Multispectral Earth Simulator

Guillermo Cid Munuera

Resumen– Durante los últimos años en el campo de la visión por computador, sin duda, la tecnología de Deep Learning más utilizada son las redes generativas antagónicas más conocidas como GANs. Estas son capaces de generar imágenes prácticamente realistas que engañan al ojo humano. En este proyecto pretendemos experimentar con esta tecnología, para implementar un modelo capaz de convertir una imagen aérea de muy baja resolución, proveniente de un satélite, a una imagen con una resolución de foto ortográfica de alta calidad, típica de avión. Se ha implementado un sistema para la obtención de datos usados en los distintos experimentos realizados, en los que se han conseguido resultados satisfactorios.

Palabras clave– Imagen aerea, Super resolución, Deep Learning, GANs

Abstract– In the last few years, in the field of computer vision, without a doubt the Deep Learning technology with more importance are the generative adversarial networks better known as GANs, capable of generating practically realistic images that fool the human eye. In this project we try to experiment with this innovative technology, to implement a model capable of converting a very low resolution image from a satellite, to an image with a resolution of an orthographic photo of high quality, typical of airplanes. A system has been implemented to obtain the data used in the different experiments performed, in which satisfactory results have been achieved.

Keywords– Aerial image, Super resolution, Deep Learning, GANs

1 Introducción

Actualmente los modelos de Deep Learning orientados a imagen aérea, en su mayoría son implementaciones dedicadas a la clasificación. En este trabajo hemos implementado una tecnología con un enfoque distinto, basándonos en recientes técnicas de super resolución y de recreación de imágenes. Hemos desarollado un modelo capaz de generar una imagen de alta resolución, a escala de avión, a partir de una de muy baja resolución, a escala de satélite.

En un trabajo previo a este, se realizó un estudio e implementación de estimación de alturas en imágenes aéreas mediante Deep Learning. Los datos usados para el entrenamiento del modelo eran imágenes de drones totalmente exclusivas y de zonas concretas, con una resolución de aproximadamente 6 cm por píxel. En este experimento previo se llegó a la conclusión que si disponíamos de imágenes aéreas de alta resolución, mediante un modelo de Deep Learning, se puede conseguir un mapa de alturas de la propia imagen. Esto forma parte del objetivo global este proyecto previo y el desarrollado en este trabajo, el cual consiste en la simulación de espacios abiertos en 3D a vista de pájaro.

En este trabajo se propone un estudio y una alternativa mediante el Deep Learning para seguir en la línea de trabajo descrita. Para ello, entrenamos un modelo capaz de convertir una imagen de satélite con una resolución aproximada de 10 m por píxel a una de alta resolución, alrededor de 25 cm por píxel. Existen numerosos servicios online que ofrecen imágenes aéreas. En nuestro caso, las imágenes de baja resolución son del satélite Sentinel-2, perteneciente a la Agencia Espacial Europea (ESA) [1]. Este recolecta datos de acceso abierto de imágenes multiespectrales con 13 bandas, con una cobertura de una gran parte del globo y con actualización de datos cada 5 días. En el caso de las imágenes de alta resolución se han obtenido los datos del Instituto Cartográfico y Geológico de Cataluña (ICGC) [3], este nos ofrece ortofotos de 25 cm/pix anuales de la región de Cataluña desde el año 2007 al 2016.

Para conseguir este reto se debe tener en cuenta la magnitud del problema, nos hemos planteado aumentos de 32 o 64 veces la resolución de una imagen, esto nos lleva a aplicar técnicas de recreación, basadas en GANs, combinado con la superresolución de imágenes.

En primer lugar, hemos realizado un estudio de las redes que han sido utilizadas para resolver problemas similares al nuestro, estudiando así los modelos más convenientes. Antes de empezar el aprendizaje del modelo, hemos desa-

• E-mail de contacte: guillermo.cidm@e-campus.uab.cat
• Menció realitzada: Computació
• Treball tutoritzat per: Felipe Lumbreras Ruiz
• Curs 2019/20

rrollado e implementado un sistema automático para obtener los datos necesarios de nuestro entrenamiento, para ello se han utilizado APIs y servicios web para su recolección. Una vez preparado el dataset hemos realizado distintos experimentos en los cuales se han testeado los límites de las arquitecturas, para así, acabar llegando al objetivo final del proyecto con resultados destacables.

Hasta ahora hemos expuesto las razones de nuestro proyecto y los puntos clave del trabajo. A continuación os listaremos los principales objetivos del proyecto en cuestión:

- Implementar un software capaz de recolectar datos de las imágenes de Sentinel-2 y que realice la corrección atmosférica pertinente.
- Implementar un software capaz de recolectar ortofotos de alta calidad del Cartográfico de Cataluña a partir de unas coordenadas geográficas.
- Generar un dataset combinando las imágenes de alta calidad junto con las de baja calidad, descartando imágenes que supongan un problema para nuestro modelo.
- Implementar un modelo capaz de generar imágenes de alta calidad de 25 cm/pix a partir de imágenes de 10 m/pix. Combinando las arquitecturas de GANs con las de superresolución.
- Evaluar los resultados obtenidos, usando las métricas más adecuadas, para escoger el mejor modelo.

2 Estado del arte

Como hemos comentado en la introducción este modelo a implementar tiene dos técnicas remarcables planteadas: la de recreación y la de superresolución. Nos referimos a recreación, o en algunos casos nombrada como alucinación, como la generación de imágenes totalmente sintéticas, que pretenden ser realistas. Para conseguir esta recreación es necesario el uso de redes generativas antagónicas (GANs) [7]. Esta tecnología es muy reciente y está en expansión. Dentro de esta tecnología existen distintos campos destacables como la síntesis de texto a imagen (Text-to-Image synthesis) [12], que consiste en la generación de imágenes de alta calidad a partir de descripciones de texto. En nuestro caso está más cerca de la reciente tecnología Image-to-Image-Translation capaz de generar una imagen a partir de otra de un campo visual distinto, el caso más novedoso es el de pix2pix [8] donde se han conseguido modelos capaces de corregir pinturas dañadas, poner color a una imagen en blanco y negro, o generar imágenes foto realistas a partir de bocetos. El caso más reciente y a la vez mediático es el de la generación de caras humanas totalmente falsas en alta resolución a partir de un vector de valores, en este caso se usa una GAN progresiva PGGAN [9] que va aumentado la resolución de la imagen de entrada a la vez que la genera.

