Tesis de Maestría - PEDECIBA Informática

Super-Resolución en Imágenes

Mercedes Marzoa Tanco

Tutores: Dr. Andrés Almansa
Dr. Matias Di Martino
Dr. Gonzalo Tejera

Instituto de Computación - Facultad de Ingeniería
Universidad de la República

June 2019

Resumen

El problema de super-resolución refiere al proceso de mejorar la resolución espacial de una imagen estimando la alta resolución a partir de una o varias imágenes degradadas de baja resolución. Estas técnicas han sido ampliamente investigadas debido al gran impacto que tienen en variadas aplicaciones, como ser aplicaciones médicas, imágenes satelitales o vídeo vigilancia. Recientemente con el desarrollo de las redes neuronales profundas han surgido varias alternativas eficaces para resolver este problema.

En este trabajo se presenta el estado del arte para los algoritmos de super-resolución, utilizando métodos mono-imagen y multi-imagen. Se comparan los métodos basados en redes neuronales profundas con un método variacional propuesto recientemente (STV). En particular, se estudia como impactan las diferentes estrategias con la incertidumbre en la estimación de la traslación entre las imágenes de baja resolución.

Para entrenar los métodos basados en redes neuronales es necesario generar muestras sintéticas a partir de imágenes de alta resolución. Se evalúa como la elección de los métodos de sub-muestreo y sobre-muestreo afecta el aprendizaje de las redes. Además, se cuantifica el impacto de entrenar y testear una red utilizando datos generados con diferentes modelo de degradación. Los experimentos sugieren que la definición de estos pasos juegan un papel fundamental y afectan significativamente lo que las redes aprenden.

Se realizan experimentos con imágenes satelitales, suponiendo el modelo de degradación desconocido.

Por último, se proporciona como resultado adicional de esta tesis, un programa de código abierto que permite mejorar la resolución de una imagen, seleccionando el tipo de red (mono-imagen o multi-imagen) y el modelo de degradación con el que se generan los datos de entrenamiento. La interfaz puede ser utilizada para reproducir los resultados aquí presentados.

Índice general

Índice de figuras

Capítulo 1

Introducción

El problema de super-resolución (SR) refiere al proceso de mejorar la resolución espacial de una imagen estimando la alta resolución a partir de una o varias imágenes degradadas de baja resolución (Figura 1.1) [41]. Las técnicas para resolver la SR han sido ampliamente investigadas debido al gran impacto que tienen en variadas aplicaciones, como ser aplicaciones médicas [16, 27, 6], imágenes satelitales [7] o videovigilancia [51].

Dentro de los impactos en aplicaciones médicas se encuentra su utilización en las imágenes de resonancia magnética. La adquisición de este tipo de imágenes en alta resolución requiere que el paciente se mantenga quieto por largos periodos de tiempo, causando disconformidad e incrementando la probabilidad de movimiento, lo que puede llevar a obtener imágenes con ruido. Por tal motivo, adquirir la imagen en baja resolución y procesarla para crear una de alta resolución reduce en gran medida el tiempo necesario de la adquisición [40, 37]. En el caso de las imágenes satelitales mejorar la resolución resulta de gran utilidad para lograr una mejor identificación y una precisa extracción de las características de cobertura de la tierra, lo que resulta de gran utilidad en diversas áreas de trabajo, dentro de las que se encuentran la inteligencia militar, planificación agrícola, y gestión de recursos hídricos [44, 43, 45]. Por último, en el caso de la videovigilancia es frecuente la utilización de técnicas de SR debido a la necesidad de reconocer los objetos de una escena, por ejemplo, el rostro de una persona o la matricula de un vehículo [51, 33].

1.1. Objetivos y aportes

En el marco del proyecto realizado por el centro nacional de estudios espaciales de Francia (CNES) y el laboratorio de matemática aplicada (MAP5) de la Université Paris Descartes se busca estudiar diferentes algoritmos de super-resolución, en particular para su utilización en imágenes satelitales.

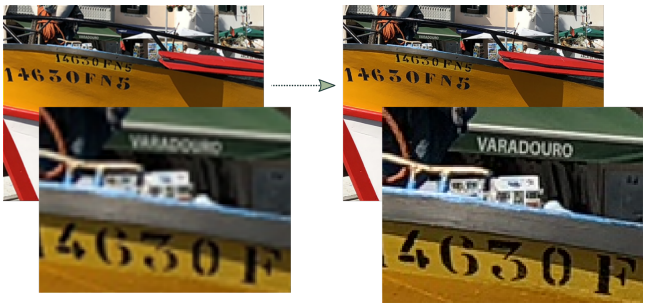


Figura 1.1 Ejemplo de super-resolución: a la izquierda se observa una imagen en baja resolución y a la derecha la misma imagen pero en alta resolución.

En este trabajo se presenta el estado del arte para los algoritmos de super-resolución, tanto para algoritmos que utilizan una única imagen de entrada como para algoritmos que utilizan varias imágenes de entrada para generar un imagen de alta resolución de salida. En particular se seleccionan algunos de los algoritmos de SR y se estudia el comportamiento de los mismos para las imágenes satelitales. A su vez, se analiza la importancia del sub-muestreo con respecto a la elección del núcleo de interpolación y como dicha elección afecta al proceso de aprendizaje de las redes. Más aún, se estudia como esta elección se encuentra relacionada al modelo de degradación y cómo su análisis numérico previo puede proporcionar información preliminar de cuán difícil es el proceso que se esta invirtiendo.

Por último, se realiza como resultado adicional de esta tesis, un programa de código abierto que permite mejorar la resolución de una imagen, seleccionando el tipo de red (mono-imagen o multi-imagen) y el modelo utilizado para el entrenamiento de la red.

1.2. Trabajos Relacionados

Los métodos de super-resolución han sido históricamente muy estudiados debido a la importancia que tiene la super-resolución en diversas áreas. Entre los algoritmos existentes se encuentran desde métodos básicos de interpolación a métodos más complejos, como los basados en ejemplos, los cuales aprenden las correspondencias entre secciones de imágenes de alta y baja resolución basándose en ejemplos existentes [13, 38, 12]. Otra familia de soluciones explota la auto-similitud de la imagen [21], por ejemplo, se basan en que estadísticamente las secciones en una imagen natural tienden a repetirse dentro de la misma imagen y a diferentes escalas. Por otro lado, los métodos basados en la dispersión (por ejemplo, [49]) utilizan una representación dispersa para las secciones de la entrada de baja resolución, y aprenden un conjunto de coeficientes óptimos de esta representación para generar la salida de alta resolución.

Recientemente debido a los buenos resultados obtenidos por las redes neuronales convolucionales se han propuesto una nueva generación de algoritmos de super-resolución basados en redes neuronales profundas, los cuales se pueden separar en dos grandes familias. Una de las familias de métodos existentes [29, 52, 9, 28, 31], llamada mono-imagen, utilizan una sola imagen como entrada, mientras un segundo tipo [42, 23, 26, 25], llamada multi-imagen, utiliza múltiples imágenes de baja resolución para reconstruir una de alta resolución.

A continuación se presentan brevemente los principales trabajos realizados separándolos según sean de una u otra familia.

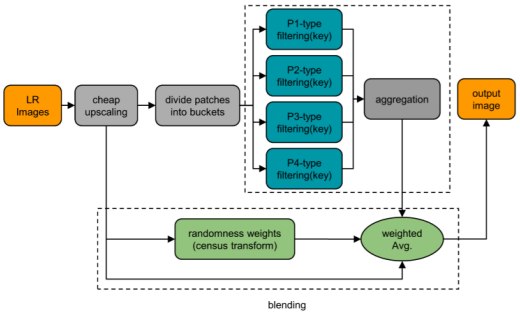
1.2.1. Super-Resolución Mono-Imagen

Un algoritmo rápido y preciso para obtener una imagen de alta resolución (HR) a partir de una de baja resolución (LR) es el algoritmo RAISR propuesto por Romano et al [38]. La idea central del algoritmo es mejorar la calidad de un método de interpolación económico (ej. Bilineal) aplicando un conjunto de filtros pre-aprendidos mediante pares LR-HR de secciones de imágenes (Ver figura 1.2). Los filtros se diseñan para minimizar la distancia euclidiana entre la imagen de entrada y la imagen real. El algoritmo sigue los siguientes pasos: (i) Aumenta el tamaño de la imagen de entrada mediante un método de interpolación económico. (ii) Crea una tabla hash conteniendo el conjunto de filtros aprendidos, utilizando la base de datos de entrenamiento, donde las claves de la tabla hash son en función de las propiedades del gradiente. Los filtros se aplican a la salida de (i) para mejorar la calidad. (iii) Se mezclan las salidas de (i) y (ii) con diferentes pesos en cada píxel para así producir la salida final.

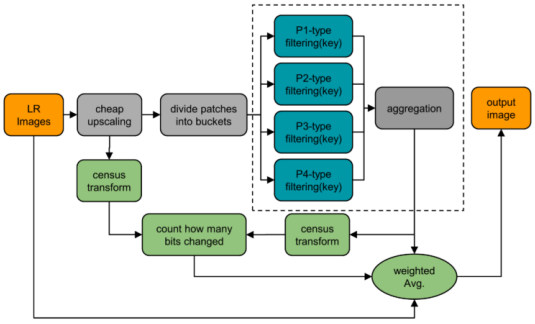
Unos de los primeros en introducir el uso de redes convolucionales para resolver el problema de SR fueron Dong et al. [9] quienes propusieron la utilización de una red convolucional de tres capas, alcanzando buenos resultados (Figura 1.3). Luego, Kim et al. [29] proponen una red convolucional recursiva, la cual permite mejorar el rendimiento sin introducir parámetros por las capas convolucionales adicionales y proponen a su vez dos métodos para evadir el problema de explosión/desvanecimiento del gradiente, supervisión de las recursiones y conexiones salteadas (Figura 1.4).

Otro tipo de red es SRGAN [31], propuesta por Leding et al. SRGAN es una *generative adversarial network* (GAN) capaz de inferir imágenes foto-realistas a escala x4. Consiste en una red profunda con arquitectura ResNet utilizando el concepto de GAN para crear una función de perdida perceptiva, remplazan el uso del error cuadrático medio (MSE) como medida de perdida con una nueva medida calculada en los mapas de características de la red, lo que lo hace más invariante a cambios en el espacio de píxeles (Figura 1.5).

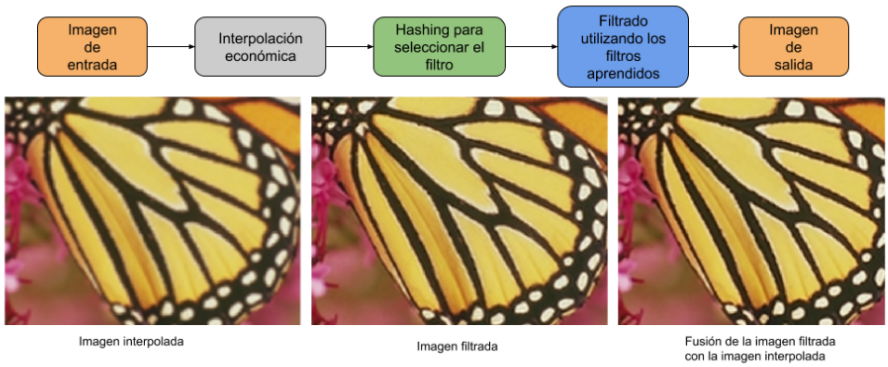
Otro tipo de red propuesto son las redes neuronales residuales. Uno de estos trabajos es el presentado por Zhang et al. [52] que proponen una red neuronal residual para así utilizar completamente todas las características jerárquicas de la imagen LR original. Proponen un bloque denso residual (RDB) como el módulo de construcción de la red. El RDB puede leer el estado del RDB anterior a través de un mecanismo de memoria contigua, y puede utilizar completamente todas las capas dentro de él a través de conexiones densas locales. Las características acumuladas se conservan de forma adaptativa mediante la fusión de características locales. Por último, proponen la fusión de características globales para fusionar de forma adaptativa las características jerárquicas de todos los RDB en el espacio LR. Con el aprendizaje residual global, combinan las características superficiales y las características profundas, dando como resultado las características densas globales de la imagen LR original (Figura 1.6).



(a) La fusión selecciona los píxeles filtrados en áreas estructuradas y los píxeles de la interpolación económica en áreas planas. Imagen tomada de [38].



(b) Permite la mejora de contraste. La interpolación falla en la recuperación de áreas estructuradas, donde los filtros pre-aprendidos juegan un rol clave. Imagen tomada de [38].



(c) Flujo para aumentar la resolución de una imagen.

Figura 1.2 **Esquema de RAISR** Proponen dos esquemas de fusión diferentes, (a) y (b), que resultan en diferentes efectos de mejora. (a) Permite la amplificación de las altas frecuencias aplicando los filtros pre-aprendidos, sin modificar las frecuencias medias o bajas. En las áreas de baja frecuencia un aumento de escala lineal produce buenos resultados dado que no hay detalles finos para recuperar ni alising para eliminar, por lo que no es necesario mejorar los resultados en estas áreas. (b) Permite la mejora de una amplia gama de frecuencias, lo que genera imágenes de mejor aspecto debido al efecto de mejora de contraste. En (c) se muestra el esquema de flujo básico de los diferentes bloques del algoritmo para aumentar la resolución de una imagen.

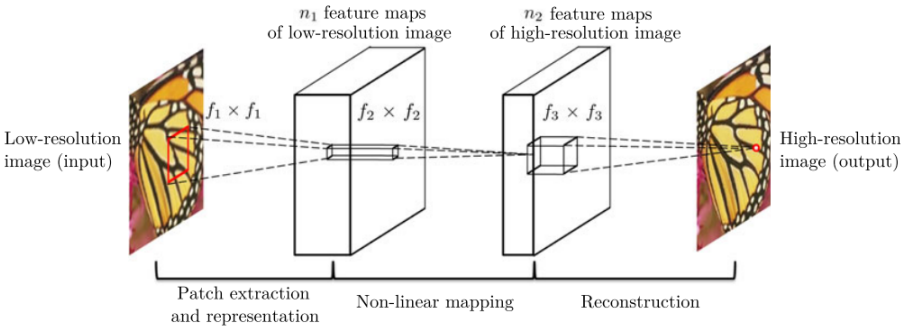


Figura 1.3 **Arquitectura propuesta por Dong et al. [9].** Dada una imagen de baja resolución Y, la primera capa de convolución extrae un conjunto de mapas de características, la segunda capa mapea dichos mapas con las secciones de alta resolución y por último, la tercer capa combina las predicciones dentro de un vecindario para producir la imagen final de alta resolución. Imagen tomada de [9].

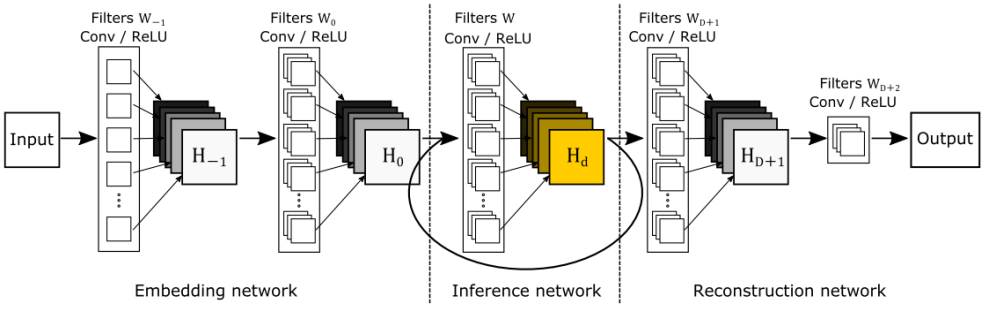


Figura 1.4 **Arquitectura propuesta por Kim et al [29].** Se compone por tres sub-redes, la *embedding network*, *inference network* y *reconstruction network*. La *inference network* es la encargada de la recursión mediante una capa recursiva. Imagen tomada de [29].

