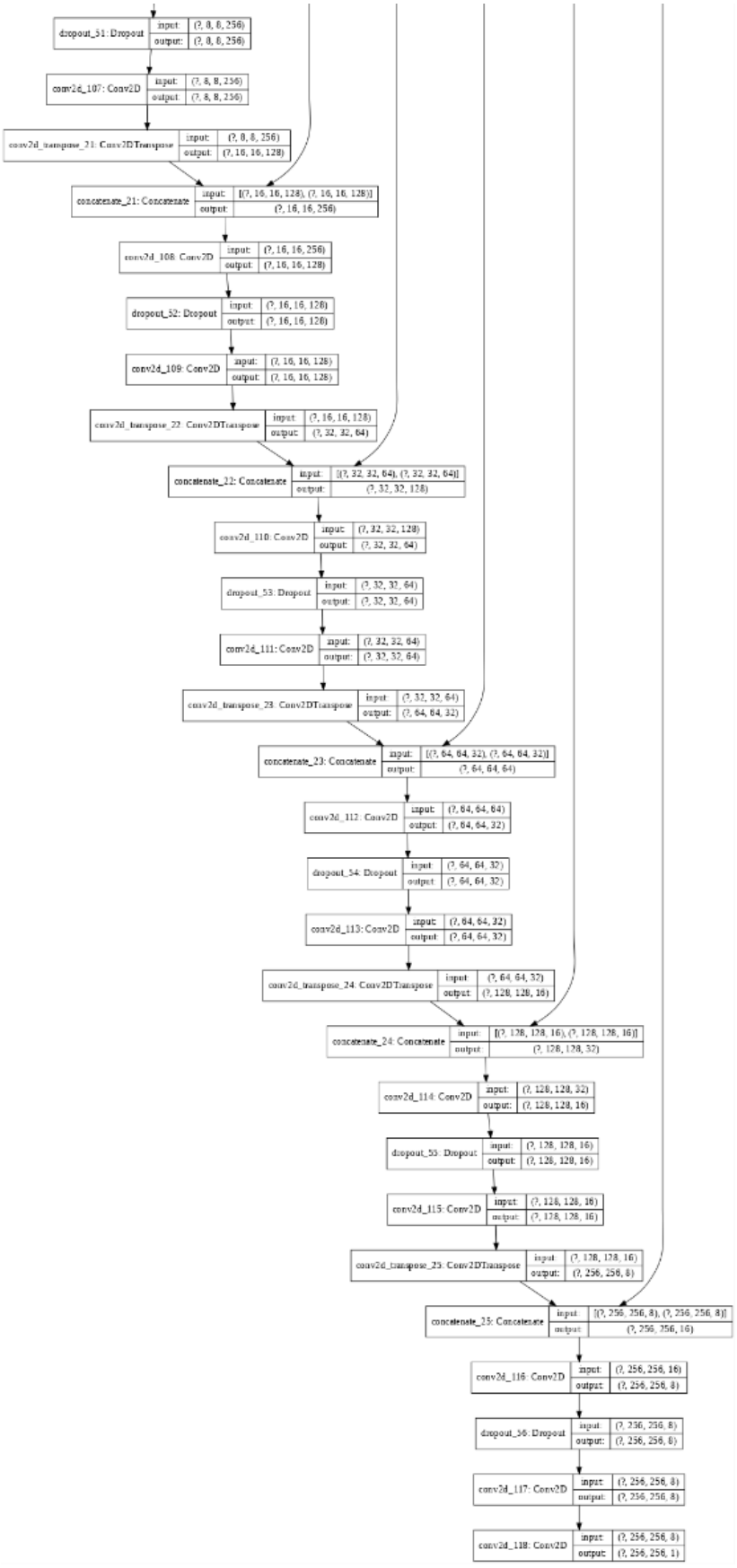
ANEXO 1: Parte 1 de la arquitectura U-Net modificada para súper resolución.



ANEXO 2: Parte 2 de la arquitectura U-Net modificada para súper resolución.



ANEXO 3: Parte 1 de la arquitectura U-Net modificada para zonificación.



ANEXO 4: Parte 2 de la arquitectura U-Net modificada para zonificación.