Por otro lado, el proceso de superresolución es aquel que recobra una imagen a alta resolución (HR) a partir de una de baja resolución (LR). Es una técnica muy importante de la visión por computador. Se aplica sobre todo en campos como la imagen médica, la vigilancia o en la seguridad. En los últimos años se han aplicado técnicas de Deep Learning en superresolución consiguiendo mejorar los métodos anteriores, usando arquitecturas simples como la CNN y otros enfoques más complejos usando GANs [11].

3 Desarrollo

3.1. Datos obtenidos

Para realizar los distintos experimentos se requiere un mecanismo que nos permita obtener una gran cantidad de datos de forma sistematizada. Para la obtención de datos de baja resolución que suponen nuestra entrada del modelo, como ya hemos comentado, hemos escogido recolectar datos del satélite Sentinel-2, este nos proporciona en sus imágenes distintos espectros como el infrarrojo cercano, el infrarrojo de onda corta y el espectro electromagnético además de ofrecer distintas resoluciones de 10 m, 20 m y 60 m. En nuestro caso nos interesa el espectro visible que corresponderían a las bandas B02 (Azul), B03 (Verde) y B04 (Rojo) que tienen una resolución de 10 m. Existen distintos servicios/APIs de pago o gratuitos para obtener estos datos, en los de pago los datos se obtienen ya procesados, en cambio, en los servicios gratuitos se obtienen sin procesar, por lo tanto, es necesario realizar correcciones a estos datos.

Para la obtención de las imágenes de alta calidad, recurriremos a servicios cartográficos de nuestra región donde se pueden obtener fotografías ortográficas de avión con resoluciones que van desde 1 metro por píxel hasta 25 centímetros por píxel, estos servicios cartográficos incluyen un servició web de mapas (WMS). A través de una API, implementaremos un software capaz de adquirir estos datos.

3.1.1. SentinelSat

SentinelSat [2] es una API que accede al Copernicus Open Access Hub [4] de la ESA , este nos proporciona datos gratuitos de los satélites Sentinel-1, Sentinel-2, Sentinel-3 i Sentinel-5P. A través de la API hemos desarrollado un software que dado unos parámetros específicos obtiene los datos de la zona concreta requerida en una fecha determinada. Estos son los parámetros más destacados:

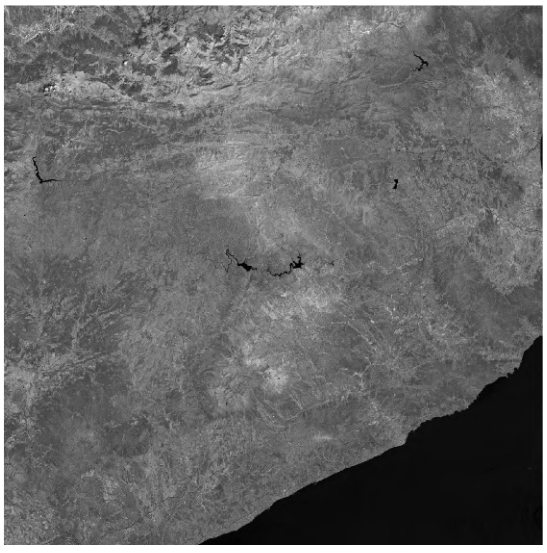


Fig. 1: Banda 4 de Sentinel-2 perteneciente a la zona de Catalunya.

TABLA 1: INFORMACIÓN BANDAS SENTINEL-2.

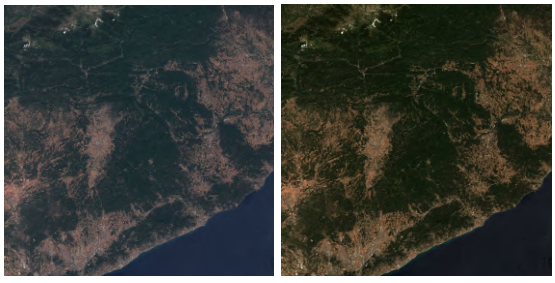
Banda	Resolución	λ central	Descripción
B1	60 m	443 nm	Ultra azul
B2	10 m	490 nm	Azul
B3	10 m	560 nm	Verde
B4	10 m	665 nm	Rojo
B5	20 m	705 nm	Visible e Infrarrojo Cercano (VNIR)
B6	20 m	740 nm	Visible e Infrarrojo Cercano (VNIR)
B7	20 m	783 nm	Visible e Infrarrojo Cercano (VNIR)
B8	10 m	842 nm	Visible e Infrarrojo Cercano (VNIR)
B8a	20 m	865 nm	Rojo de borde (RedEdge)
B9	60 m	940 nm	Vapor de agua
B10	60 m	1375 nm	Cirrus
B11	20 m	1610 nm	Onda Corta Infrarroja (SWIR)
B12	20 m	2190 nm	Onda Corta Infrarroja (SWIR)

- Región: zonas con de un tamaño de 100×100 km^2.
- Fecha: rango de fechas deseado.
- Plataforma: satélite de donde se obtienen los datos, en nuestro caso el Sentinel-2.
- Porcentaje de nubes: porcentaje de nubes máximo aceptado en la región obtenida. Hemos usado un 10 %.

En nuestro caso hemos escogido fechas del verano de 2016 así podemos hacer coincidir estas imágenes con las fechas en que el ICGC realizó las orto fotos de alta calidad. SentinelSat nos proporciona un sistema de ficheros dónde se encuentra información geográfica y las correspondientes 13 bandas del Sentinel-2. Estas imágenes están codificadas en un formato J2P de 12 bits y tienen un procesado atmosférico pobre (L1C). En la Fig. 1 podéis observar una de los mapas perteneciente a parte de la zona de Cataluña, este en concreto pertenece a la banda 8, que es infrarroja.

3.1.2. Generación del dataset

El que las distintas bandas del Sentinel-2 tengan un procesado de tipo L1C supone una menor calidad de imagen. Este procesado que requiere de un software llamado Sen2Cor para conseguir una mejor corrección atmosférica (L2A), este software realiza una corrección del terreno y de nubes que distorsionan la calidad de la imagen. Así pasamos de una reflectancia TOA (Top-of-atmosphere) a una BOA (Bottom-of-atmosphere). TOA representa los valores de reflectancia crudos medidos desde el espacio. En cambio, BOA representa también la reflectancia, en este caso, de las áreas de la superficie terrestre. Podemos observar la diferencia entre los dos procesados en la Fig. 2.