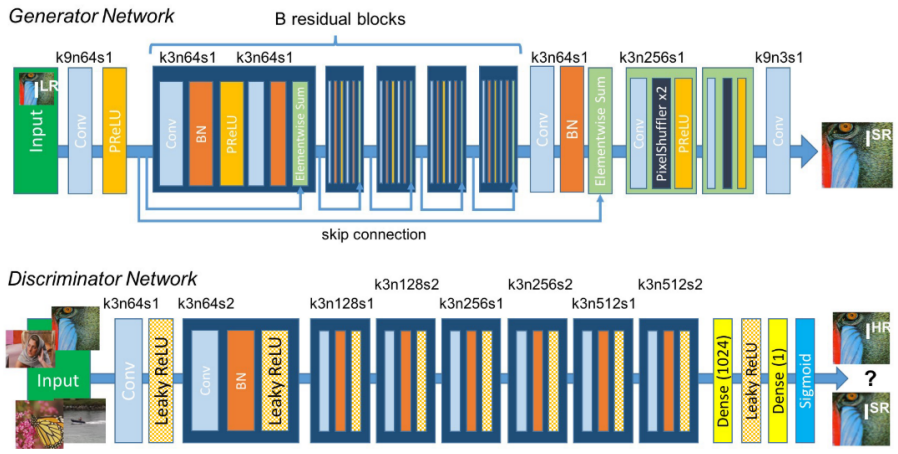
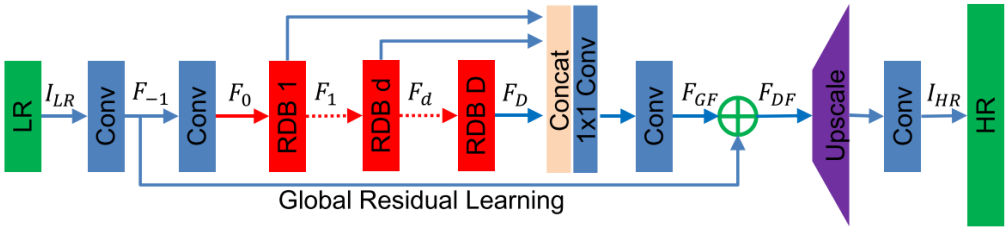
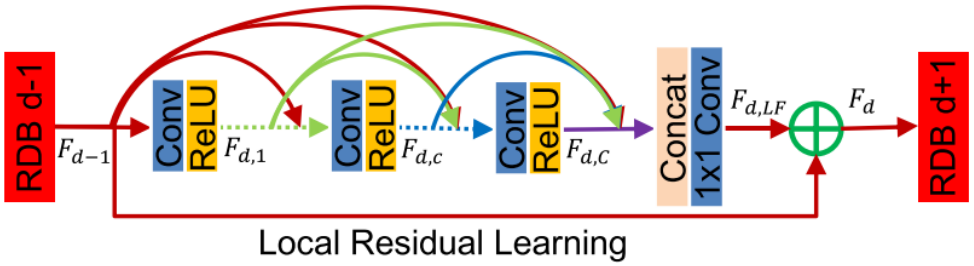


Figura 1.5 **Arquitectura propuesta por Ledig et al. [31].** Por cada capa de convolución se indica su correspondiente tamaño de núcleo (k), número de mapas (n) y stride (s). Para discriminar las imágenes reales HR de las SR generadas, crean la red discriminadora. Imagen tomada de [31].



(a) Consta principalmente de cuatro partes: la red de extracción de características superficial (SFENet), los bloques densos residuales (RDB), la fusión densa de característica (DFF) y la red de muestreo ascendente (UPNet).



(b) Arquitectura del bloque residual denso (RDB)

Figura 1.6 **Arquitectura propuesta por Zhang et al. [52].** Imágenes tomadas de [52].

1.2.2. Super-Resolución Multi-Imagen

Los algoritmos de SR multi-imagen generalmente resuelven dos problemas, el de estimación de registrado donde se estima el movimiento entre las imágenes y el de estimar la imagen HR a partir de la información recuperada en el registrado.
Para el registrado existen numeroso trabajos tanto de flujo óptico [20, 24] cómo de block matching [2, 53].

Hay diferentes enfoques para realizar la SR, algunos realizan la estimación de movimiento y luego la estimación de la imagen HR entrenando por separado cada parte, mientras otros trabajos realizan el entrenamiento de punta a punta, es decir entrenan la estimación de movimiento y de la imagen HR en conjunto. Uno de los trabajos que realiza el entrenamiento por separado es el de Kappeler et al. [26], quienes proponen una red neuronal convolucional (CNN), primero estiman el movimiento con el algoritmo de Druleas [11] y luego compensan el movimiento entre las imágenes consecutivas, para ser utilizadas como entrada en la CNN encargada de proveer la imagen HR (Figura 1.7). Un procedimiento alternativo es el propuesto por Tao et al. [42]. Presentan una red que realiza ambos entrenamientos en conjunto. Proponen una red neuronal convolucional que tiene una capa de compensación de movimiento sub-píxel y la entrenan de punta a punta, tanto para la estimación de movimiento como para la estimación de la imagen de alta resolución (Figura 1.8).

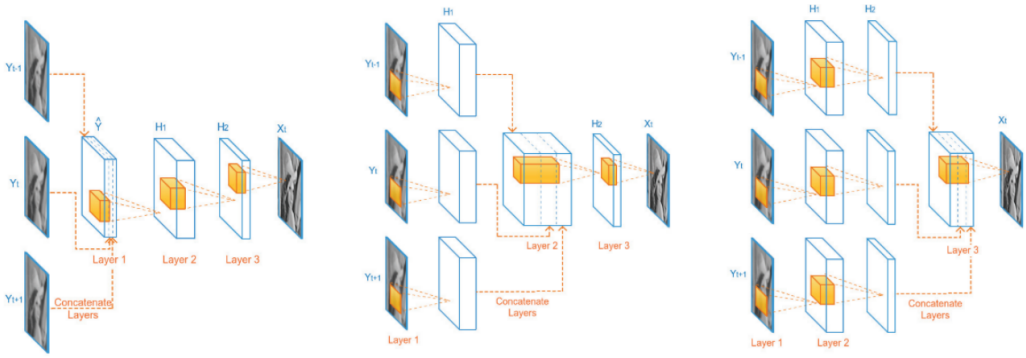


Figura 1.7 **Arquitectura propuesta por Kappler et al. [26].** Se compone por tres capas de convolución. Se muestran las tres opciones de incorporación de las imágenes previas y posteriores en el proceso, por simplicidad se muestra la arquitectura para tres imágenes de entrada. Imagen tomada de [26].

1.3. Organización del documento

El resto del documento queda organizado de la siguiente forma, en el capítulo 2 se presentan los aspectos teóricos que enmarcan el trabajo, en particular se presenta el tema de

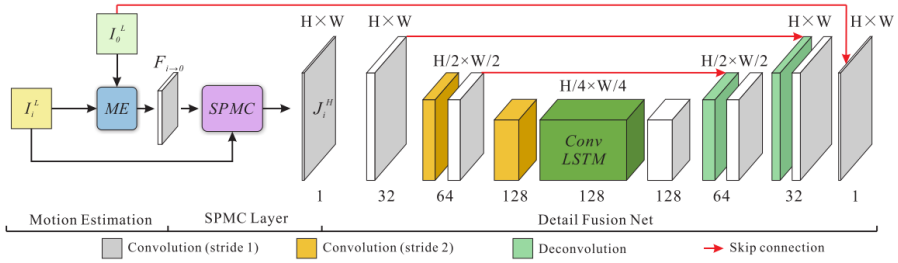


Figura 1.8 **Arquitectura propuesta por Tao et al [42]**. Presenta una estructura del tipo *encoder-decoder* con conexiones salteadas. Imagen tomada de [42].

super-resolución, estimación de movimiento, redes neuronales y técnicas utilizadas para la evaluación. En el capítulo 3 se presentan los tres métodos de súper-resolución utilizados en este trabajo, y las extensiones llevadas a cabo en este contexto, explicando cual es el motivo de las mimas.

En el capítulo 4 se presentan los experimentos realizados. Por último, en el capítulo 5 se presenta la discusión final sobre los resultados obtenidos y las principales lineas de investigación a explotar a partir de lo aquí realizado.

Capítulo 2

Marco Teórico

Este capítulo presenta los aspectos teóricos generales que enmarcan el trabajo. El mismo cuenta con cuatro secciones, en la primera sección se presenta el problema de super-resolución, en la segunda sección se presentan las técnicas de estimación de movimiento, en la tercera sección se presentan las redes neuronales y por último, en la cuarta sección se presentan técnicas de evaluación.

2.1. Super-Resolución

En la mayoría de aplicaciones de imágenes digitales es critico contar con imágenes de alta resolución (HR), buscando mejorar el aspecto para la percepción humana o para mejorar la representación para el posterior procesamiento por las máquinas [16, 27, 6, 7, 51]. La resolución de una imagen determina el nivel de detalle que puede ser extraído de la misma.

Una imagen digital está formada por pequeños elementos llamados píxeles, la resolución espacial refiere a la densidad de píxeles en una imagen y se mide en píxeles por unidad de área. Una imagen HR es entonces una imagen con alta densidad de píxeles. La resolución espacial de una imagen se ve limitada por el dispositivo de adquisición, los dos principales tipos de sensores utilizados son los CCD o dispositivos de carga acoplada y los CMOS o Semiconductores Complementarios de Óxido Metálico [41]. Estos sensores normalmente están dispuestos en una matriz bidimensional. El tamaño del sensor, o de forma equivalente, el número de elementos del sensor por unidad de área determinan la resolución espacial de la imagen a capturar. Cuanto mayor densidad tienen los sensores, mayor resolución espacial tienen las imágenes. Una manera directa de aumentar la resolución espacial es aumentar la densidad del sensor reduciendo el tamaño del mismo. Sin embargo, a medida que el tamaño del sensor disminuye, la cantidad de luz que incide en cada sensor también disminuye, causando el llamado ruido de disparo. Además, el costo del hardware del sensor aumenta

con el aumento de la densidad del sensor o con el incremento de la densidad de píxeles de la imagen. Por lo tanto, la limitación del hardware en el tamaño del sensor restringe la resolución espacial de la imagen que puede ser capturada [22].

Mientras que los sensores de adquisición limitan la resolución espacial de la imagen, los detalles de la imagen, es decir las bandas de alta frecuencia, también se encuentran limitados por la óptica debido a los desenfoques de la lente (asociados con la función de dispersión del sistema (PSF)), los efectos de aberración de la lente, la difracción o el desenfoque óptico debido al movimiento.

La construcción de los sensores de adquisición y los componentes ópticos para capturar las imágenes de alta resolución son muy caros. Además del costo, en muchas aplicaciones la resolución también está limitada por la velocidad de la cámara y el hardware de almacenamiento. Otra forma de abordar el problema de obtener imágenes de alta resolución es aceptar imágenes degradadas y mediante técnicas de procesamiento de señales mejorar la resolución de las mismas. Estas técnicas se denominan super-resolución (SR).

Se define entonces a la super-resolución como una técnica que construye un imagen de alta resolución a partir de una o varias imágenes observadas de baja resolución (LR), aumentando las componentes de alta frecuencia y eliminando la degradación causada por el proceso de captura de la imagen mediante cámaras de baja resolución. Los algoritmos de SR se pueden separar en dos grandes familias, los que utilizan una única imagen denominados comúnmente SR mono-imagen y los que utilizan varias imágenes combinando la información no redundante para formar una única imagen, denominados SR multi-imagen [22, 50].

2.1.1. Super-Resolución Mono-Imagen

Una imagen digital es una representación discreta de un campo de luz continuo. La formación de una imagen digital puede ser modelada como

$$u(i,j) = S_{i,j}(k * u(x,y)) \;\; (x,y) \in \Omega \subset \mathbb{R}^2, \tag{2.1}$$

donde el núcleo k representa el desenfoque óptico de la lente y S representa la operación de muestreo realizada por el sensor discreto de la cámara [41]. Las coordenadas discretas (i,j) se muestrean sobre el dominio regular $[1...H] \times [1...W] \stackrel{def}{=} \Omega^h$, donde h ilustra la distancia entre las muestras discretas, o lo que es lo mismo, el tamaño del píxel, W y H representan el ancho y alto de la imagen.

Formulación Discreta: Dada una imagen continua $u(x,y)$, se consideran dos muestras discretas $u^h(i,j)$ y $u^{rh}(i,j)$ asociadas a la grilla discreta Ω^h y Ω^{rh} respectivamente, siendo r el coeficiente de resolución, en el caso de $r = 2$ se tiene una muestra con el doble de

resolución que la otra. La primera muestra se considera una representación de alta resolución de la segunda.

El problema de super-resolución se formula entonces como la estimación a partir de una muestra de baja resolución u^{rh} de su contraparte de alta resolución u^h.

2.1.2. Super-Resolución Multi-Imagen

La super-resolución multi-imagen se define como el proceso de alinear y combinar varias imágenes de baja resolución con pequeños desplazamientos entre ellas para producir una de mayor resolución (Figura 2.1) [22].
Al igual que en el modelo matemático para una única imagen (Ecuación 2.1), si se tienen n capturas se puede modelar el problema de super-resolución multi-imagen como

$$u(i,j) = S_{i,j}(k * \phi_p(u(x,y))), \;\; p = 1,..,n, \;\; (x,y) \in \Omega \subset \mathbb{R}^2, \tag{2.2}$$

siendo ϕ_i la deformación geométrica o desplazamiento.

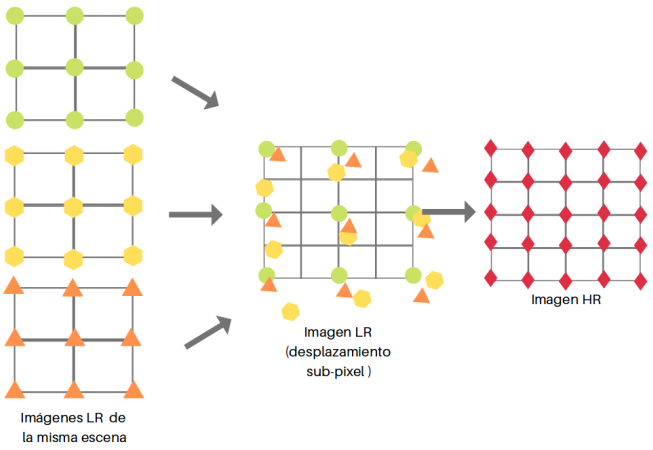


Figura 2.1 **Super-resolución multi-imagen.** A la izquierda se encuentran las capturas LR de una misma escena, en el centro se encuentran las imagen LR con alineación sub-pixel y a la derecha se muestra la imagen HR resultado de fusionar las imágenes LR utilizando métodos de super-resolución.

2.2. Estimación de movimiento

Las técnicas de alinear imágenes y estimar el movimiento en secuencias de imágenes han sido ampliamente utilizadas, uno de los primeros ejemplos es el algoritmo de alineación de traslación basado en secciones (patches en ingles) desarrollado por Lucas y Kanade en 1981

[34]. Con el tiempo se han desarrollado una gran cantidad de algoritmos más sofisticados, [3, 15] presentan un resumen de dichas técnicas. Dos de las técnicas más utilizadas para la estimación de movimiento son las llamadas Block Matching y Flujo Óptico.

Block Matching

La técnica de Block Matching (Figura 2.2) consiste en estimar el movimiento entre dos imágenes buscando la correspondencia de bloques entre ellas, siendo un bloque una porción rectangular de la imagen. Para cada píxel en la primera imagen, o imagen de referencia, se calcula la distancia entre los bloques, para ello se centra un bloque en el píxel de referencia en la primer imagen y se calcula la distancia a todos los bloques de la segunda imagen dentro de un rango (ventana) alrededor del píxel de referencia. El bloque que tenga la menor distancia es el bloque de correspondencia. Una función de costo común para calcular la distancia es la suma de la diferencia al cuadrado de los píxeles dentro del bloque (SDD), pero se pueden utilizar otras.

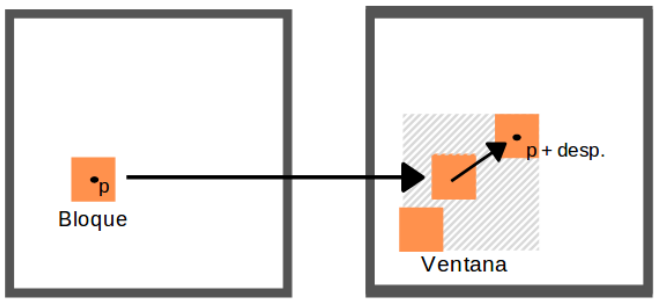


Figura 2.2 **Block Matching:** Busca calcular la distancia entre el bloque de la imagen de referencia (izquierda) centrado en el píxel **p** y todos los bloques de la segunda imagen (derecha) en una ventana, es decir, centrados en el píxel **p** más un desplazamiento (**p + desp.**).

Flujo Óptico

Una manera de establecer las correspondencias entre dos imágenes consiste en estimar las traslaciones locales que deben aplicarse a cada región de una imagen para que, localmente, coincida con la segunda[41]. Dada una imagen de referencia $I_0(x)$ muestreada en ubicaciones discretas de píxeles $\mathbf{x}_i = (x_i, y_i)$, se quiere encontrar dónde se ubica con respecto a la imagen $I_1(x)$. Una solución de mínimos cuadrados a este problema es encontrar el mínimo de la

suma de la diferencia al cuadrado (SSD)

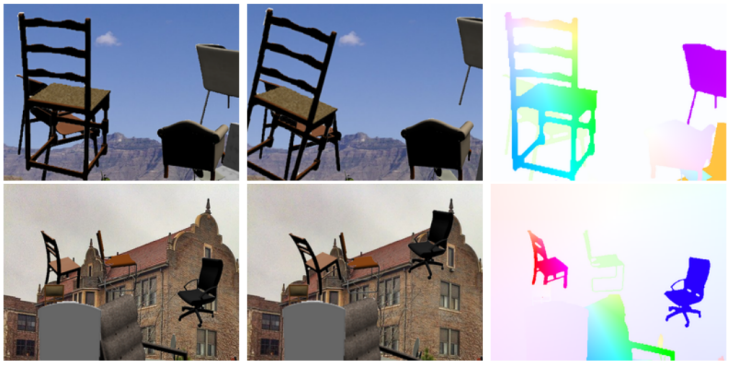
$$E_{SSD}(\mathbf{u}) = \sum_i [I_1(x_i + u) - I_0(x_i)]^2 = \sum_i e_i^2, \tag{2.3}$$

donde $\mathbf{u} = (u, v)$ es el desplazamiento y $e_i = I_1(x_i + u) - I_0(x_i)$ se denomina el error residual o diferencia de desplazamiento entre imágenes. Se puede tener una medida de error más robusta a outliers remplazando el termino de error cuadrático con una función más robusta. Los algoritmos de flujo óptico calculan el movimiento de cada píxel de forma individual, es decir, el número de vectores de movimiento es igual al número de pixeles (ver Figura 2.3). La ecuación de flujo óptico análoga a la ecuación 2.3 se puede escribir como

$$E_{SSD-OF}(\{u_i\}) = \sum_i [I_1(x_i + u_i) - I_0(x_i)]^2, \tag{2.4}$$

Dado que el número de variables $\{u_i\}$ es dos veces el numero de medidas, el problema es indeterminado. Hay dos formas clásicas de resolver este problema, una es realizando la sumatoria local sobre la superposición de regiones y la otra es agregar términos de suavizado en el campo u_i, utilizando regularización o campos aleatorios de Markov, y luego se busca un mínimo global.

Un número importante de trabajos para la estimación del flujo óptico se basan en la utilización de patches [34], otros proponen regularizar minimizando sobre todos los vectores de flujo [20], mientras otros lo hacen mediante redes neuronales [10, 30].



(a) Dos imágenes de la base de datos Flying Chairs y su campo de flujo.

(b) Código de colores del campo de flujo. El píxel central no tiene movimiento y el desplazamiento del resto de los píxeles es el vector desde el centro al píxel.

Figura 2.3 **Flujo óptico.** Imágenes tomadas de [10].

2.3. Redes Neuronales

Antes de introducir las redes neuronales se introduce brevemente el aprendizaje automático. Un algoritmo de aprendizaje automático es un algoritmo que puede aprender de los datos [14]. Mitchell [35] define aprender como sigue: *Se dice que un programa de computadora aprende de la experiencia E con respecto a alguna clase de tareas T y la medida de rendimiento P, si su desempeño en las tareas en T, medido por P, mejora con la experiencia E.*

Sea un *ejemplo* una colección de características que se han medido cuantitativamente a partir de algún objeto o evento, representado normalmente como un vector $x \in \mathbb{R}^n$ donde cada entrada x_i del vector es una característica. Las tareas de aprendizaje automático se describen generalmente en términos de cómo el sistema de aprendizaje automático debe procesar el *ejemplo*. Un ejemplo de una tarea de aprendizaje automático es la clasificación, en este tipo de tarea se le pide al programa que especifique a cuál de las categorías k pertenece una entrada. Para resolver esta tarea, al algoritmo de aprendizaje generalmente se le pide que produzca una función $f : \mathbb{R}^n \to 1,...,k$. Cuando $y = f(x)$, el modelo asigna una entrada descrita por el vector x a una categoría identificada por el código numérico y. Hay otras variantes de la tarea de clasificación, por ejemplo, donde f genera una distribución de probabilidad sobre las clases. Un ejemplo de una tarea de clasificación es el reconocimiento de objetos, donde la entrada es una imagen, y la salida es un código numérico que identifica el objeto en la imagen [14].

Para evaluar las habilidades de un algoritmo de aprendizaje automático, se debe diseñar una medida cuantitativa de su rendimiento. Por lo general, esta medida de rendimiento P es específica de la tarea T que realiza el sistema.

Los algoritmos de aprendizaje automático pueden clasificarse en términos generales como supervisados o no supervisados, según el tipo de experiencia que se les permite tener durante el proceso de aprendizaje. El aprendizaje no supervisado implica observar varios ejemplos de un vector aleatorio x, e intentar aprender implícita o explícitamente la distribución de probabilidad $p(x)$, o algunas propiedades interesantes de esa distribución, mientras que el aprendizaje supervisado implica observar varios ejemplos de un vector aleatorio x y un valor asociado o vector y, y aprender a predecir y a partir de x, generalmente estimando $p(y|x)$.

En 1957 Rosenblatt crea Perceptron [39], considerada una de las primeras máquinas capaces de aprender (Figura 2.4). Estaba compuesta por una única capa capaz de separar datos linealmente separables. Las redes neuronales que se utilizan actualmente manejan las ideas introducidas por Rosenblatt intercalando millones de neuronas interconectadas de diversas maneras.

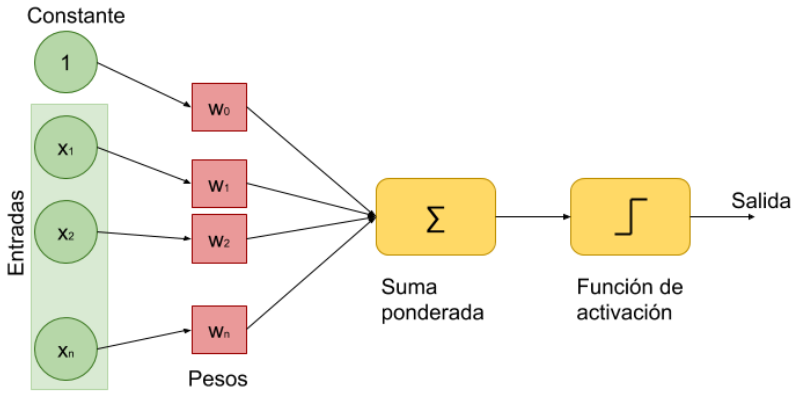


Figura 2.4 Perceptron.

Las redes neuronales consisten en un conjunto de unidades, llamadas neuronas, las cuales se encuentran conectadas entre sí. Usualmente los modelos de redes neuronales se organizan en diferentes capas, el tipo de capa más común es la capa totalmente conectada en la que las neuronas entre dos capas adyacentes están conectadas por pares, pero las neuronas dentro de una misma capa no comparten conexiones. A continuación se presentan algunos de los tipos de redes más utilizados.

2.3.1. Deep Feedforward Networks

Las Redes Neuronales Prealimentadas (Feedforward Neural Networks en ingles) tienen como objetivo aproximar una función f^*, por ejemplo, para un clasificador $y = f^*(x)$ se mapea una entrada x a una categoría y. La red define un mapeo $y = f(x;\theta)$ y aprende el valor de los parámetros θ que resultan en la mejor función de aproximación [14].

Se denominan redes *feedforward* debido a que la información fluye a través de la función que evalúa x, a través de los cálculos intermedios utilizados para definir f, y finalmente a la salida y. No existen conexiones de retroalimentación en las que las salidas de una capa sean entrada de la misma capa (red neuronal recurrente). Se representan por la composición de varias funciones, por ejemplo, si se tienen tres funciones conectadas la red sería $f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$ donde $f^{(1)}$ es la primera capa de la red, $f^{(2)}$ es la segunda capa y así, siendo el largo de la cadena lo que determina la profundidad del modelo. A la capa final de una red se la denomina capa de salida. Durante el entrenamiento no se especifica la salida de cada capa, sino que se da la entrada de la red y la salida, es decir los datos de punta a punta. A cada elemento de una capa $f^{(i)}$ se lo denomina neurona o unidad y busca, dada una serie de entradas, calcular un valor de salida utilizando su función de activación. Una neurona se

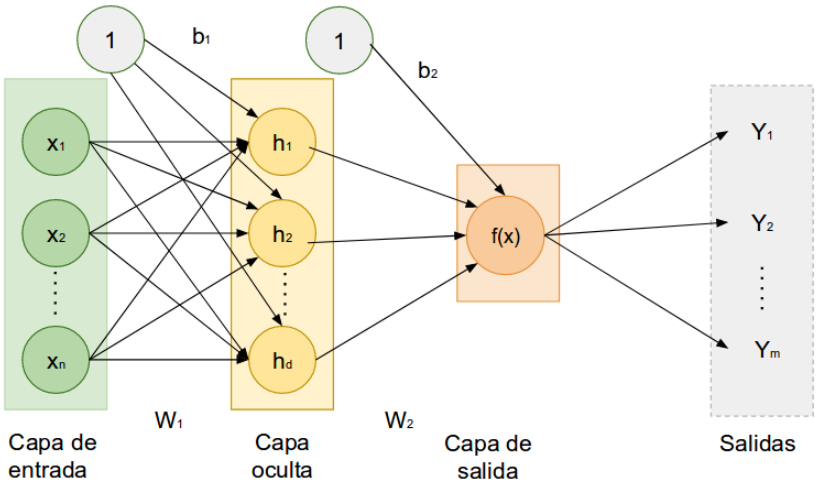


Figura 2.5 **Red neuronal de dos capas.** Red compuesta por n entradas, una capa oculta con d neuronas y una capa de salida con m neuronas. Notar que hay conexiones entre las capas pero no dentro de una misma capa.

compone en general de una operación lineal y una función de activación no lineal. En el caso de una red con una única capa oculta (Figura 2.5) se tiene:

- Entrada: x
- Pre-activación: $a = w^{(1)T}x + b^1$
- Activación capa oculta: $h^{(1)}(x) = g(a(x))$
- Capa de salida: $x = g(w^{(2)T}h^{(1)} + b^{(2)})$

Al momento de diseñar una red de este tipo se debe definir cuantas capas debe contener la red, cómo deben estar conectadas dichas capas y cuántas neuronas debe contener cada una. También es necesario definir la función de activación que se utilizará para la capa oculta. Una de las funciones de activación más utilizadas es la función ReLU - unidad lineal rectificada, la cual se define como $g(x) = max(0,x)$.

El aprendizaje profundo permite que los modelos computacionales compuestos de múltiples capas de procesamiento aprendan representaciones de datos con múltiples niveles de abstracción. Se utiliza el algoritmo de propagación hacia atrás (Backpropagation en inglés) para indicar cómo se deben transformar los parámetros internos utilizados para calcular la representación en cada capa a partir de la representación de la capa anterior. Si bien una única capa oculta con un número arbitrariamente grande de neuronas es un aproximador

universal, las redes poco profundas son ineficientes para representar funciones complejas. Sin embargo, las redes neuronales profundas pueden calcular funciones con un número de regiones que crece de manera exponencial con la profundidad de la red, siendo mucho más rápidas que una red de una única capa oculta con el mismo número de neuronas. La Figura 2.6 compara el límite de decisión de una red de una única capa con una de dos capas con el mismo número de unidades ocultas, ilustrando las ventajas de la profundidad. Se puede ver como el modelo profundo consigue el limite deseado de forma más precisa.

Otro de los motivos para utilizar redes profundas es debido a la organización jerárquica que

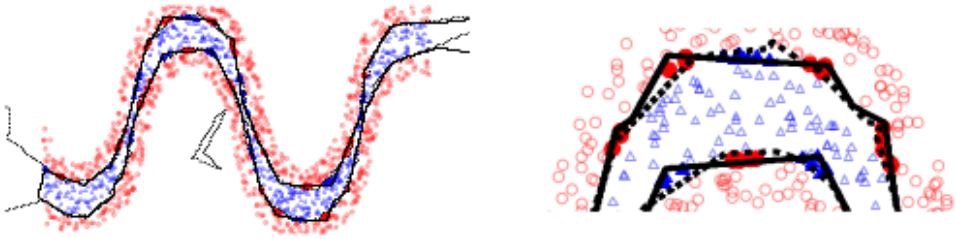


Figura 2.6 **Límite de decisión de una red de una capa vs. de dos capas con igual número de neuronas.** Clasificación binaria utilizando un modelo con 20 neuronas ocultas (linea continua) y un modelo profundo con dos capas de 10 neuronas cada una (linea punteada). A la derecha se muestra una sección aumentada de la figura de la izquierda. Las marcas rellenas indican los errores realizados por el modelo de 20 neuronas ocultas. Imagen tomada de [36].

presentan en general los datos. Por ejemplo, una cara esta compuesta de ojos, los cuales están compuestos de bordes, etc. por lo que tener varias capas de procesamiento tiene un sentido intuitivo en estos datos.

Veamos ahora el aprendizaje basado en optimización. Se quiere encontrar $f(x;\theta)$ que aproxime datos en un conjunto de puntos $(x_1,y_1),..,(x_n,y_n)$ y se tiene la red definida por $f(x;\theta) = f^{(n)}(f^{(n-1)}(..(f^{(1)}(x)))$. Se plantea el problema como una minimización

$$\hat{\theta} = argmin_\theta \sum_{i=1}^{n} L(f(x_i;\theta),y_i), \tag{2.5}$$

donde $\theta = [W^{(1)},W^{(2)},..,W^{(n)}]$ y L es la función de ajuste (conocida en ingles como loss). Se realiza la optimización mediante el descenso por gradiente.

Lo que se busca en realidad es minimizar el error de predicción esperado:

$$\theta^* = argmin_\theta \mathbb{E}_{x,y}\{L(f(x_i;\theta),y_i)\}. \tag{2.6}$$

Para obtener una buena aproximación del error esperado se necesitan contar con muchos datos. El objetivo del aprendizaje automático es que el algoritmo entrenado funcione bien en datos no conocidos, a esta habilidad se le llama generalización y se mide con el error de generalización o test error.

Sea $L(x,y,\theta) = L_i(x,y,\theta) + \lambda R(\theta)$, donde el primer termino (L_i) penaliza el error de clasificación sobre las muestras de entrenamiento y el segundo termino ($R(\theta)$) impone que el ajuste a los datos se haga de manera suave. λ es una constante que ajusta el balance entre estos dos términos, y se debe ajustar para balancear el error de clasificación y la generalización a nuevos datos.

Para encontrar los pesos óptimos que minimicen L se utiliza descenso por gradiente

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta L(\theta_t), \eta{>}0, \tag{2.7}$$

siendo η el learning rate.

Si el número de muestras $n \gg 1$ calcular

$$\nabla_\theta L(\theta) = \sum_{i=1}^{n} \nabla_\theta L_i(x_i, y_i; \theta) + \lambda \nabla_\theta R(\theta), \tag{2.8}$$

es muy costoso dado que se tienen n términos. Una forma es aproximar utilizando pequeños conjuntos (minibatch), donde en cada paso se utiliza un subconjunto diferente.

2.3.2. Redes Neuronales Convolucionales

Las redes convolucionales son redes neuronales que utilizan, en al menos una de sus capas, la convolución en lugar de la multiplicación general de matrices [14].

Convolución en imágenes

La convolución en imágenes es una operación lineal entre una imagen y un filtro (Figura 2.7), generando una nueva imagen. El valor de cada píxel se calcula como la suma ponderada de los píxeles de la imagen u y un núcleo de convolución h:

$$(u * h)(i,j) = \sum_{k,l} u(i-k, j-l) h(k,l). \tag{2.9}$$

Capa de convolución

Una capa típica de una red de convolución consiste en tres etapas (Figura 2.8):

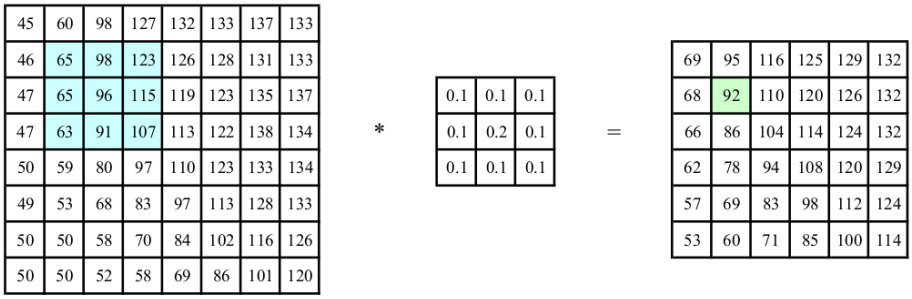


Figura 2.7 La imagen de la izquierda es convolucionada con el filtro del medio creando la imagen de la derecha. Los píxeles en celeste indican la ventana del origen que se convierte mediante la convolución con el filtro en el píxel de destino verde. Imagen tomada de [41].

1. Se realizan varias convoluciones en paralelo, una por cada filtro, generando un conjunto de mapas de activación.
2. A cada mapa de activación se le aplica una función no lineal de activación (ej: ReLU).
3. Se utiliza una función de poolig para reducir la salida de la capa, la cual consiste en comprimir cada mapa de activación por separado.

Alguna de las ventajas de utilizar este tipo de redes son las siguientes:

- Conexiones esparzas: teniendo el núcleo de convolución menor que la entrada se tienen menos operaciones.
- Parámetros compartidos: a diferencia de las capas totalmente conectadas, se comparten los pesos (filtros) en varios elementos de la entrada.
- Equivarianza: si la entrada se traslada, la salida se traslada.
- Diferente tamaño de entrada: a diferencia de las las redes que tienen definida una multiplicación de matrices con un tamaño fijo de matriz, la convolución procesa cualquier tamaño de datos sin la necesidad de modificar la arquitectura.

Pooling

La capa de pooling ayuda a que la representación sea aproximadamente invariante a pequeñas traslaciones de la entrada. Es común insertar de forma periódica capas de pooling entre las capas de convolución para extraer características multi-escala y a la vez para reducir la cantidad de parámetros de la red.