L1C L2A (corregida)

Fig. 2: Corrección atmosférica con software Sen2Cor.

Una vez disponemos de todas las bandas corregidas, creamos un archivo virtual juntando las bandas B04, B03 y B02 en este preciso orden para conseguir la True-Color-Image (TCI). Y posteriormente, aplicamos una conversión de 12 bits a 8 para construir la imagen en RGB. La ESA también nos proporciona un clasificador de la región Scene-Classification-Map (SCL) por tipo de zona, como por ejemplo bosque, urbano, mar o cultivo. Este clasificador se usara para la fase de generación de los datasets para la red, ya que algunos de ellos están compuestos por una zona en concreto.

Una vez los datos están procesados obtenemos una imagen de grandes dimensiones, de esta generaremos los recortes necesarios para la fase de entrenamiento, cada uno de ellos con información sobre sus coordenadas geográficas.

Hasta ahora se han conseguido las imágenes de satélite (10 m/pix), ahora se necesitan las imágenes de avión (25 cm/pix). Estas son las de alta calidad que queremos como salida ideal de nuestro modelo. Como ya hemos comentado obtendremos estos datos del ICGC, este dispone de un servicio web de mapas (WMS) en el que se pueden solicitar datos de orto fotos de avión de distintos años. En nuestro caso recolectaremos las imágenes que correspondan a las obtenidas de baja resolución. A través de la librería de Python OWSLib [5] se realiza la petición correspondiente pudiendo obtener la imagen deseada a partir de unas coordenadas geográficas, dichas coordenadas las extraemos de los recortes realizados a la imagen del satélite Sentinel-2.

Algunas de las imágenes del conjunto de datos del ICGC hemos visualizado que tienen un cierto grado de distorsión, estas deben ser descartadas de la fase de entrenamiento de nuestra red ya que puede repercutir en la calidad de los resultados, para descartar que este tipo de datos entren en el conjunto de entrenamiento de la red se ha aplicado una técnica que asume que el ruido de la imagen sigue una distribución Gaussiana y estima el valor de la desviación estándar de esta gaussiana. Las imágenes de mejor calidad con un mayor conjunto de contornos y mayor definida serán las que tienen una valor estimado más elevado, y las imágenes más borrosas con un valor menor de contornos tendrán un valor de la desviación estándar menor, en la imagen se puede ver la diferencia de calidad entre imágenes de la misma región. En la Fig.3 se comparan dos imágenes, una con una mayor distorsión y la otra más nítida.

Fig. 3: Comparación entre dos imágenes del ICGC, una mas emborronada (izquierda) y la otra con mayor nitidez(derecha).

Por ahora podemos acceder a dos servicios, SentinelSat y ICGC, para la recolección de los datos necesarios. Hemos implementado un software encargado de generar el conjun-

to de datasets para cada tipo de experimento realizado. Esta implementación consta de una serie de pasos:

1. Obtención de la TCI y el SCL: A este programa se le debe primero proporcionar un usuario y contraseña del Copernicus Open Acces Hub, y las coordenadas geográficas de la zona deseada, en nuestro caso se trata de una parte de la provincia de Barcelona en concreto las coordenadas (Oeste: 400830, Norte: 4613500, Este: 428950, Sur: 4594560) según el sistema de coordenadas EPSG: 32631 - WGS 84. A partir de estos datos, obtenemos la TCI y el SCL de esta zona.

2. Generar recortes: A partir de las imágenes obtenidas en el paso anterior generamos recortes de estas dos imágenes, estos son de 32×32, suponen una resolución de 8 m/pix, estás siguen manteniendo su referencia geográfica.

3. Selección de datos: En el paso anterior obtenemos un total de 8100 imágenes, usamos el SCL para clasificar el tipo de imagen TCI según zona, en el caso de zona de bosque obtenemos un total de 3000 imágenes.

4. Descargar imágenes alta calidad: Una vez obtenidas las imágenes que nos interesan, realizamos peticiones al servicio del ICGC utilizando las coordenadas geográficas de cada una de las imágenes TCI, para así descargar las imágenes en alta calidad de 1024×1024 con una resolución de 25 cm/pix.

5. Generar datasets: A partir de las imágenes obtenidas, podemos generar los distintos datasets que usaremos en la fase experimental, aplicaremos la técnica de la validación cruzada para el conjunto de datos de entrada y groundtruth, dividiéndolos en 80 % para entrenamiento, 10 % para validación y 10 % para testeo. Por último, combinaremos estos datos para que la red los pueda trabajar.

3.2. Arquitectura utilizada

3.2.1. Superresolución

Para resolver el problema de superresolución usamos la arquitectura diseñada en [10]. CARN es un modelo basado en ResNet, pero los módulos residuales pasan a ser tener interconexiones de tipo local y global. Es decir, se utilizan módulos de cascada (Cascading modules). Los outputs de capas intermedias se usan como inputs de capas más avanzadas. Se puede ver en la Fig.4. Como es la estructura de cada bloque y del conjunto de la red. Esta arquitectura esta compuesta relativamente de un número bajo de bloques cascada y a diferencia de otras redes donde el aumento de tamaño de imagen se hace al inicio, CARN tiene un bloque final para realizar el “upsampling”.

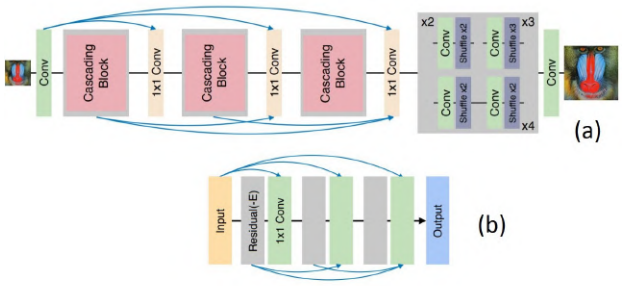


Fig. 4: Arquitectura de la CARN: (a) capas que la componen, (b) cascading block.