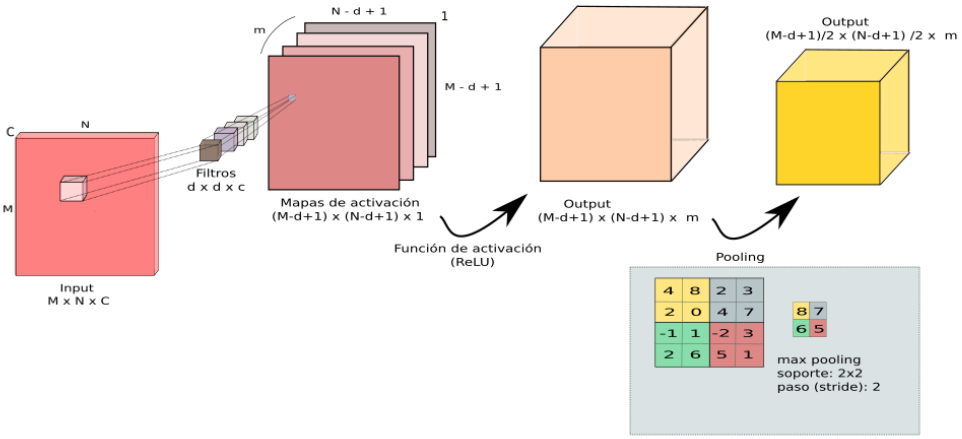


Figura 2.8 Capa de convolución: Dada una imagen de entrada de tamaño $M \times N \times C$ y m filtros de tamaño $d \times d \times c$, se realiza primero la convolución generando m mapas de activación. Luego se aplica una función de activación a cada mapa de activación para luego aplicarle pooling y así reducir el espacio de representación. Por ejemplo, se puede implementar una capa de pooling con filtros de tamaño 2×2 aplicados con un paso (stride) de 2, reduciendo así cada capa por dos a lo largo y ancho. En el recuadro gris hay un ejemplo de pooling (en paticular de Max Pooling).

Volumen de salida

Hay tres hiper-parámetros que controlan el tamaño del volumen de la salida:

- Profundidad, refiere al número de filtros a utilizar.
- Stride, refiere a cuanto se mueve el filtro. Típicamente 1 o 2 (ver Figura 2.9).
- Zero-padding, refiere a cuantos ceros se agregan al volumen de entrada para controlar el volumen de salida (Ver figura 2.10).

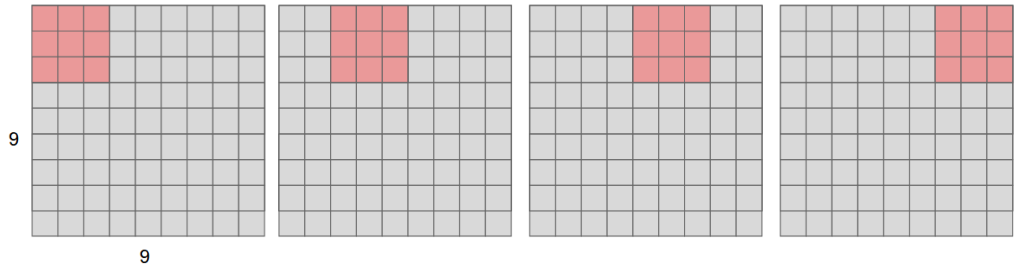
Se calcula entonces el tamaño del volumen de salida en función del volumen de entrada (W), el campo receptivo de las neuronas de la capa de convolución (F), el stride (S) y la cantidad de ceros del zero-padding utilizados en el borde (P)

$$\frac{(W - F + 2P)}{S} + 1. \tag{2.10}$$

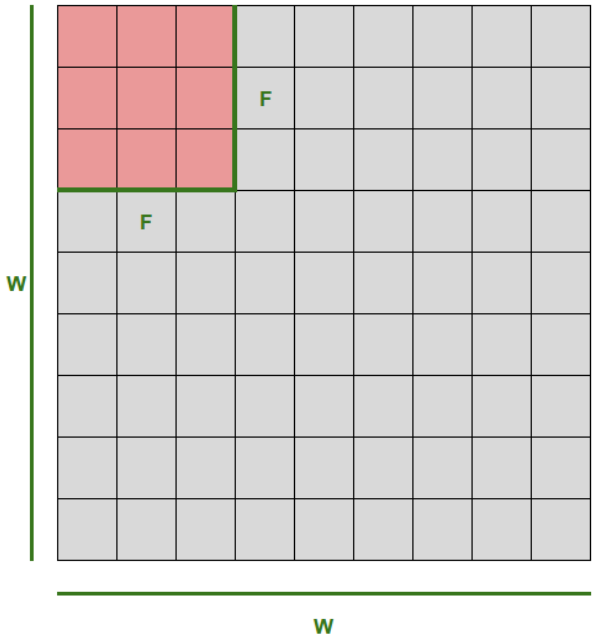
Viéndolo con un ejemplo [32] si se tiene una entrada de tamaño $32 \times 32 \times 32$, 10 filtros de tamaño 5×5, stride de 1 y zero-padding de 2, el tamaño de salida es

$$\frac{(32 - 5 + 2 * 2)}{1} + 1 = 32,$$

teniendo en cuenta la cantidad de filtros, el tamaño final es de $32 \times 32 \times 10$.



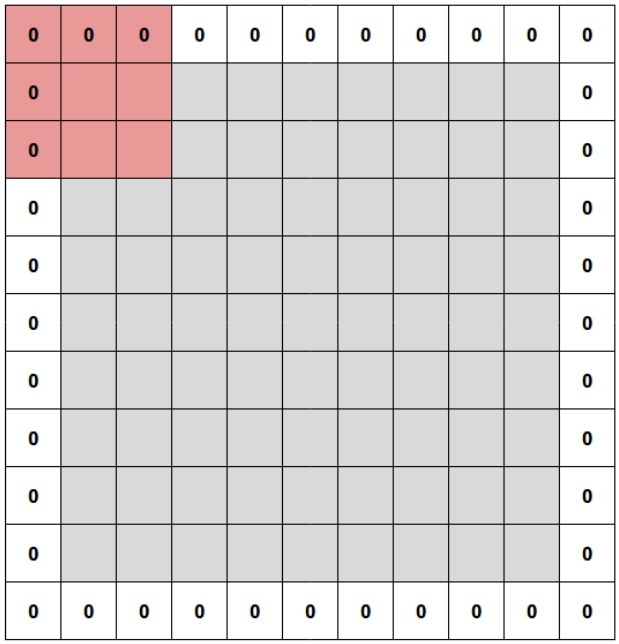
(a) Imagen de entrada: 9×9,
Filtro: 3×3 Stride: 2,
Salida: 4×4.



(b) Tamaño de salida: $(W - F)/\text{stride} + 1$,
Ej:$(9-3)/2+1 = 4$

Figura 2.9 Dimensiones de la salida según el tamaño de la entrada y del *stride*.

Una dificultad que los algoritmos de optimización de redes neuronales deben superar surge cuando el grafo computacional se vuelve extremadamente profundo. Las redes feedforward con muchas capas tienen grafos computacionales muy profundos, lo mismo ocurre con las redes recurrentes, que construyen grafos computacionales muy profundos al aplicar repetidamente la misma operación en cada paso de tiempo de una secuencia temporal larga. La aplicación repetida de los mismos parámetros da lugar a dificultades.



(a) Imagen de entrada: 9×9,
Filtro: 3×3 Stride: 1, Pad de borde 1
Salida: 9×9.

Figura 2.10 Zero-padding: Normalmente se utilizan capas de convolución con stride 1, filtros de tamaño $F \times F$ y zero-padding de $(F-1)/2$ para preservar el tamaño.

Por ejemplo, si se tiene un grafo que contiene un camino que consiste en multiplicar repetidamente por una matriz W, después de t pasos, esto es equivalente a multiplicar por W^t. Suponiendo que W tiene una descomposición en valores propios $W = V\text{diag}(\lambda)V^{-1}$. En este caso simple, es sencillo ver que $W^t = (V\text{diag}(\lambda)V^{-1})^t = V\text{diag}(\lambda)^t V^{-1}$. Cualquier valor propio λ_i que no sea cercano al valor absoluto 1 explotará si es mayor que 1 en magnitud o desaparecerá si es menor que 1 en magnitud. El problema de desvanecimiento y explosión de gradiente refiere al hecho de que los gradientes a través de dicho grafo también se escalan de acuerdo con $diag(\lambda)^t$. Los gradientes que se desvanecen dificultan saber en qué dirección deben moverse los parámetros para mejorar la función de costo, mientras que los gradientes en explosión pueden hacer que el aprendizaje sea inestable.

2.3.3. Redes Neuronales Recurrentes y Recursivas

Las redes neuronales recurrentes son una familia de redes neuronales para procesar datos secuenciales. Al igual que una red convolucional es una red neuronal especializada en procesar una grilla (p.e. imágenes), una red neuronal recurrente (RNN) es una red neuronal

especializada en procesar una secuencia de valores $x^{(1)}, ..., x^{(\tau)}$. Las redes recurrentes pueden procesar secuencias de longitud variable, utilizan la información de la salida de la red en el tiempo anterior o el valor de un estado oculto interno a la red.

Muchas de las redes neuronales recurrentes utilizan la ecuación 2.11 para definir los valores de las unidades ocultas

$$h_t = f(h_{(t-1)}, x_t; \theta), \tag{2.11}$$

donde h_t representa el estado interno oculto en el tiempo t, $h_{(t-1)}$ el estado interno oculto en el tiempo $(t-1)$, x_t la entrada en el tiempo actual t y f la función de recurrencia, dependiente de los paramentos θ e independiente de t.

Una red neuronal recurrente puede pensarse como múltiples copias de la misma red, donde cada una le pasa un mensaje (estado) al sucesor.

En la Figura 2.11 se muestra una arquitectura básica de RNN. Típicamente se agregarán características arquitectónicas adicionales, tales como capas de salida que leen información fuera del estado h para hacer predicciones.

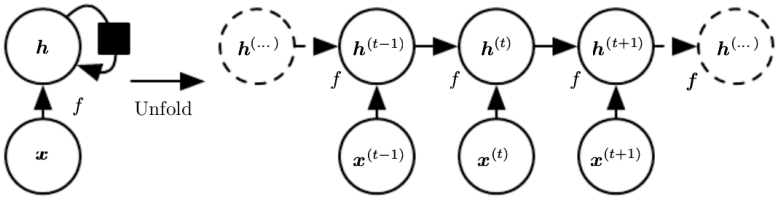


Figura 2.11 Red Neuronal Recurrente sin salidas. Simplemente procesa la información de la entrada x incorporándola al estado h que se transmite a través del tiempo. A la izquierda se muestra la arquitectura de la red, el cuadrado negro indica el paso de un solo paso de tiempo. A la derecha se presenta la misma red vista como un grafo computacional desplegado, donde cada nodo ahora está asociado con una instancia de tiempo particular. Imagen tomada de [14].

Las redes neuronales recursivas son un tipo de generalización de las redes recurrentes (Figura 2.12). Tienen un tipo diferente de gráfico computacional, estructurado como un árbol profundo en lugar de la estructura en forma de cadena de las redes recurrentes.

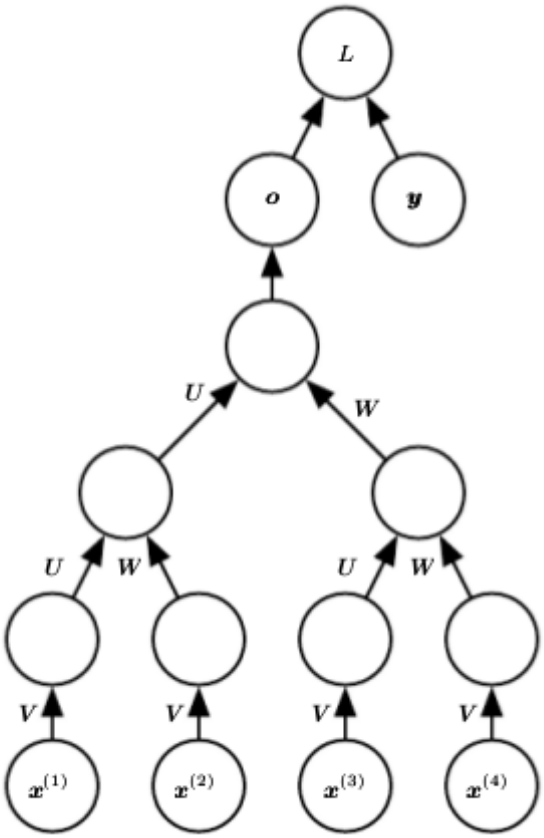


Figura 2.12 Red Neuronal Recursiva, tiene un grafo computacional que generaliza la red recurrente, pasando de un grafo de tipo cadena a un árbol. Una secuencia de tamaño variable $x_{(1)}, x_{(2)}, ..., x_{(t)}$ se puede asignar a una representación de tamaño fijo (la salida o), con un conjunto fijo de parámetros (las matrices de peso U, V, W). La figura ilustra un caso de aprendizaje supervisado en el que se proporciona un objetivo y que está asociado con toda la secuencia. Imagen tomada de [14].

2.4. Medidas de error

Para estudiar el rendimiento de los algoritmos de super-resolución analizados y utilizados en este trabajo es necesario previamente definir las métricas que se van a utilizar.

PSNR - Peak Signal to Noise Ratio

El peak signal to noise ratio (PSNR) es una medida cuantitativa de la diferencia entre dos imágenes, comúnmente utilizada para medir la calidad entre una imagen degradada y la original [41]. Sea $I(x)$ la imagen original y $\hat{I}(x)$ es la imagen a comparar, el PSNR se deriva del error cuadrático medio

$$MSE = \frac{1}{n}\sum_{x}(I(x) - \hat{I}(x))^2, \tag{2.12}$$

o lo que es equivalente, a la raíz error cuadrático medio (RMS)

$$RMS = \sqrt{MSE}.$$

El PSNR se define como

$$PSNR = 10\log_{10}\left(\frac{I_{max}^2}{MSE}\right) = 20\log_{10}\left(\frac{I_{max}}{RMSE}\right), \tag{2.13}$$

donde I_{max} es el máximo valor posible de la señal (ej: 255 para imágenes de 8 bits). El valor de PSNR se aproxima a infinito cuando el MSE se aproxima a cero, es decir, un valor mayor de PSNR significa mayor similitud de las imágenes que se están comparando.

SSIM - Structural Similarity Index Measure

El valor structural similarity index measure (SSIM) es otra de las métricas de calidad conocida para medir la similitud ente dos imágenes. Fue desarrollada por Wang et al. [47] y es considerada por tener una buena correlación con el sistema de percepción visual humano. En vez de utilizar métodos tradicionales de medida de error (ver Figura 2.13), modela cualquier distorsión como una combinación de tres factores que son la luminiscencia, el contraste y la estructura (Figura 2.14).

Se calcula la luminiscencia como el promedio de la intensidad:

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{2.14}$$

Suponiendo X e Y dos imágenes, la función de comparación de la luminiscencia $l(X,Y)$ es una función de μ_x y μ_y.

Luego, se remueve de la señal el promedio de la intensidad. $X - \mu_x$ corresponde a la proyección del vector X en el hiperplano definido por

$$\sum_{i=1}^{N} x_i = 0.$$

Se utiliza la desviación estándar (raíz cuadrada de la varianza) como el estimado del contraste de la señal. Un estimado imparcial esta dado por

$$\sigma_x = \left(\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \mu_x)^2 \right)^{1/2}. \tag{2.15}$$

La comparación de contraste $c(X,Y)$ es la comparación de σ_x y σ_y.

Por último, la señal se normaliza por su propia desviación estándar. La comparación de estructura, $s(X,Y)$, se realiza en las señal normalizada $(X - \mu_x)/\sigma_x$ y $(Y - \mu_y)/\sigma_y$.

Se integran las tres componentes para así obtener la medida de similaridad completa:

$$SSIM(X,Y) = l(X,Y)c(X,Y)s(X,Y). \tag{2.16}$$

l, c y s se definen en la ecuación 2.17. Un punto importante es que los tres componentes son relativamente independientes, por ejemplo, el cambio de luminiscencia y/o contraste no afecta la estructura de la imagen.

$$\begin{aligned} l(X,Y) &= \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \\ c(X,Y) &= \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \\ s(X,Y) &= \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}. \end{aligned} \tag{2.17}$$

La constante C_1 se incluye para evitar la inestabilidad cuando $\mu_x^2 + \mu_y^2$ tiene un valor cercano a cero, específicamente $C_1 = (K_1 L)^2$, donde L es el valor rango dinámico de los píxeles (255 para imágenes en escala de grises de 8 bits), y $K_1 \ll 1$ es una constante pequeña. Las mismas consideraciones se tienen para C_2 y C_3.

El valor máximo es 1 y se da unicamente cuando $X = Y$.

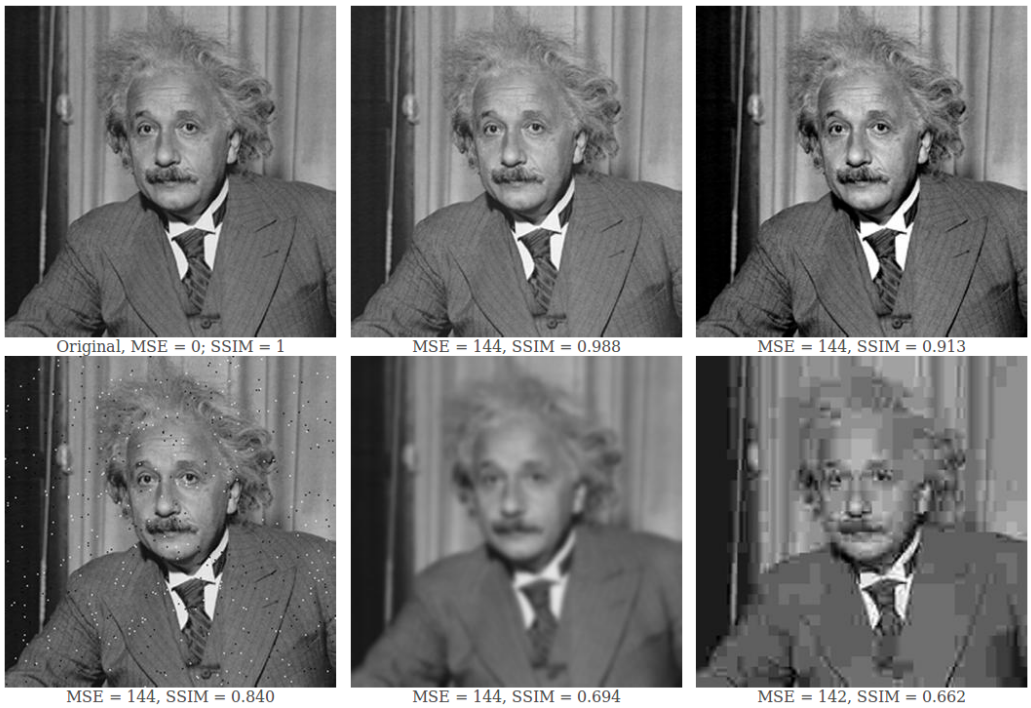


Figura 2.13 **Ejemplo MSE vs SSIM.** Todas las imágenes distorsionadas tienen aproximadamente los mismos valores de error cuadrático medio (MSE) con respecto a la imagen original, pero la calidad es muy diferente. La medida SSIM da una mejor noción de la similitud entre las imágenes. Imagen tomada de [47].

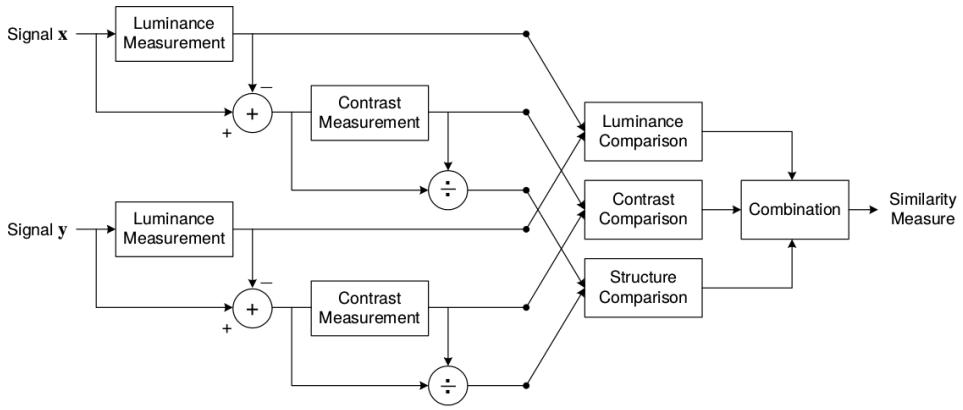


Figura 2.14 Diagrama del sistema de medida de similitud estructural SSIM. Figura tomada de [47].

Capítulo 3

Trabajo propuesto

En este capítulo se presenta el estudio de las diferentes técnicas de súper resolución, en particular para su posterior utilización con imágenes satelitales. En este capítulo y los siguientes, nos enfocaremos en un subconjunto de los algoritmos de SR existentes. La selección de los algoritmos a estudiar se basa en su relevancia, actualidad, y disponibilidad de código o detalles de implementación provista por los autores. Se estudian algoritmos para los dos enfoques principales, es decir, para SR mono-imagen y multi-imagen.

Si bien existe un gran número de técnicas propuestas en la literatura, es común que los detalles de implementación y descripción de los algoritmos sea insuficiente. Aun peor, la mayoría de los trabajos publicados no acompañan la publicación con la publicación del código utilizado. Una de los principales aportes de este trabajo es proveer y publicar de manera abierta los códigos desarrollados.

En la primera sección de este capítulo se presenta la importancia de la elección del método de interpolación para problemas de súper resolución basados en redes neuronales, en la segunda sección se presentan las redes utilizadas para problema de súper resolución, tanto mono-imagen como multi-imagen y en la última sección se presenta el algoritmo variacional (STV) para súper resolución.

3.1. Interpolación

Como vimos en el capítulo 2, el problema de súper resolución se puede definir de forma discreta como sigue, dada una imagen continua $u(x,y)$, consideramos dos muestras discretas $u^h(i,j)$ y $u^{2h}(i,j)$ asociadas a la grilla discreta Ω^h y Ω^{2h} respectivamente. La primera se considera la representación de alta resolución de la segunda ya que se muestrea con el doble de la resolución.

El problema de súper resolución se formula como la estimación a partir de una muestra de baja resolución u^{2h} de su contraparte de alta resolución u^h. Transformar imágenes discretas del dominio Ω^h a Ω^{2h} y viceversa requiere de la previa definición de operaciones de *down-sampling* y *over-sampling*.
Definiendo I_{Ω^h} como el conjunto de imágenes discretas en el dominio Ω^h, se define el operador de Prolongación como el mapeo de la cuadrícula gruesa a la cuadrícula fina

$$P : I_{\Omega^{2h}} \rightarrow I_{\Omega^h}.$$

Por otro lado, la transformación inversa es realizada por el operador de Restricción

$$R : I_{\Omega^h} \rightarrow I_{\Omega^{2h}}.$$

La restricción R se define entonces como sigue,

$$R(u^h) = R_0(K_h * u^h) \tag{3.1}$$

donde K_h es un kernel definido en el dominio de alta resolución y $R_0(u^h)$ es el operador de *down-sampling*,

$$R_0(u^h)(i,j) = u^h(2i,2j). \tag{3.2}$$

De forma similar se define la prolongación P utilizando la definición de *over-sampling*,

$$P(u^{2h}) = K^h * P_0(u^{2h}), \tag{3.3}$$

$$P_0(u^{2h})(i,j) = \begin{cases} u^{2h}(i/2, j/2) & \text{Si } i,j = \dot{2} \\ 0 & \text{en otro caso} \end{cases} \tag{3.4}$$

Los operadores de restricción y prolongación se definen de la forma recién mencionada para así controlar los efectos de aliasing durante la restricción y evitar la generación artificial de altas frecuencias durante la prolongación. Como se puede observar en la Figura 3.1 , los operadores de restricción y prolongación llevan respectivamente el espacio de frecuencia al intervalo $[\frac{-1}{2h}, \frac{1}{2h}]$ y $[\frac{-1}{4h}, \frac{-1}{4h}]$. En este proceso las frecuencias $\{v_0, v_0 \pm (\frac{1}{2h})\}$ en el intervalo $[\frac{-1}{2h}, \frac{1}{2h}]$ se mapean a las mismas frecuencias $v_0 \in [\frac{-1}{4h}, \frac{-1}{4h}]$.

Es entonces que la definición del kernel de suavizado K^h es de crucial importancia para el control de aliasing [19, 8]. Algunos de los kernels más populares son los siguientes:

- Bi-linear: $K = \frac{v^t v}{\|v\|^2}$, $v = \left[\frac{1}{2}, 1, \frac{1}{2}\right]$.
- Bi-cubic: $K = \frac{v^t v}{\|v\|^2}$, $v = \left[\frac{-1}{16}, 0, \frac{9}{16}, 1, \frac{9}{16}, 0, \frac{1}{16}\right]$.

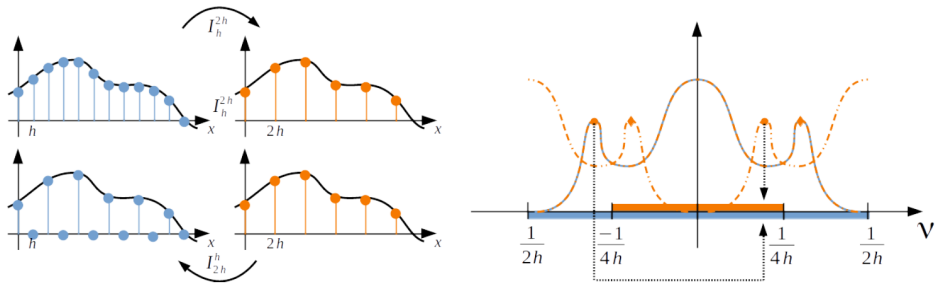


Figura 3.1 Operadores de restricción y prolongación. Figura tomada de [8].

- Seven: $K = \frac{1}{4}\left[\left[\frac{1}{2},\frac{1}{2},0\right]^t,\left[\frac{1}{2},1,\frac{1}{2}\right]^t,\left[0,\frac{1}{2},\frac{1}{2}\right]^t\right]$.

Un método alternativo para la interpolación puede ser implementado en el dominio de Fourier [1]. Sea $S(z_x,z_y)$ la representación del operador de muestreo de factor (z_x,z_y) definido por

$$S_{z_x,z_y}[u](i,j) = U(z_x j, z_y i) \tag{3.5}$$

donde U es la interpolación de Shannon:

$$U(x,y) = \frac{1}{HW}\sum_{\substack{\alpha\in\left[\frac{-W}{2},\frac{W}{2}\right]\\ \beta\in\left[\frac{-H}{2},\frac{H}{2}\right]}} \varepsilon_M(\alpha)\varepsilon_N(\beta)\hat{u}(\alpha,\beta)e^{i2\pi\left(\frac{\alpha x}{M}+\frac{\beta y}{N}\right)}. \tag{3.6}$$

$\hat{u}$ denota la transformada de fourier discreta de u y

$$\forall P\in\mathbb{N},\forall\gamma\in\mathbb{Z},\varepsilon_P(\gamma) = \begin{cases} 1/2 & \text{si } |\gamma| = P/2 \\ 1 & \text{sino.} \end{cases} \tag{3.7}$$

Es de destacar que si bien a primera vista puede parecer que utilizar un kernel Gaussiano presenta buenos resultados, no es apropiado pues el decaimiento que presenta a altas frecuencias no es suficiente para controlar el aliasing introducido por el operador P_0.

Problema inverso: Las operaciones de prolongación y restricción recién definidas pueden ser expresadas como operaciones lineales sobre el conjunto de píxeles de una imagen. Más precisamente, si se representan las imágenes discretas como columnas de vectores, se puede escribir la restricción y la prolongación como multiplicación de matrices. Las matrices P y R tienen dimensión $[HW \times \frac{H}{2}\frac{W}{2}]$ y $[\frac{H}{2}\frac{W}{2} \times HW]$ respectivamente. Notar que incluso para dominios Ω^h de resolución pequeña el tamaño de las matrices P y R puede ser muy grande. Por simplicidad se asume que el dominio de alta resolución Ω^h está definido por W y H par, lo que se puede asegurar mediante preprocesamiento de *padding*.

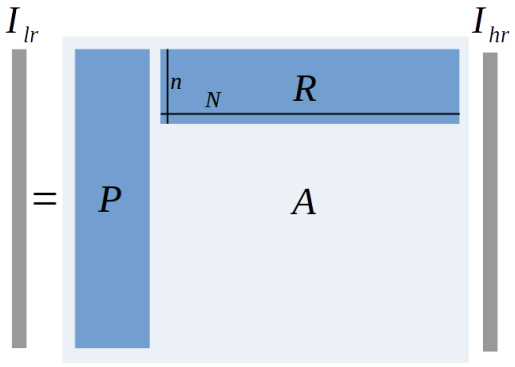


Figura 3.2 Ilustración de las operaciones de restricción y prolongación como multiplicación de matrices.

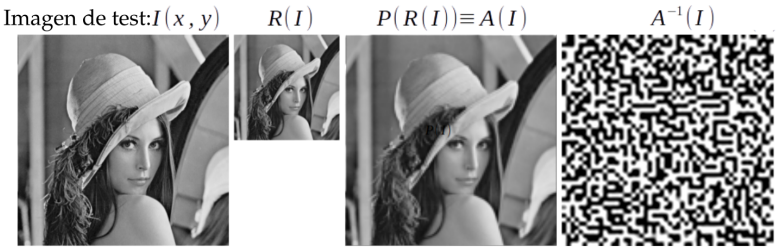


Figura 3.3 De izquierda a derecha: (i) Un ejemplo de imagen de test (I_{HR}), (ii) mapeo de la imagen de entrada de I_{Ω^h} a $I_{\Omega^{2h}}$, (iii) interpolación de $R(I_{HR})$ en Ω^h, y por último (iv) el resultado de la inversión de A. En este experimento se utilizó *Backslash* de Matlab para la inversión de A.

Definiendo la matriz $A = PR$ se puede expresar la imagen de baja resolución I_{lr}, interpolada en Ω^h, como la transformación lineal $I_{lr} = AI_{hr}$ como se ilustra en la figura 3.2. Por lo tanto, el problema de súper resolución se puede interpretar como una inversión de la transformación A. Analizando la estructura de la matriz A, es fácil de ver porque el problema de súper resolución es un desafío. $N = WH$ denota el numero de píxeles en el dominio de alta resolución Ω^h y $n = \frac{HW}{4}$ el número de píxeles en la grilla grilla de baja resolución Ω^{2h}.

El rango de la matriz A de $N \times N$ esta acotado por $n = \frac{N}{4}$, por ello A es una matriz no invertible. Más aún, al menos $N - n$ de los valores propios de A son nulos, lo que significa que que tres cuartas partes del espacio de entrada se colapsan en el núcleo de la matriz. La figura 3.3 muestra una imagen de test y su versión de baja resolución. La imagen de la derecha muestra el resultado de obtener una solución simple de súper resolución calculando el seudo-inverso de A mediante métodos numéricos estándar, por ejemplo, aplicando soluciones numéricas eficientes como el *Backslash* de Matlab.

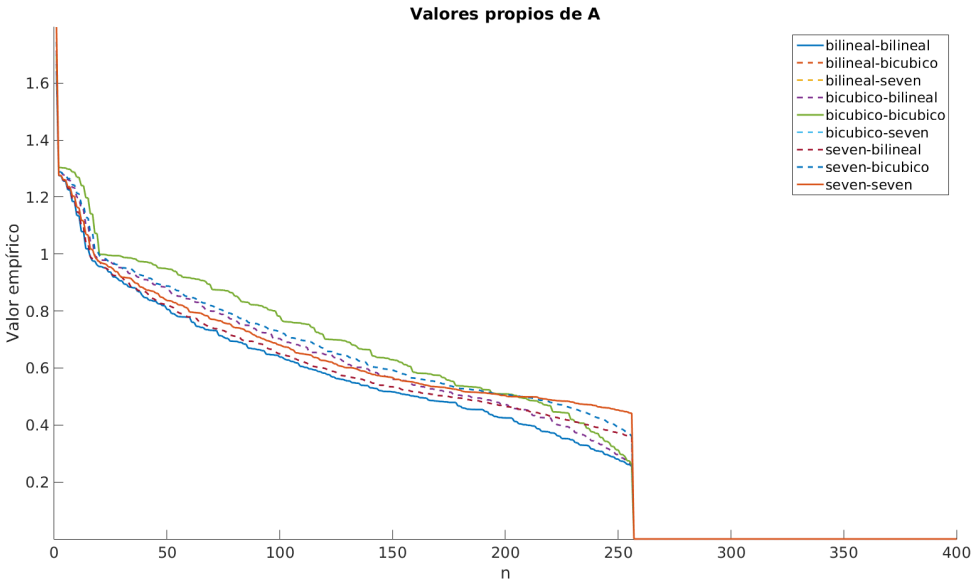


Figura 3.4 El producto de las matrices R y P se pueden expresar como una transformación lineal de A en el dominio de alta resolución. El gráfico ilustra los valores propios de A para diferentes combinación de los núcleos de restricción y prolongación. Se indica primero el utilizado para la restricción y luego para la prolongación.