Estos factores comportan que la red sea ligera, por lo tanto, tiene menos parámetros y realiza menos operaciones que el resto de redes de su ámbito, sin que su rendimiento se vea afectado significativamente. Para el entrenamiento de la red usamos recortes de imágenes de 64×64, y aplicamos técnicas para aumentar el número de muestras de forma “online”. A cada imagen cargada que la red trabaja le realizamos un flip de izquierda a derecha y de arriba a abajo aleatorio, y entre 0 y 4 rotaciones de 90 grados. Con esto conseguimos aumentar el número de imágenes de entrada trabajadas por el modelo, como se puede ver en la Fig.5.

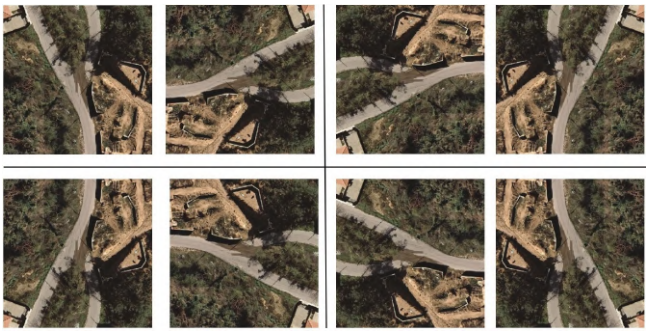


Fig. 5: Estrategia para aumentar datos de forma online en la fase de entrenamiento de superresolución.

3.2.2. GANs

Para conseguir la alucinación/recreación nos hemos fijado en las arquitecturas que ya han conseguido esta funcionalidad en otros ámbitos, estas arquitecturas son las Redes generativas antagónicas (Generative Adversarial Networks), más conocidas como GANs. Estas están compuestas por dos subredes que compiten entre ellas. Por un lado tenemos la red generadora, encargada de generar la imagen de salida, por lo tanto, esta tiene una arquitectura con capas totalmente convolucionales. Por otro lado, tenemos la red discriminadora, esta analiza el producto de la red generadora y determina si se ajusta a lo que está buscando. Estos conjuntos de redes permiten que la red generadora, al querer engañar a la otra red, produzca imágenes de salida que no existen, solo están basadas en la realidad para que la discriminadora sea engañada.

El forzar a que genere contenido irreal da pie a que las GANs tengan poca estabilidad. Es difícil crear un modelo que dé buenos resultados ya que es fácil que el propio modelo colapse Los datos naturales suelen tener concentraciones de muestras de datos similares, pero distintos en cada

tipo de imagen. Como el generador solo quiere engañar al discriminador puede llegar a producir muestras de un solo tipo de concentraciones de datos o patrón de datos.

Otra de las complicaciones de las GANs es la de saber cuándo se debe acabar la fase de entrenamiento. No podemos determinar cuando el modelo converge a partir de la función de "Loss" del generador y del discriminador. Por lo tanto, es difícil determinar cuando el generador está realizando imágenes de calidad y en muchos casos se debe recurrir a la visualización.

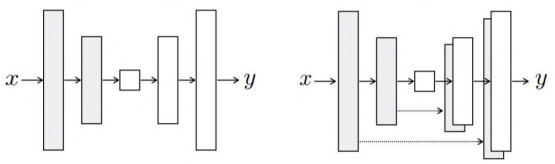


Fig. 6: Diferencias de arquitectura entre Unet (derecha) con un Encoder-Decoder (izquierda).

Nuestro modelo escogido para realizar el conjunto de pruebas es la pix2pix, esta realiza el proceso de Image-To-Image Translation, es decir, traslada una representación de una imagen a otro tipo de representación. El generador de este modelo usa una arquitectura totalmente convolucional basada en la Unet, en concreto la Unet256, que se basa en la arquitectura de un codificador-descodificador (Encoder-Decoder), pero esta arquitectura también tiene implementado una serie de conexiones entre las capas del Encoder y el Decoder. En la Fig.6 se puede visualizar dicha arquitectura como difiere de la del Encoder-Decoder. El discriminador usa un arquitectura de tipo PatchGAN la cual esta compuesta por cinco bloques convolucionales. En la fase de entrenamiento se minimiza la Loss entre la imagen generada y la original. Se puede usar tanto la regularización L1 como L2. Por lo tanto, el generador trata de minimizar este valor y, por otra parte, el discriminador trata de maximizarlo.

3.3. Fase experimental

Disponemos de una implementación que generara los distintos datasets necesarios para cada una de las fases experimentales realizadas. En la primera fase usamos la arquitectura pix2pix para realizar un primer experimento sencillo y comprobar su funcionamiento, los datos de entrada consisten en la imágenes de alta calidad de 25 cm/pix decimadas a 8 m/pix. Así podemos testear con los distintos hiperparámetros de la red. Continuamos con la fase en la que hemos querido juntar las dos arquitecturas (CARN y pix2pix) para combinar de distintas formas las técnicas de super resolución y de recreación, y comprobar que combinación funciona mejor, en este caso los datos de entrada siguen siendo imágenes de alta calidad decimadas. En la tercera fase hemos querido acercar nuestro modelo a datos de entrada más realistas. Para ello primero hemos realizado un entrenamiento con datos similares a los de Sentinel-2 para después realizar un finetunning, y segundo hemos entrenado con datos de Sentinel-2 como entrada.

3.3.1. Fase inicial

Una vez que disponemos de todo el conjunto de datos explicado en el apartado 3.1.2, el primer paso a realizar ha sido el de testear con la arquitectura de pix2pix realizando un conjunto de pruebas.

Se ha cogido un conjunto reducido, aproximadamente 5000 imágenes de la zona de la provincia de Barcelona, a una resolución de 25 cm/pix. Basandonos en trabajos similares al nuestro, a estas imágenes se les ha reducido su resolución con la técnica de interpolación por aproximación hasta resolución de 8 m/pix, resolución similar a la de Sentinel-2. En este primer entrenamiento de testeo, hemos podido sacar varias conclusiones. Los resultados han sido imágenes en que la red ha recreado algunos patrones propios de las imágenes reales como por ejemplo árboles, cultivos, caminos, etc. En cambio, otros patrones de zonas urbanas no los ha podido recrear. Esto es debido a que estos no son mayoritarios y, además, son más difíciles de generar correctamente. Al querer cumplir el conjunto de objetivos propuestos al inicio del proyecto, se realizarán los próximos experimentos con un conjunto de datos más concretos. Este conjunto excluye las zonas urbanas y se centra en zonas de bosque de la provincia de Barcelona. Estos datos deberán ser muy concretos y similares entre sí para evitar las complicaciones que pueden surgir al entrenar este tipo de red.