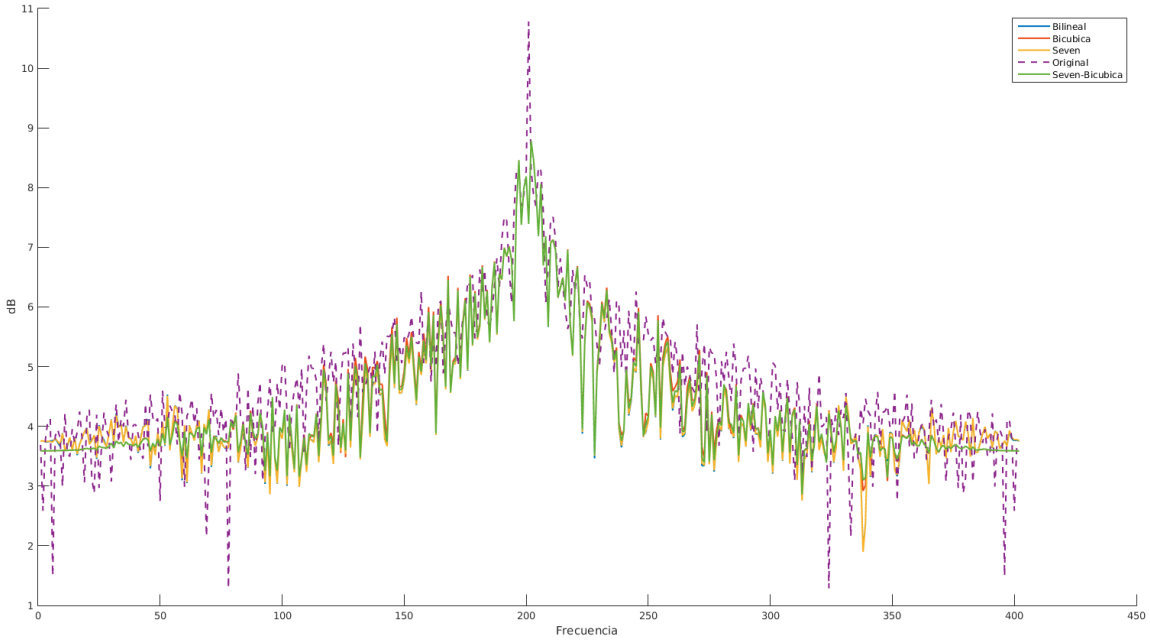
En la figura 3.4 se muestran los valores propios de A cuando se utilizan diferentes núcleos para las definiciones P y R. Los mejores resultados se dan cuando con n cercano a cero los valores propios son altos, y para n mayor a 256 son cero.

Se puede ver en la figura 3.5 el dominio de frecuencia de una imagen de ejemplo y los dominios de frecuencia cuando se realiza la reducción y prolongación con el mismo núcleo o cuando se realiza la reducción con un núcleo y la prolongación con otro, mientras en la figura 3.6 se ven las imágenes del mismo ejemplo.

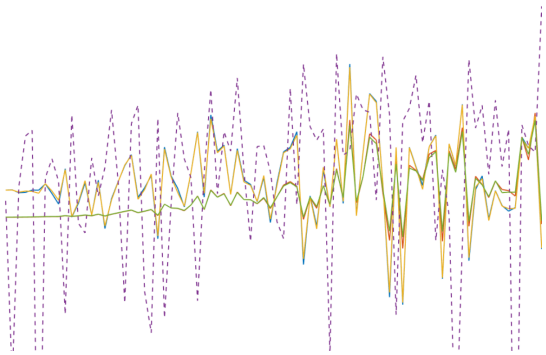
3.2. Súper Resolución basada en redes neuronales profundas

Si bien la inversión explicita de $I_{LR} \rightarrow I_{HR}$ es extremadamente desafiante, métodos de DNN han demostrado muy buenos resultados para resolver este problema. Los métodos supervisados utilizan pares de muestras $\{I_{LR}, I_{HR}\}_i$ para optimizar millones de parámetros θ y aprender un mapeo empírico $f_\theta(I_{LR}) \rightarrow I_{HR}$.

Se han propuesto muchas arquitecturas de red alternativas para realizar la súper resolución. A pesar de ello, la gran mayoría de dichos métodos [29, 42, 9, 38, 28, 31, 23] consideran



(a) En la etiqueta se muestra el núcleo utilizado para la restricción y para la prolongación, cuando se indica uno solo significa que se utiliza el mismo, en caso contrario se indica primero el utilizado para la restricción y luego para la prolongación (Seven-Bicubica).



(b) Ampliación del gráfico de la imagen 3.5a.

Figura 3.5 **Reducción y Prolongación en el dominio de frecuencia** Se muestra la Transformada de Fourier de la imagen original y de las imágenes luego de aplicarle la restricción y prolongación utilizando el mismo núcleo en algunos casos y distinto núcleo en otro.

(a) Imagen Original

(b) Detalles de la imagen original

(c) Detalles luego de aplicarle a (a) la reducción y prolongación (Seven)

(d) Detalles luego de aplicarle a (a) la reducción (Seven) y prolongación (Bicubica)

Figura 3.6 Imágenes de ejemplo luego de realizar Reducción y Prolongación.

una interpolación bi-cúbica como un primer paso. Este primer paso aumenta el tamaño de la imagen de entrada (LR) a la forma de salida deseada como se ilustra en la Figura 3.7, y la red queda entrenada consiguiendo mejorar la entrada borrosa para que coincida con su muestra original (HR). Por lo tanto, los sub-muestreos de interpolación y los núcleos de muestreo son de crucial importancia durante el proceso de aprendizaje.

3.2.1. Súper Resolución mono-imagen.

A continuación se presenta la red "Deeply-Recursive Convolutional Network for Image Super-Resolution"propuesta por Kim et al. [29]. Proponen una red neuronal convolucional profunda con una capa recursiva de hasta 16 recurrencias, con el fin de obtener mejores resultados sin necesidad de introducir nuevos parámetros para las capas de convolución. Para la SR, el campo receptivo de una red convolucional determina la cantidad de información que se puede obtener para inferir los componentes faltantes de alta frecuencia. Dado que la SR es un problema inverso mal condicionado, es decir, no se puede obtener toda la información a partir de los datos disponibles, la recopilación y el análisis de mayor cantidad de píxeles puede generar mayor cantidad de datos para así estimar los datos perdidos en la baja resolución.

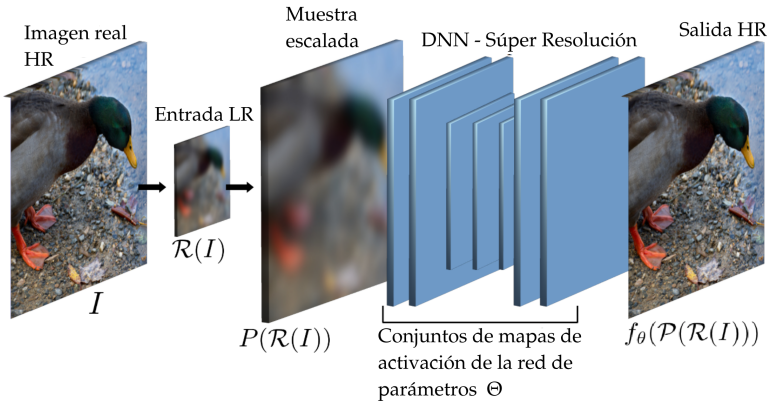


Figura 3.7 Esquema clásico de una DNN para súper resolución. I representa una imagen (real de alta resolución), la cual es submuetreada $(R(I))$ simulando una muestra de baja resolución. El primer paso es aumentar la escala de la imagen de baja resolución $(P(R(I)))$, luego la imagen es mejorada utilizando la red DNN con parámetros θ. Utilizando datos de entrenamiento, se optimizan los parámetros θ para minimizar $\|I - f_\theta(P(R(I)))\|_2$.

Una forma de extender el campo receptivo de una red es aumentar la profundidad de la misma, ya sea agregando una o varias capas de convolución o utilizando una capa de pooling. Ambos enfoques tienen inconvenientes, una capa de convolución introduce mayor cantidad de parámetros y una capa pooling comúnmente descarta parte de la información. Para problemas de SR los detalles de la imagen son muy importantes, por tal motivo, no es bueno utilizar pooling. Una alternativa es aumentar la profundidad agregando nuevas capas con un peso diferente para cada una introduciendo una mayor cantidad de parámetros, lo que puede llevar a necesitar más datos o a sobre-ajustarse. Para evitar estos inconvenientes es que proponen la utilización de una red convolucional profunda recursiva. La red aplica repetidamente la misma capa de convolución tantas veces como se desee, haciendo así que no aumente el número de parámetros.

La red tiene un campo receptivo de 41×41 por lo que optimizarla utilizando descenso estocástico de gradiente resulta muy complejo debido a que no converge fácilmente. A su vez, el aprendizaje de las dependencias entre los píxeles con una sola capa de pesos también resulta complejo. Para lidiar con estas dos dificultades proponen que todas las recursiones sean supervisadas, es decir, utilizan luego de cada recursión los mapas de características para reconstruir la imagen objetivo (HR) utilizando el mismo método de reconstrucción para todas las recursiones. Dado que en cada recursión se predice una imagen HR distinta, se combinan al final todas las predicciones de los diferentes niveles de la recursión para obtener una predicción más precisa. Otra propuesta para lidiar con las dificultades es la utilización de

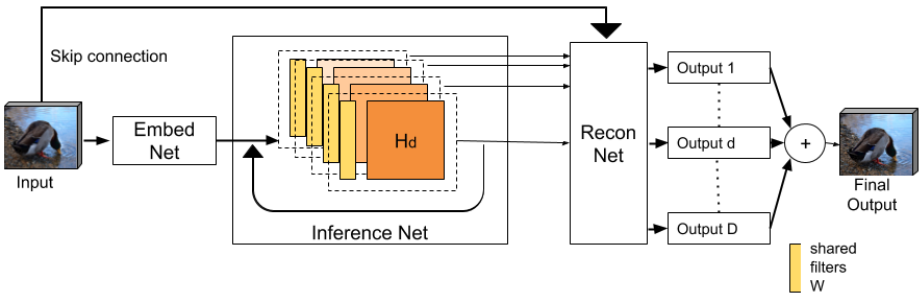


Figura 3.8 Arquitectura propuesta por Kim et al. Kim et al. [29].

una conexión salteada desde la entrada de la red a la capa de reconstrucción, esto es de gran utilidad debido a la alta correlación entre la entrada y la salida de la red en los problemas de SR.

Proponen un modelo compuesto por tres sub-redes (Figura 3.8): La *Embedding net* representa la entrada como un conjunto de mapas de características, la *Inference net* es la componente principal, encargada de llevar a cabo la súper resolución y por último la *Reconstruction net* transforma el mapa de características de las salidas de la recursión al espacio original (1 o 3 canales).

El modelo matemático propuesto es el siguiente, la red toma como entrada una imagen x interpolada (para obtener al tamaño final deseado) y predice la imagen de salida y. El objetivo es obtener un modelo f que prediga los valores $\hat{y} = f(x)$, donde $\hat{y}$ es el estimado de la salida real y. Definen las funciones f_1, f_2, f_3 de cada sub-red, siendo el modelo final la composición de las tres: $f(x) = f_3(f_2(f_1))$.

La primer sub-red *Embedding net* se define como,

$$H_{-1} = max(0, W_{-1} * x + b_{-1})$$
$$H_0 = max(0, W_0 * H_{-1} + b_0)$$
$$f_1(x) = H_0$$

La sub-red *Inference net* (f_2) toma como entrada la matriz H_0 y computa la matriz de salida H_D. Se utilizan los mismos pesos y bias para todas las operaciones. Sea g la función que modela la recursión de la capa de recursión, se tiene:

$$g(H) = max(0, W * H_{d-1} + b)$$

La relación de recurrencia es entonces:

$$H_d = g(H_{d-1}) = max(0, W * H_{d-1} + b) \text{ para d = 1,..,D.}$$

f_2 es equivalente a la composición de la misma función elemental g:

$$f_2(H) = (g \circ g \circ .....\circ)g(H) = g^D(H)$$

La sub-red *Reconstruction net* f_3 toma como entrada el estado oculto H_D y devuelve la imagen de alta resolución.

$$H_{D+1} = max(0, W_{D+1} * H_D + b_{D+1})$$
$$\hat{y} = max(0, W_{D+2} * H_{D+1} + b_{D+2})$$
$$f_3(H) = \hat{y}$$

Para evitar el desvanecimiento o explosión de gradiente le agregan al modelo la supervisión de la recursión. Bajo la asunción de que la misma representación se puede usar una y otra vez durante las convoluciones de la red de inferencia, la misma red de reconstrucción se utiliza para predecir las imágenes HR para cada recursión, es decir, la red de reconstrucción ahora genera *D* predicciones y todas se supervisan simultaneámente durante el entrenamiento. Luego, todas las predicciones se promedian durante la etapa de test. Dado que en los problemas de SR la entrada y la salida de los algoritmos tienen una fuerte correlación, proponen agregar una conexión salteada directa desde la entrada a la red de reconstrucción.

Cada predicción intermedia bajo la supervisión recursiva se encuentra dada por:

$$\hat{y}_d = f_3(x, g^{(d)}(f_1(x))),$$

para d = 1,2,....D, donde f_3 toma dos entradas, una desde la conexión salteada. La salida final es el promedio de pesos de todas las predicciones intermedias:

$$\hat{y} = \sum_{d=1}^{D} w_d . \hat{y}_d$$

donde w_d denota los pesos de las predicciones reconstruidas de cada estado oculto intermedio durante la recursión.

Para el entrenamiento, dado un conjunto de entrenamiento $\{x^i, y^i\}_{i=1}^N$, el objetivo es encontrar el mejor modelo f que prediga correctamente los valores $\hat{y} = f(x)$. En la configuración de regresión de mínimos cuadrados, típica en SR, se minimiza el error cuadrático medio $\frac{1}{2}||y - f(x)||^2$ promediado durante el entrenamiento, lo que favorece el PSNR. Con la supervisión recursiva se tienen $D+1$ objetivos para minimizar, D salidas de la capa de recursión más la salida final.

Para salidas intermedias, se tiene la función de loss,

$$l_1(\theta) = \sum_{d=1}^{D} \sum_{i=1}^{N} \frac{1}{2DN} ||y^{(i)} - \hat{y_d}^{(i)}||^2$$

donde θ denota el conjunto de parámetros y $\hat{y_d}^{(i)}$ es la salida de la recursión número d. Para la salida final, se tiene

Figura 3.9 Imágenes ejemplo de la base de datos ScSR [49] propuestas en [29] como base de entrenamiento.

$$l_2(\theta) = \sum_{i=1}^{N} \frac{1}{2N} ||y^{(i)} - \sum_{d=1}^{D} w_d * y_d^{(i)}||^2$$

Por lo que la función final de loss $L(\theta)$, es:

$$L(\theta) = \alpha l_1(\theta) + (1-\alpha) l_2(\theta) + \beta ||\theta||^2$$

donde α denota la importancia del objetivo en las salidas intermedias y β denota el multiplicador de la disminución del peso. Establecer un valor de α alto hace estable el procedimiento de entrenamiento ya que las primeras recurrencias convergen fácilmente. A medida que el entrenamiento avanza, α debe decaer para aumentar el rendimiento de la salida final.

Para el entrenamiento utilizan 91 imágenes propuestas en [49]. Para testing, utilizan cuatro datasets, Set5 y Set 14, usados usualmente como benchmarks, B100 que consiste en imágenes naturales y Urban 100 que consiste en imágenes urbanas (Figura 3.9).

3.2.2. Súper resolución multi-imagen.

A continuación se presenta la red *Detail-revealing Deep Video Super-resolution* de Xin Tao et al. [42] para realizar SR utilizando múltiples imágenes. Se realiza tanto la estimación de movimiento como la fusión de imágenes utilizando una red neuronal *end-to-end*. Diseñan un red entrenable de punta a punta, la cual toma una secuencia de largo N_F de imágenes en LR y produce una imagen I_0^H en HR. La red esta compuesta por tres módulos: (i) Estimación de movimiento, modulo responsable de estimar el movimiento entre las imágenes. (ii) Compensación de movimiento, modulo responsable de alinear las imágenes utilizando la estimación del movimiento calculado. (iii) Fusión de detalles, modulo responsable de aumentar la escala de la imagen y agregar los detalles a la misma.

El modulo de estimación de movimiento toma dos frames de baja resolución y produce el campo de movimiento LR:

$$F_{i \to j} = Net_{ME}(I_i^L, I_j^L, \theta_{ME}), \tag{3.8}$$

donde $F_{i\to j} = (u_{i\to j}, v_{i\to j})$ es el flujo de movimiento de la imagen I_i^L a la I_j^L y θ_{ME} es el conjunto de parámetros.

Si bien probaron FlowNet [10] utilizan el modulo de compensación de movimiento (MCT) de VESPCN [4] dado que tiene menos parámetros, lo que lo hace más veloz.

Compensación de movimiento (Capa SPMC)

Crean una capa que utiliza la información sub-pixel para realizar la compensación de movimiento y a la vez mejorar la resolución. Se define como

$$J^H = Layer_{SMPC}(J^L, F; \alpha), \tag{3.9}$$

donde J^L es la imagen de entrada LR y J^H la imagen de salida HR, F es el flujo óptico y α el factor de escala.

La capa SMPC contiene dos sub-modulos: el primero, llamado *Sampling Grid Generator*, calcula las coordenadas transformadas de acuerdo al flujo óptico estimado $F = (u, v)$ como

$$\begin{pmatrix} x_p^s \\ y_p^s \end{pmatrix} = W_{F,\alpha} \begin{pmatrix} x_p \\ y_p \end{pmatrix} = \alpha \begin{pmatrix} x_p + u_p \\ y_p + v_p \end{pmatrix}, \tag{3.10}$$

donde p representa las coordenadas en LR, x_p e y_p son las dos coordenadas del píxel p, u_p y v_p los vectores de movimiento estimados en el paso anterior, $W_{F,\alpha}$ son las transformadas de las coordenadas (dependen del flujo F y del factor de escala α) y x_p^s e y_p^s son las coordenadas transformadas en el espacio ampliado, como se puede ver en la figura 3.10.