3.3.2. Fase intermedia

Dado los resultados de la fase anterior hemos decidido realizar una serie de experimentos para comprobar la magnitud de la arquitectura utilizada, y saber hasta que punto la red genera imágenes factibles para el ojo humano. Como ya hemos comentado en puntos anteriores queremos convertir una imagen de 32×32 y 8 m/pix a una de 1024×1024 y 25 cm/pix, es decir, un aumento de $\times32$. Realizaremos un total de 3 experimentos para poder sacar una serie de conclusiones, estos experimentos tendrán una primera fase de recreación o alucinación donde se usará la arquitectura comentada en el punto 3.2.2 y una segundo fase de superresolución, punto 3.2.1, si es necesaria. A continuación listamos los tres experimentos realizados:

1. Recrear imagen de 8 m/pix a 1 m/pix ($\times8$) y superresolución de 1 m/pix a 25 cm/pix ($\times4$).

2. Recrear imagen de 8 m/pix a 50 cm/pix ($\times16$) y superresolución de 50 cm/pix a 25 cm/pix ($\times2$).

3. Recrear imagen de 8 m/pix a 25 cm/pix sin fase de superresolución ($\times32$).

Con este experimento podremos llegar a observar el alcance que tienen estas arquitecturas y, a través de métricas específicas, explicadas en el apartado de resultados, hemos podido sacar conclusiones. Para el entrenamiento hemos generado un dataset el 80 % de las imágenes para la fase de entrenamiento, 10 % para validación y 10 % para testear. En primer lugar, hemos entrenado la pix2pix, y a partir de los resultados visuales hemos podido determinar cuando la red ha convergido. Por otro lado, hemos entrenado también el modelo CARN, para la fase de super resolución, hemos realizado el entrenamiento con dos tipos de aumento de resolución, el primero es un aumento de $\times2$, pasamos de una

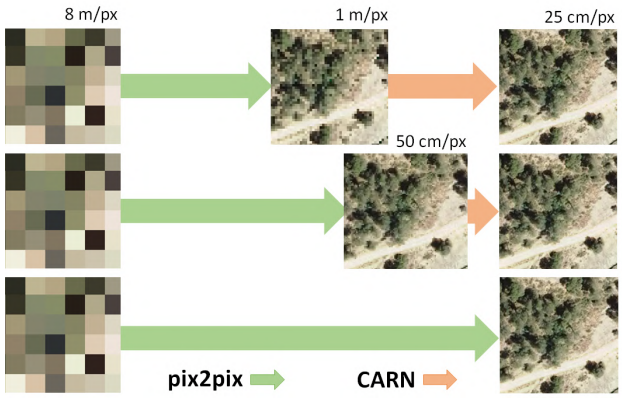


Fig. 7: Esquema seguido durante la fase intermedia, indicando la arquitectura utilizada en cada experimento y las resoluciones por píxel que se consiguen.

imagen con una resolución de 50 cm/pix a 25 cm/pix. El segundo tipo de entrenamiento ha sido de un aumento de $\times 4$, pasamos de una imagen de 1 m/pix a 25 cm/pix de resolución. Por último, las imágenes de test generadas con la pix2pix entrenada se utilizan como entrada de la CARN para aplicar la super resolución.

3.3.3. Fase Final

En esta última fase queremos ver si el modelo es capaz de recibir como entrada una imagen directamente de Sentinel-2 y generar una imágen de alta resolución, sabiendo que las de satélite contienen una menor información que las imágenes de entrada de los experimentos realizados hasta ahora.

Antes de entrenar el modelo con imágenes de Sentinel-2 como entrada, primero hemos entrenado con imágenes de avión procesadas, de tal manera, hemos implementado un método para degradar una imagen de avión y que esta se parezca lo mayor posible a las de Sentinel-2. Además, se ha modificado la implementación de la red neuronal pix2pix, se calcula la media y desviación estándar de cada mini-batch (en el caso de nuestra arquitectura los batches son de una única imagen), y antes de cargar la imagen a la red se normaliza este mini-batch a partir de este cálculo. Por lo tanto, cada imagen de entrada tiene unos valores de normalización distintos. Esto permite entrenar con las imágenes degradadas y posteriormente realizar un fine tunning con las imágenes de satélite. Finalmente, hemos realizado el entrenamiento con estas modificaciones de las imágenes de Sentinel-2 como entrada.

4 Resultados

En este apartado explicamos de un inició que métricas nos son más útiles para evaluar el conjunto de resultados. Más adelante exponemos tanto visualmente como numéricamente los resultados de cada una de las fases experimentales explicadas en el punto anterior.

4.1. Métricas de evaluación

Para ser consciente de la calidad de los resultados hemos escogido varios tipos de métricas para evaluar nuestros resultados. En problemas similares al nuestro es común usar el PSNR y el SSMI. “Peak signal to Noise Ratio” es una métrica que consiste en la inversa proporcional del logaritmo del error medio cuadrático (MSE) entre la imagen de salida y el groundtruth. Cuanto mayor es el PSNR, mejor resultado.

$$\text{PSNR} = 10 \log_{10} \frac{L^2}{MSE}. \quad (1)$$

La ecuación 1 es la fórmula del PSNR donde L es el valor máximo del píxel de la imagen en concreto, es decir, si la imagen es RGB de 8 bits el valor es 255. Esta métrica es realmente simple ya que solo cálcula el error por píxel, lo cual, en algunos casos no representa una medida adecuada a la percepción del ojo humano.

El SSIM es una métrica para medir la similaridad estructural entre imágenes, donde se compara entre la imagen de salida y el groundtruth la luminancia, el contraste y la estructura. Cuanto mayor es el SSMI, mejor resultado.

$$\text{SSMI}(i,j) = \frac{(2\mu_i\mu_j + C_1)(\sigma_{ij} + C_2)}{(\mu_i^2 + \mu_j^2 + C_1)(\sigma_i^2 + \sigma_j^2 + C_2)}, \quad (2)$$

donde la μ_i representa la media de la imagen, la σ_i es la desviación estandard y C1, C2 son constantes para evitar la inestabilidad.

Estas dos últimas métricas explicadas que evalúan el error por pixel y la estructura como bien hemos comentado, son un problema para evaluar de forma numérica las imágenes generadas por las GANs. Estas son imágenes generadas son totalmente nuevas, por lo tanto, contienen algo de distorsión y ruido, lo más probable es obtener un mal resultado de PSNR y SSMI aunque perceptiblemente, al ojo humano, el resultado sea aceptable. El FID es una métrica capaz de evaluar imágenes con resultados significativos y comparables para imágenes generadas con GANs, en el [13] se demuestra como la métrica es robusta a ruido como el gausiano, emborronamiento y “salt and pepper”.

$$\text{FID} = ||\mu_x - \mu_g||_2^2 + Tr(\sum x + \sum g - 2(\sum x \sum g)^{\frac{1}{2}}). \quad (3)$$

4.2. Evaluación de los resultados

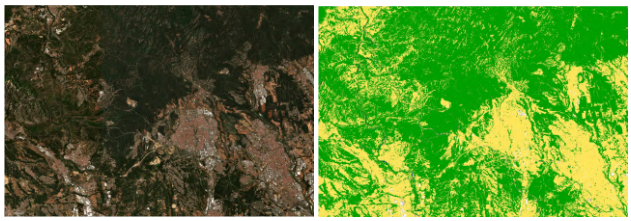
Una vez realizado todos los experimentos y justificado las métricas que se deben usar. Pasamos a mostrar los resultados del trabajo. Como ya hemos comentado en el punto 3.3.1, durante la primera fase de aprendizaje los resultados visuales no han sido satisfactorios. Al entrenar con imágenes mezclando zonas geográficas muy distintas como es la zona urbana y la de bosque, la red ha aprendido a obviar algunos objetos de la imagen. Esto es debido a que la red generadora de la GAN es capaz de engañar a la discriminadora repitiendo patrones que son usuales en el conjunto del dataset ante una situación poco usual para el modelo entrenado.

Como este dataset es mayoritariamente de zonas de bosque, es con este tipo de paisaje al que la red se adapta. En la Fig.8, podemos ver un claro ejemplo de como la red no ha convergido de forma correcta. Podemos observar como en una zona donde se encuentra una carretera, la imagen de salida si que genera un patrón similar a la imagen original, pero no se parece en nada a la estructura real. Esto es un pequeño ejemplo de muchos casos similares, sobretodo con edificios y estructuras que no son de la naturaleza. Estas al

(a) (b)

Fig. 8: Comparación entre imágenes generadas por la red (a) y la orginal (b) durante la primera fase de experimentos.

tener formas más complejas su patrón es más difícil de imitar, la red no las acaba de interpretar bien y se suele ver esa zona con un emborronamiento. Es por esta razón que hemos decidido generar un dataset descartando las zonas que no sean de bosque, para ello hemos usado el SCL comentado en el punto 3.1.2, este nos permite utilizar imágenes que únicamente consten de 100 % de bosque, podemos ver la Fig.9 el clasificador de mapas que hemos usado.



True color image Classification map

Fig. 9: TCI de zona de Cataluña comparado con mapa con las zonas clasificadas.

Una vez tenemos el nuevo dataset nos encontramos en la segunda fase de experimentos descrita en el punto 3.3.2, como ya hemos comentado, realizaremos 3 experimentos donde jugamos con la escala que aumentamos la imagen de entrada de 32×32, hasta finalmente conseguir una imagen de 1024×1024. Para ello, primero hemos entrenado la CARN, para que aprenda a realizar el escalado de ×2 y de ×4. Escogemos esta arquitectura ya que es rápida al no realizar un número excesivo de operaciones y en la Fig.10 se pueden observar los resultados, tanto visuales como los valores de las métricas de PSNR y SSMI, estos resultados son satisfactorios ya que conseguimos mejores valores que otro tipo de escalado más sencillo, sin que el coste computacional sea excesivo, con un tiempo relativamente bajo. Esto es muy favorable para nuestro trabajo global ya que si queremos realizar un vuelo simulado necesitaremos que el modelaje 3D sea lo más rápido posible. En la tabla 2 se observan los resultados finales comparados con el reescalado de interpolación bicubica.

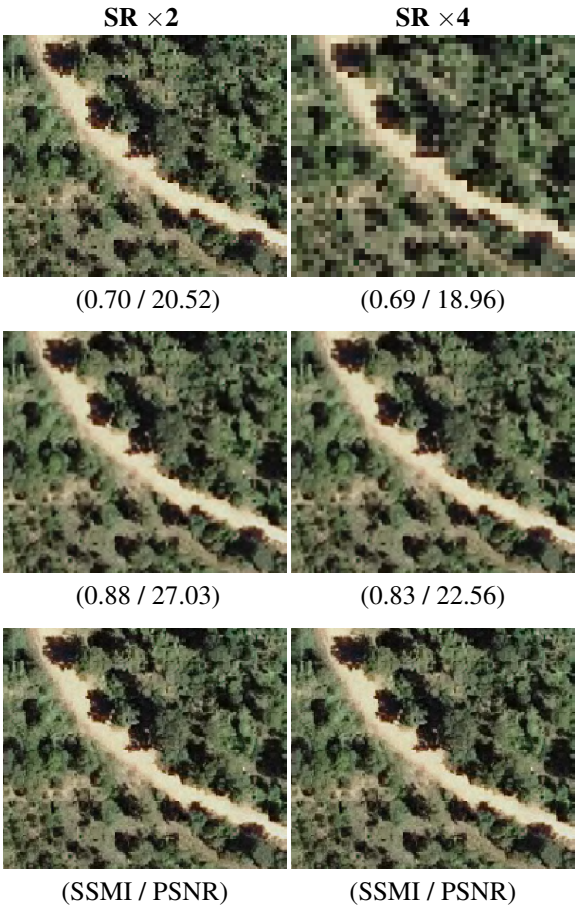


Fig. 10: Resultados visuales y numéricos del módelo CARN, de arriba a abajo, input output groundtruth.