El segundo módulo, llamado *Differentiable Image Sampler*, donde la imagen de salida se construye en el espacio ampliado de acuerdo a x_p^s e y_p^s. El resultado es la imagen J_q^H:

$$J_q^H = \sum_{p=1} J_p^L M(x_p^s - x_q) M(y_p^s - y_q), \tag{3.11}$$

donde q indexa los pixels en la imagen HR, x_q e y_q son las coordenadas para el píxel $q$en la grilla HR, $M(.)$ es el kernel de muestreo, en este caso bilineal, $M(x) = max(0, 1 - |x|)$.

Fusión de detalles

Dado que J_i^H tiene tamaño HR, el costo computacional es un problema. J_i^H es dispersa y la mayoría de los píxeles son cero. Esto genera que la red sea muy grande para poder capturar los patrones en J_i^H. No es posible utilizar una interpolación simple para rellenar los píxeles desconocidos dado que los valores interpolados dominan durante el entrenamiento. Por último, hay que tener especial atención con el uso de la imagen de referencia. Por un lado,

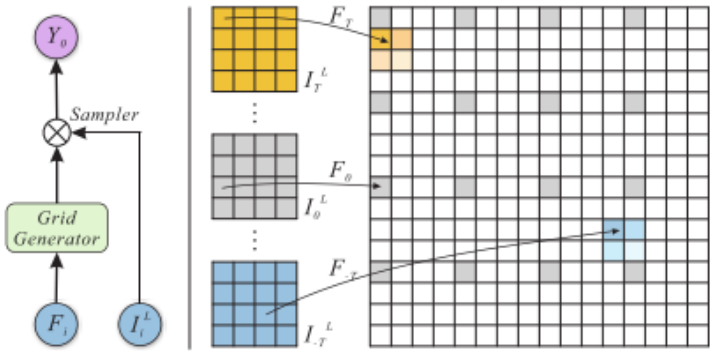


Figura 3.10 Capa SMPC (Factor de escala 4) A la izquierda se encuentra el diagrama de la capa y a la derecha se encuentra la ilustración de la capa SPMC. Figura tomada de [42].

la imagen de referencia es la guía para la SR por lo que la salida HR es consistente con la imagen de referencia en términos de estructura de la imagen. Pero, a su vez, en búsqueda de darle énfasis a la imagen de referencia, se puede generar un descuido con las otras iamgenes, quedando como un sistema de SR de una única imagen.

Arquitectura de la red

Proponen una estructura del tipo encoder−decoder con conexiones salteadas para evitar los problemas recién mencionados. La sub-red *encoder* reduce el tamaño de la imagen de entrada (a $1/2$ en este caso), permitiendo reducir el costo computacional, también ayuda a que el mapa de características sea menos esparzo para poder utilizar la información sin necesidad de que la red sea demasiado profunda. Las conexiones salteadas se utilizan en todas las etapas para acelerar el entrenamiento. Se agrega un modulo ConvLSTM [48] en el medio para trabajar la entrada secuencial.

La estructura de la red incluye:

$$\begin{aligned} f_i &= Net_E(J_i^H \theta_E), \text{ Encoder,} \\ g_i, s_i &= convLSTM(f_i, s_{i-1}, \theta_{LSTM}), \\ I_0^{(i)} &= Net_D(g_i, s_i^E; \theta_D) + I_0^{L\uparrow}, \text{ Decoder.} \end{aligned} \tag{3.12}$$

Donde $I_0^{(i)}$ es la i-ésima salida, s_i^E es el mapa de características de Net_E e $I_0^{L\uparrow}$ es la interpolación bi-cúbica utilizada para el *upsample* de I_0^L.

En la figura 3.11 se puede ver la arquitectura de la red.

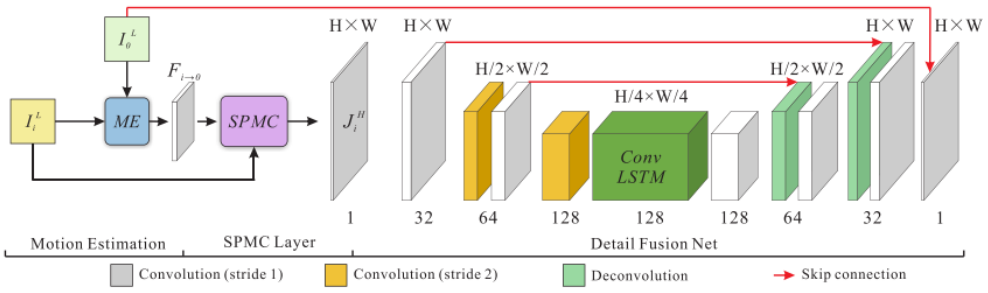


Figura 3.11 Arquitectura de la red para el paso i. Figura tomada de [42].

Entrenamiento

Separan el entrenamiento en tres etapas, en la primer etapa consideran únicamente Net_{ME}. Dado que el valor real del flujo óptico es desconocido, utilizan entrenamiento no supervisado, siendo el costo a minimizar:

$$L_{ME} = \sum_{i=-T}^{T} ||I_i^L - \tilde{I}_{0\to i}^L||_1 + \lambda_1 ||\nabla F_{i\to 0}||_1, \tag{3.13}$$

donde $\tilde{I}_{0\to i}^L$ es el la imagen desplazada de I_0^L de acuerdo al flujo estimado $F_{i\to 0}$.

En la segunda etapa fijan los pesos aprendidos θ_{ME} y entrenan Net_{DF}. Utilizan el error euclídeo entre la imagen de referencia estimada y la real:

$$L_{SR} = \sum_{i=-T}^{T} k_i ||I_0^H - I_0^{(i)}||_2^2. \tag{3.14}$$

Empíricamente fijan $k_{-T} = 0{,}5$, $k_T = 1{,}0$ e interpolan linealmente los valores intermedios.

En la tercera y última etapa consideran el sistema entero, utilizando la perdida total como

$$L = L_{SR} + \lambda_2 L_{ME}, \tag{3.15}$$

donde λ_2 es el peso de balanceo entre las dos clases.

3.3. Regularización STV

A continuación se presenta el método variacional Shannon Total Variation (STV), el cual se toma como método de referencia. Sea el problema lineal inverso, abordado con similitud

de datos cuadrática y regularización STV [1], se tiene

$$u_{HR} = \underset{u:\Omega\to R}{argmin}||Au-u_0||_2^2+\lambda STV_2(u) \tag{3.16}$$

donde u_0 es la imagen observada, A el operador lineal que modela la deformación ϕ_p seguida de un filtro k y de un submuestreo S y λ el parámetro de regularización que compensa el termino de similitud de datos $||Au-u_0||_2^2$ y el termino de regularización $\mathrm{STV}_2(u)$ en el proceso de minimización. STV_2 denota la variación total de Shannon.

El operador A realiza una interpolación de Shannon de la imagen HR desconocida u [1]. Relacionando esto con la ecuación 2.2, A concatena los operadores ϕ_p, k y S, donde la deformación geométrica ϕ_p se realiza mediante una interpolación de Shannon. Para el caso en que ϕ_p sea una traslación, como en el caso de las imágenes satelitales, $A_{i,j} = (k*U)\phi_p)$ donde U es la interpolación de shannon de u. Para el caso de una deformación más compleja, el núcleo k se deforma y hay que hacer aproximaciones para que sea calculable.

En el caso multi-imagen A es la pila de operadores A_p, uno por cada imagen, cada una con una traslación ϕ_p diferente y la imagen u_0 es la pila de imágenes de baja resolución correspondiente.

El tema de como se simula de manera eficaz el operador A y su adjunto es otro tema mas complejo que excede este trabajo.

La variación total de Shannon, STV_2, se define a continuación. Sea $\Omega = I_M \times I_N$ el dominio discreto de tamaño $M \times N$ y $u \in \mathbb{R}^\Omega$ una imagen discreta en escala de grises de dominio Ω, siendo $\Omega_2 = I_{2M} \times I_{2N}$. Sea U la interpolación de Shannon de u y $\nabla U : \mathbb{R}^2 \to \mathbb{R}^2$ el gradiente de U.

Se define la variación total de Shannon como

$$\mathrm{STV}_2(u) = \frac{1}{4}\sum_{(k,l)\in\Omega_2} ||\nabla_2 u(k,l)||_2, \tag{3.17}$$

donde $\nabla_2(k,l) = \nabla U(\frac{k}{2},\frac{l}{2})$ es el gradiente de la interpolación de Shanon de u en el punto $(k/2,l/2)$,

Como se demuestra en [1], el término $\mathrm{STV}_2(u)$ puede ser formulado en forma dual,

$$\mathrm{STV}_2(u) = \underset{p:\Omega_2\to\mathbb{R}^2}{\text{máx}} \langle \frac{1}{4}\nabla_2 u, p\rangle - \delta_{\beta *}(p) \tag{3.18}$$

donde

$$\delta_{\beta*}(p) = \begin{cases} 0 & \text{si } \max_{(k,l)\in\Omega_2} ||p(k,l)||_2 \leq 1 \\ +\infty & \text{en caso contrario} \end{cases}$$

Del mismo modo, se puede mostrar que el término de similitud de datos se puede escribir de forma dual,

$$||Au - u_0||_2^2 = \max_{q\in\mathbb{R}^\omega} \langle Au, q\rangle - ||\frac{q}{2} + u_0||_2^2 + ||u_0||_2^2, \tag{3.19}$$

donde $u_0 \in \mathbb{R}^\omega$ es la imagen observada, siendo ω un subconjunto finito de $\mathbb{Z}^2$, posiblemente $\omega = \Omega$ y $A : \mathbb{R}^\Omega \to \mathbb{R}^\omega$ el operador lineal que modela la deformación.

Reemplazando el término cuadrático en 3.16, eliminando la constante $||u_0||_2^2$ (no cambia el conjunto de minimizadores), y reemplazando el termino de regularización con la formulación 3.18 se obtiene

$$u_{HR} = \underset{u:\Omega\to R}{argmin} \max_{p:\Omega_2\to R, q\in R^\omega} \langle \frac{\lambda}{4}\nabla_2 u, p\rangle + \langle Au, q\rangle - \delta_{\beta*}(p) - ||\frac{q}{2} + u_0||_2^2. \tag{3.20}$$

Para resolver el problema 3.20 se utiliza la implementación *stvsuperres* de Abergel et al. la cuál se basa en el algoritmo Chambolle-Pock [5].

En la figura 3.12 se puede ver el resultado de aplicar *stvsuperres* para la imagen satelital Pdbouc (FTM= 0,3, SNR= 45) variando el valor de λ. Se utilizan hasta 250 iteraciones dado que la energía calculada luego de dicha cantidad aumenta en el orden de 10^{-4}.

(a) Imagen Original

(b) Detalles de la imagen original

(c) stvsuperres $\lambda = 1$ / PSNR=43.4 dB

(d) Detalles de stvsuperres $\lambda = 1$

(e) stvsuperres $\lambda = 7$ / PSNR=42.6 dB

(f) Detalles de stvsuperres $\lambda = 7$

Figura 3.12 **Súper Resolución utilizando stvsuperres para Pdbouc (FTM=0.3, SNR=45)** En la primera fila se ve la imagen original (25cm) a la izquierda y un zoom a la derecha donde se pueden ver los detalles. En la segunda fila se presenta el resultado de la reconstrucción utilizando $\lambda = 1$ y en la tercera fila se presenta el resultado para $\lambda = 7$.

3.4. Interfaz desarrollada

Se crea una interfaz gráfica [1] que permite cargar una imagen y mejorar la resolución de la misma (Ver Figura. 3.13).

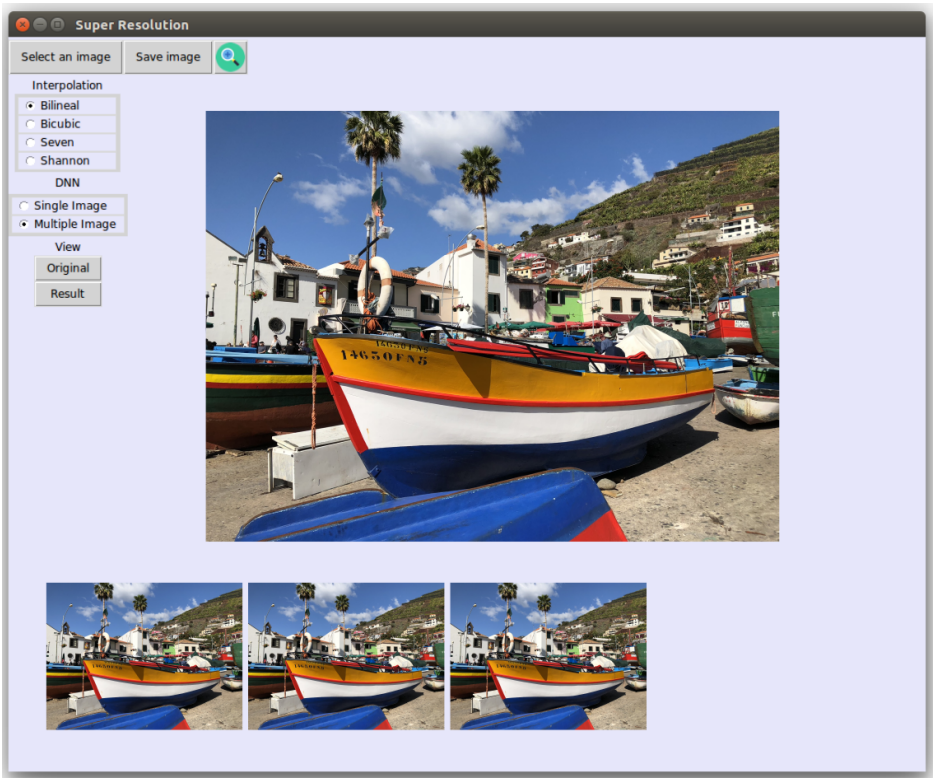


Figura 3.13 Interfaz gráfica para aumentar la resolución de una imagen

La interfaz permite cargar una o varias imágenes, seleccionar que tipo de red se va a utilizar, una red mono-imagen o una red multi-imagen y seleccionar que tipo de interpolación utilizar, es decir, que interpolación se utilizó para entrenar la red y que interpolación se va a utilizar para aumentar el tamaño de la imagen antes de entrar a la red.

La interfaz se desarrolló utilizando Python, Opencv y Tkinter.

[1] https://gitlab.fing.edu.uy/mmarzoa/sr-gui.git

Capítulo 4

Experimentos

En este capítulo se presentan los experimentos realizados para cada una de las redes estudiadas. En la primer sección se presenta la base de datos utilizada, en la segunda sección se presentan los experimentos realizados al utilizar una red neuronal profunda (DNN) para el problema de súper resolución (SR) mono-imagen y en la tercer sección se presentan los experimentos realizados con una DNN para el problema de SR multi-imagen. Por último, en la cuarta sección se realiza una comparación de todos los métodos estudiados.

4.1. Base de datos

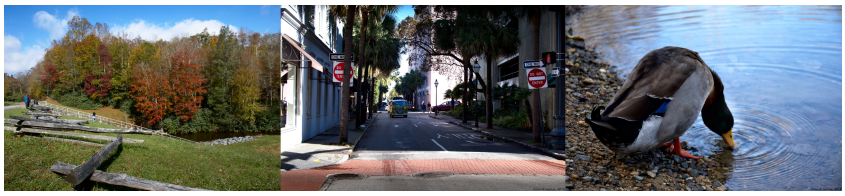
A continuación se presentan las tres bases de datos utilizadas en este trabajo.

- **DIV2K** es una base de datos desarrollada por Timofte et al. [46]. La base de datos consiste en 1000 imágenes de alta resolución. Son imágenes de alta calidad y con una cantidad insignificante de ruido y distorsión óptica. DIV2K presenta contenido variado, incluyendo imágenes de personas, flora, fauna, entornos naturales, ciudades y objetos (Ver figura 4.1a). Se selecciona esta base de datos debido a que es pública y cuanta con gran variedad de imágenes.

- **MatDB** es una base de datos compuesta por 14 imágenes de alta resolución, adquiridas por Matias Di Martino (Ver figura 4.1b). Se utiliza está base de datos para poder realizar experimentos con otra base de datos, diferente a DIV2K, y en particular con imágenes de alta resolución y con gran variedad de texturas.

- **SatDB** es una base de datos compuesta por imágenes satelitales cortesía de la agencia espacial francesa (CNES). La base se compone de tres imágenes, *Amiens* (urbana), *Balma* (semi-urbana) y *Pdbouc* (industrial) (Ver figura 4.1c), las cuales se tienen con

(a) Imágenes de la base de datos DIV2K [46].



(b) Imágenes de la base de datos MatDB.

(c) Imágenes de la base de datos SatDB. A la izquierda se encuentra la imagen satelital *Pdbouc* (industrial) , en el centro *Balma* (semi-urbana) y a la derecha *Amiens* (urbana)

Figura 4.1 **Muestras de imágenes de las base de datos.**

una resolución sobre el terreno de 25cm y se encuentran también con FTM (Modulation Transfer Function) de 0.05, 0.1, 0.2, 0.3. A su vez se cuenta con la simulación de las imágenes trasladadas y un archivo de texto con las traslaciones entre ellas. Las imágenes son simuladas con un ruido de luminiscencia SNR de 8, 30 y 45.