Una vez acabado el entrenamiento y testeo de esta fase parcial, nos disponemos a entrenar la otra fase, la de recreación. Entrenamos la pix2pix con siempre el mismo tipo de entrada 32×32 (8 m/pix) pero con los distintos tamaños de salida de la red 256×256 (1 m/pix), 512×512 (50 cm/pix) o 1024×1024 (25 cm/pix), las imágenes que testeamos del primer y segundo experimento las pasamos por el modelo de CARN, ya entrenado, haciendo un re escalado de ×2 y ×4, respectivamente. En las Fig. 11, 12, 13 se pueden ver los resultados visuales. En la tabla 2 se pueden observar los resultados numéricos, podemos observar como la recreación de ×32 obtiene los mejores resultados pero el tiempo de ejecución es 3 veces mayor que la segunda más rápida. Queremos remarcar con inciso el tema del tiempo ya que la finalidad de este trabajo es la simulación de vuelos 3D a partir de estas imágenes generadas.

Por último, pasamos a la fase experimental final comentada en el punto 3.3.3, en ella hemos cogido la imagen en alta calidad (25 cm/pix) y la hemos degradado hasta el punto que sea lo más similar posible a la imagen de Sentinel-2, en este proceso el conjunto de imágenes de Sentinel-2 se ha aumentado su resolución a 1024×1024 y de las imágenes de avión se ha aplicado un emborronamiento de tipo Gaussiano. Aplicando varios tipos de máscara para emborronar, a partir de un conjunto reducido del dataset se ha obtenido una máscara óptima que se ha aplicado a la totalidad

TABLA 2: RESULTADOS NÚMERICOS DE LOS EXPERIMENTOS.

FASE	Resampling	Método	SSMI	PSNR	FID	tiempo
Superresolución	×2 (de 50 cm a 25 cm)	Bicúbico	0.74	23.33	-	0.001 s
		CARN	0.90	27.03	-	0.005 s
	×4 (de 1 m a 25 cm)	Bicúbico	0.72	20.51	-	0.002 s
		CARN	0.88	21.80	-	0.007 s
Recreación + Superresolución	×32 (de 8 m a 25 cm)	Pix2Pix	0.13	15.2	98.70	1.500 s
	×16 + ×2 (de 8 m a 50 cm y a 25 cm)	Pix2Pix + CARN	0.10	14.9	105.31	0.510 s
	×8 + ×4 (de 8 m a 1 m y a 25 cm)	Pix2Pix + CARN	0.11	14.33	113.78	0.160 s
Recreación (input emborronado)	×32 (de 8 m a 25 cm)	Pix2Pix	0.23	15.70	92.95	2.020 s
Recreación (input imagen satélite)	×32 (de 8 m a 25 cm)	Pix2Pix	0.06	13.24	131.54	1.930 s

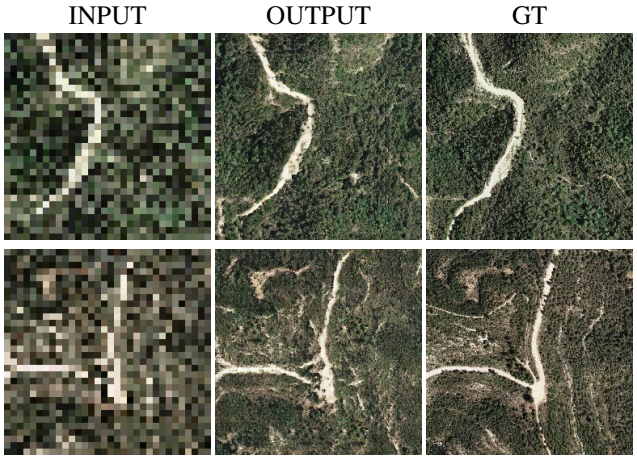


Fig. 11: Resultados visuales de imágenes testeadas en la fase de 8 m/pix a 1 m/pix usando pix2pix y de 1 m/pix a 25 cm/pix usando CARN.

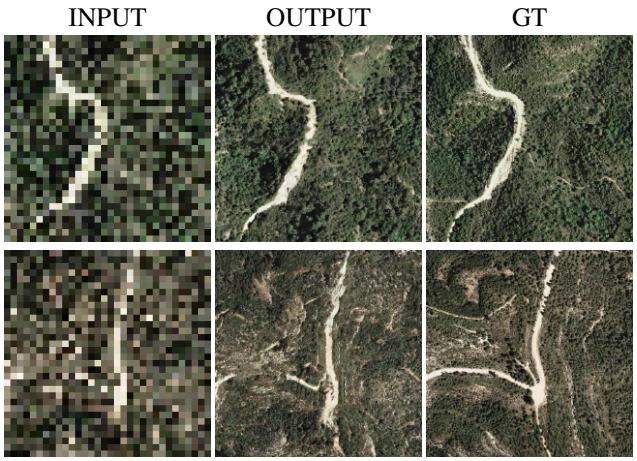


Fig. 12: Resultados visuales de imágenes testeadas en la fase de 8 m/pix a 50 cm/pix usando pix2pix y de 50 cm/pix a 25 cm/pix usando CARN.

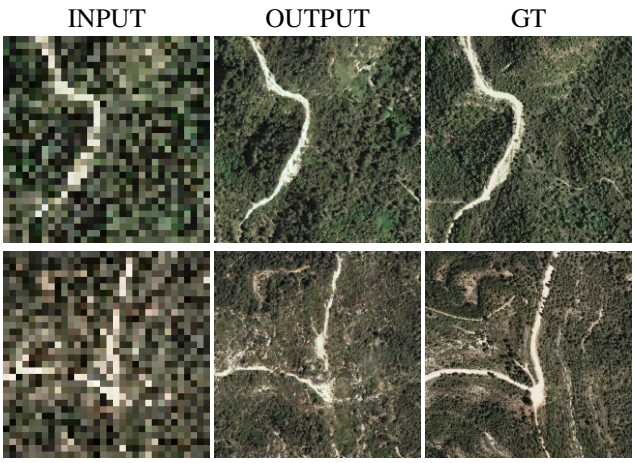


Fig. 13: Resultados visuales de imágenes testeadas en la fase de 8 m/pix a 25 cm/pix usando pix2pix.

de imágenes del dataset. Para medir la similaridad de las imágenes de Sentinel-2 con las emborronadas, se normaliza por banda las imágenes y se calcula el error por píxel aplicando la suma de todas las diferencias absolutas entre los propios píxel de las imágenes. Las imágenes emborronadas con mayor similitud a las de satélite son las que se utilizan como conjunto de entrenamiento. En la Fig.14 podemos observar el error por píxel aplicando distintos tamaños de máscara. Hemos realizado un nuevo entrenamiento con este último dataset generado, podemos observar los resultados visuales en la Fig.15, y las métricas en la tabla 2. Con este modelo se ha realizado un fine-tunning dando como entrada las imágenes de satélite, pero los resultados no han sido favorables como se puede ver en la Fig.15. Por último hemos hecho un entrenamiento con las modificaciones de normalización explicadas en el punto 3.3.3, en este caso la imagen de entrada es la de Sentinel-2. Podemos observar los resultados en la Fig.15 y en la tabla 2. En este caso los resultados no están al nivel de los otros experimentos, y visualmente se percibe una imagen menos realista. Aunque a rasgos generales si que observamos como imita el patrón de la imagen original.

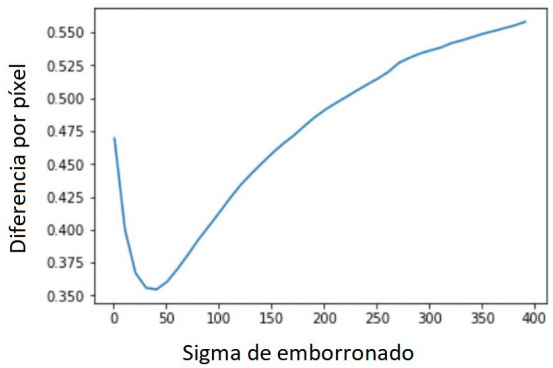


Fig. 14: Obtención del parámetro σ de emborronado entre la imagen Sentinel-2 y la simulación decimada que proviene directamente del groundtruth.

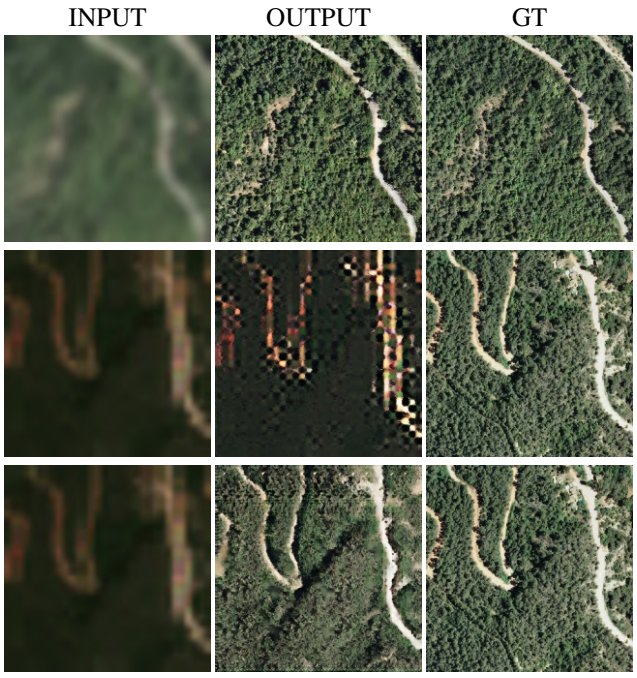


Fig. 15: Resultados visuales de la útima fase: (arriba) entrenamiento con dataset emborronado, (medio) fine tuning de dataset emborronado a Sentinel-2, (abajo) entrenamiento con dataset imágenes Sentinel-2.

5 Conclusiones

Basándonos en los resultados finales podemos comentar que estos, por lo general, han sido satisfactorios, es bien cierto que estas pruebas están enfocadas a un solo tipo de región, zona de montaña y bosque de la provincia de Barcelona. Pero demostramos que este tipo de tecnología es capaz de enfrontarse a problemas de esta magnitud. En la fase de super resolución basándonos en nuestros objetivos buscamos un modelo rápido y que funcione mejor que una simple interpolación, y lo conseguimos con el modelo desarrollado. Los experimentos de la segunda fase nos han servido para saber cual es la mejor forma de obtener imágenes de 25 cm/pix a partir de una de 8 m/pix, la recreación directa ($\times$32) da el mejor resultado pero requiere 3 veces más de tiempo que la segunda mejor. En los últimos experimentos hemos, en primera instancia, entrenado con datos los más similares a los de satélite para así readaptar el modelo, y finalmente hemos entrenado con datos reales de satélite. Podemos ver que los resultados de satélite son peores ya que estos tienen menos información que el resto, una de las causas puede ser que estos sean de peor calidad de la esperada. Estos resultados como guia de trabajo de este tipo de tecnologías y como trabajo futuro se espera seguir readaptando la arquitectura para así poder conseguir mejores resultados.

Referencias

[1] ESA Sentine Homepage, `https://sentinel.esa.int`. Last accessed 6 Oct 2019

[2] SentinelSat API Homepage, `https://sentinelsat.readthedocs.io/en/stable/api.html`. Last accessed 9 Nov 2019

[3] Institut Cartogràfic i Geològic de Catalunya Homepage, `https://www.icgc.cat/`. Last accessed 9 Nov 2019

[4] Copernicus Open Access Hub Homepage, `https://scihub.copernicus.eu`. Last accessed 9 Nov 2019

[5] OWSLib 0.18.0 documentation, `https://geopython.github.io/OWSLib/`. Last accessed 9 Nov 2019

[6] Sen2cor Homepage `https://step.esa.int/main/third-party-plugins-2/sen2cor/`.Last accessed 9 Nov 2019

[7] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In NIPS'2014.

[8] Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-Image Translation with Conditional Adversarial Networks. arXiv.org (2016)

[9] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation.

[10] N. Ahn, B. Kang, and K.-A. Sohn. Fast, accurate, and, lightweight super-resolution with cascading residual network. arXiv preprint arXiv:1803.08664, 2018.

[11] Z. Wang, J. Chen, and S. C. Hoi. Deep Learning for Image Super-resolution: A Survey. arXiv preprint arXiv:1902.06068, 2019.

[12] Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. N. StackGAN++: Realistic image synthesis with stacked generative adversarial networks. TPAMI.

[13] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In NIPS, pp. 6626–6637. 2017.