4.2. Red Mono-Imagen

En esta sección se detalla el entrenamiento realizado, se presenta el estudio sobre la elección de la interpolación, luego como dicha elección afecta a los resultados de la red y por último los experimentos realizados por la red mono-imagen con imágenes satelitales.

Entrenamiento

En función de los recursos de computo disponibles, se definió para este trabajo utilizar conjuntos de 30, 10 y 20 imágenes de entrenamiento, validación y test respectivamente de la

base de datos DIV2K. Como la red esta entrenada con *patches* individuales y las imágenes de entrada son de alta resolución, incluso un número reducido de imágenes produce un gran volumen de datos diversos para el entrenamiento. Para test se utilizaron además de DIV2K dos bases de datos complementarias, MatDB y SatDB.

Se realizan 16 recursiones. Si se desdobla la capa de recursión, la cadena más larga desde la entrada a la salida pasa por 20 capas de convolución, con un campo receptivo de 41 por 41. Como se recomienda en el artículo, para la optimización numérica de la red se utilizo el algoritmo adam con valores: momentum=0.9 y weight decay=10^{-4}.

Se utilizan 256 filtros de tamaño 3×3 para todos los pesos de las capas. Las imágenes de entrenamiento se dividen en patches de 41 por 41 con paso 21 y se realizan mini-batch de 64 patches para el descenso estocástico del gradiente.

Para inicializar los pesos en las capas no recursivas, se utiliza el método descrito por He et al [17]. Para las capas recursivas, se ponen todos los pesos en cero excepto las conexiones a la misma neurona de la siguiente capa. Los *bias* inicializan en cero. El *learning rate* se establece inicialmente en 0.01 y luego se reduce por un factor de 10 si el error de validación no disminuye durante 5 *epochs*. Si la velocidad de aprendizaje es inferior a 10^{-6}, se termina el procedimiento.

Elección de la interpolación

Para testear el impacto de la elección del núcleo de reducción R y prolongación P (sección 3.1) se crean cuatro grupos de datos utilizando el mismo conjunto de imágenes HR pero diferentes métodos de interpolación. Por lo tanto, hay disponibles cuatro modelos de DNN para súper resolución: Bilineal, Bicúbico, Seven y Shannon. Específicamente, la referencia al modelo DNN refiere a los parámetros θ (Como se muestra en la figura 3.7) excluyendo explícitamente la elección de la interpolación P.
Para evaluar el impacto de la elección de la interpolación R y P, se prueban los cuatro modelos con diferentes estrategias de sub-muestreo durante el entrenamiento y el test. Los resultados de estos experimentos se resumen en la Tabla 4.1.

Como se puede ver en la tabla, los mejores resultados se encuentran cuando el núcleo utilizado para la interpolación de entrenamiento y de test son el mismo. Adicionalmente, se repiten los experimentos previos sobre el conjunto de datos DIV2K agregando ruido Gaussiano de media cero y desviación estándar 1.43, es decir, un SNR de 45. Estos resultados son los que se encuentran en la fila inferior en la tabla 4.1. Como se puede ver, incluso con la presencia de ruido, las estrategias de interpolación juegan un rol importante. Estos resultados muestran que el mejor PSNR se obtiene utilizando la interpolación Seven para entrenamiento

Base de Datos	Interpolación (Test)	Interpolación (Train)			
		Bilineal	Bicubic	Seven	Shannon
DIV2K	Blineal	**32.67/0.68**	31.40/0.67	31.60/0.67	29.18/0.59
	Bicubic	31.47/0.70	**32.73/0.70**	30.09/0.69	30.36/0.63
	Seven	31.92/0.68	30.75/0.67	**32.75/0.69**	29.05/0.59
	Shannon	26.80/0.62	26.96/0.61	26.92/0.62	**31.37/0.65**
MatDB	Blineal	**37.59/0.83**	35.70/0.82	37.89/0.83	35.05/0.78
	Bicubic	37.50/0.84	**38.21/0.84**	36.26/0.83	36.14/0.81
	Seven	37.19/0.82	35.44/0.81	**38.31/0.83**	34.89/0.78
	Shannon	34.79/0.79	35.14/0.79	34.36/0.78	**36.70/0.82**
SatDB	Blineal	**37.59/0.90**	34.26/0.87	37.30/0.89	33.78/0.83
	Bicubic	36.97/0.89	**38.21/0.91**	34.98/0.87	34.88/0.86
	Seven	36.70/0.89	33.93/0.87	**38.53/0.90**	33.68/0.83
	Shannon	35.01/0.87	35.93/0.87	34.22/0.86	**37.41/0.89**
Promedio	Blineal	**35.95/0.80**	33.79/0.79	35.60/0.80	32.67/0.73
	Bicubic	35.31/0.81	**36.38/0.82**	33.78/0.80	33.79/0.77
	Seven	35.27/0.80	33.37/0.78	**36.53/0.81**	32.54/0.73
	Shannon	32.20/0.76	32.68/0.76	31.83/0.75	**35.16/0.79**
DIV2K noise	Blineal	**32.83/0.62**	31.16/0.60	32.42/0.62	29.88/0.56
	Bicubic	32.31/0.64	**32.93/0.64**	31.68/0.64	31.01/0.60
	Seven	32.19/0.62	31.07/0.59	**32.82/0.63**	29.83/0.56
	Shannon	28.05/0.60	29.35/0.61	28.85/0.61	**31.48/0.63**

Cuadro 4.1 Promedio de PSNR y SSIM (PSNR/SSIM) en los conjuntos de test DIV2K, MatDB y SatDB. Para el entrenamiento se utiliza el mismo P y R (R-train = P-train). Para el test se utiliza R dado que la baja resolución de las imágenes es simulada, y al igual que para el entrenamiento, se utiliza el mismo P y R (R-test = P-test).
La penúltima fila de la tabla corresponde al promedio de las tres bases de datos, mientras que la última fila corresponde a los resultados obtenidos al agregar ruido Gaussiano (media cero y desviación estándar 1.43) a la base de entrenamiento DIV2K y a los conjuntos de test.

Base de Datos	P	Interpolación (Train)			
		Bilineal	Bicubic	Seven	Shannon
Sat Images	Blineal	33.60/0.82	35.44/0.84	33.58/0.82	**36.13/0.84**
	Bicubic	32.76/0.81	34.97/0.84	32.73/0.81	**35.72/0.84**
	Seven	33.82/0.83	35.53/0.84	32.86/0.82	**35.95/0.84**
	Shannon	33.53/0.83	35.39/0.84	33.59/0.82	**36.17/0.84**

Cuadro 4.2 Resultado para las imágenes satelitales cuando se asume desconocido el modelo de degradación. El resultado obtenido se muestra utilizando cada uno de los modelos entrenados (utilizando R-train = P-train) y cada tipo de kernel utilizando (P) para aumentar el tamaño de las imágenes de entrada.

y test. Además este núcleo presenta el decaimiento más lento de valores propios, como se puede ver en la figura 3.4. Esto puede indicar que el núcleo de suavizado definido por este método es el más simple de invertir, lo que explicaría los resultados observados.

En la figura 4.2 y 4.4 se pueden ver ejemplos del aumento de la resolución en las imágenes de experimentación utilizando los modelos basados en DNN descritos anteriormente.

Modelos de degradación desconocidos

En la práctica es común no saber el modelo de degradación $(R(I))$ de una configuración óptica en particular. Para evaluar el impacto de la opción de interpolación cuando el método de degradación es desconocido, se utilizan las imágenes provistas por el CNES. Para dichas imágenes se tiene la imagen de alta resolución y la imagen degradada, pero no se utilizara información del proceso de degradación. Se experimenta con las imágenes de baja resolución, mejorando la misma con los cuatro modelos de la red neuronal entrenados con dicho fin.

En la tabla 4.2 se puede ver el PSNR y SSIM obtenido para el conjunto de las imágenes satelitales.

Como interpolación inicial se prueban los cuatro enfoques (bilineal, bicubica, Seven y Shannon). Como se puede ver en la tabla para este caso de estudio, el modelo entrenado generando los datos con la interpolación de Shannon presenta los mejores resultados, sin importar que núcleo de interpolación P se utilice.

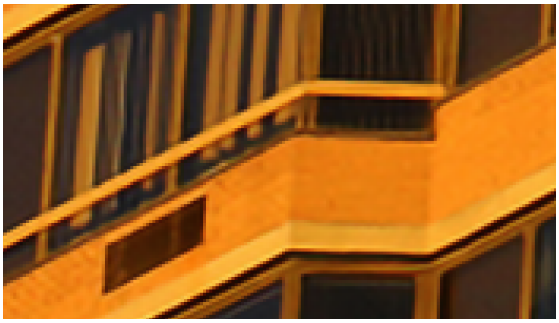
Resultados Visuales

En la figura 4.5 se muestran los resultados visuales de aumentar la resolución de una imagen utilizando distintos modelos de entrenamiento de la red.

(a) Imagen Original

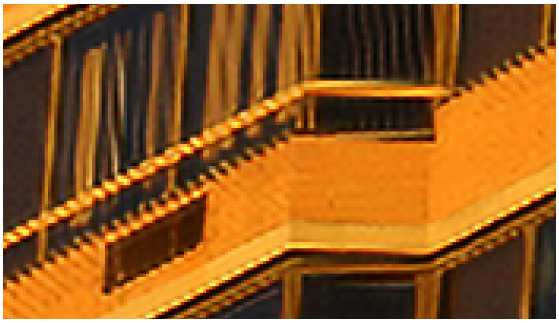
(b) Detalles de la imagen original

(c) Imagen LR / P = R = Bicubica



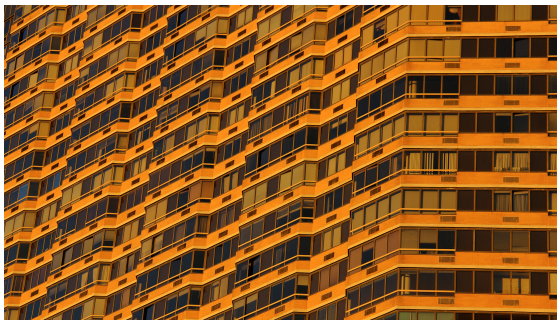
(d) In: Img (c) - Train: Bicubica - PSNR:31.71

(e) Imagen LR / P = R = Shannon



(f) In: Img (e) - Train: Bicubica - PSNR:24.96

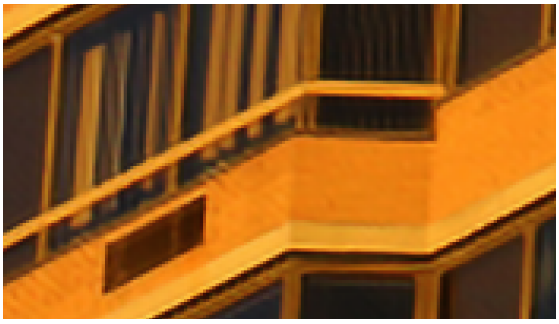
Figura 4.2 Resultado de aplicar la red neuronal para súper resolución a una imagen cuando se utilizan diferentes métodos de interpolación. La columna de la izquierda corresponde a la entrada de baja resolución y la de la derecha a el resultado de la red al utilizar el modelo entrando con la interpolación indicada, y generando la imagen de test con el P y R indicados.



(a) Imagen Original

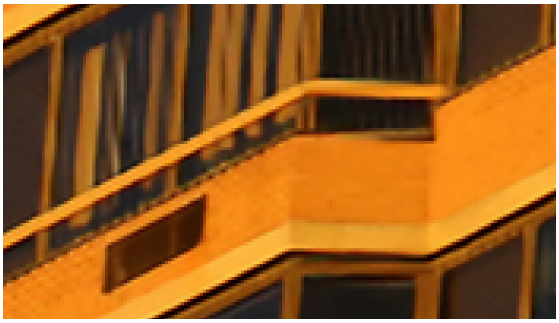
(b) Detalles de la imagen original

(c) Imagen LR / P = Bicubica R = Shannon



(d) In: Img (g) - Train: Bicubica - PSNR:24.96

(e) Imagen LR / P = R = Shannon



(f) In: Img (i) - Train: Shannon - PSNR:29.99

Figura 4.3 Resultado de aplicar la red neuronal para súper resolución a una imagen cuando se utilizan diferentes métodos de interpolación. La columna de la izquierda corresponde a la entrada de baja resolución y la de la derecha a el resultado de la red al utilizar el modelo entrando con la interpolación indicada, y generando la imagen de test con el P y R indicados.

(a) Imagen Original

(b) Detalles de la imagen original

(c) Detalles de la imagen LR (Bilineal)

(d) SR de (c) / PSNR = 28.55

(e) Detalles de la imagen LR (Bicubica)

(f) SR de (e) / PSNR = 28.76

(g) Detalles de la imagen LR (Seven)

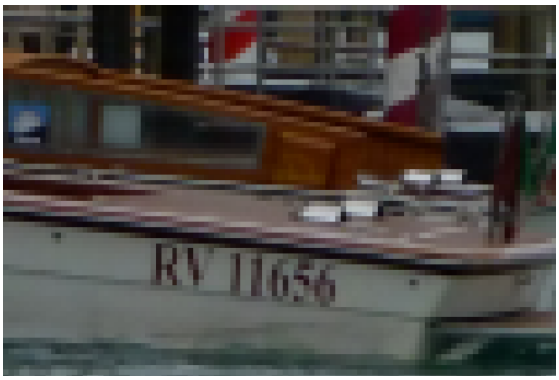
(h) SR de (g) / PSNR = 28.37

(i) Detalles de la imagen LR (Shannon)

(j) SR de (i) / PSNR = 27.90

Figura 4.4 Resultados de la red mono-imagen utilizando la misma interpolación para test y train

(a) Imagen Original



(b) Detalles de la imagen original

(c) Detalles de la imagen SR (Bilineal)

(d) Detalles de la imagen SR (Bicubica)

(e) Detalles de la imagen SR (Seven)

(f) Detalles de la imagen SR (Shannon)

Figura 4.5 Resultados visuales la red mono-imagen

4.3. Red Multi-Imagen

En esta sección se presentan los experimentos realizados con la red neuronal de súper resolución multi-imagen presentada en el capítulo anterior. Se comparan los resultados de la red con los obtenidos utilizando el algoritmo STV.

Entrenamiento

Los autores de la red ponen a disposición el modelo entrenado y no el código para realizar el entrenamiento. De todas formas detallan los parámetros del entrenamiento realizado para dicho modelo. Entrenan la red con imágenes de alta resolución y sin ruido. Adquieren 975 secuencias de imágenes de vídeo alta resolución de 1080p, compuestos por escenas naturales y urbanas, de 31 frames cada uno. El video se submuestrea utilizando la interpolación bicubica y toman aleatoriamente 945 de ellas como datos de entrenamiento, y las otras 30 secuencias para validación y prueba.

Para el entrenamiento del modelo utilizan *Adam solver* con un learning rate de 0,0001, $\beta_1 = 0{,}9$ y $\beta_2 = 0{,}999$. Primero entrenan el módulo de estimación de movimiento usando solo la pérdida $\mathscr{L}_{ME}$ de la ec. 3.13 con $\lambda_1 = 0{,}01$. Después de aproximadamente 70,000 iteraciones, fijan los parámetros Θ_{ME} y entrenan el sistema utilizando solo la pérdida $\mathscr{L}_{SR}$ de la ecuación 3.13 por 20,000 iteraciones. Finalmente, todos los parámetros se entrenan usando la pérdida total $\mathscr{L}$ de la ecuación 3.15, λ_2 se elige empíricamente como 0,01.

Prueba 1: Número óptimo de imágenes

Se busca determinar la cantidad óptima de imágenes vecinas a tener en cuenta por la red neuronal para realizar súper resolución multi-imagen. Los autores obtienen los mejores resultados con solo tres imágenes, lo cual es consistente con el resultado que se obtiene en las imágenes proporcionadas por el CNES (Ver tabla 4.3). Este resultado puede explicarse por el hecho de que la red ha sido entrenada para estimar el movimiento en secuencias de vídeo, en el que las brechas entre cuadros son espacialmente más irregulares que en este caso (una sola traslación entre dos imágenes dadas) y temporalmente más regulares que en este caso.

En los siguientes experimentos se restringe el uso a 3 imágenes tanto para la red neuronal como para el algoritmo STV, para realizar la comparación bajo las mismas condiciones.

Imagen	PSNR
Balma (3F)	**36.6333**
Balma (5F)	31.6038
Pdbouc (3F)	**31.5642**
Pdbouc (5F)	30.7472
Amiens (3F)	**35.2890**
Amiens (5F)	32.1663

Cuadro 4.3 Desempeño de la red con respecto a el número de cuadros considerado. (.F) representa el número de cuadros utilizados.

Imagen (Método)	PSNR / SSIM
Balma (NN)	19.6012 / 0.4470
Balma (STV)	**29.0050 / 0.8601**
Pdbouc (NN)	18.8397 / 0.4095
Pdbouc (STV)	**25.2679 / 0.7732**
Amiens (NN)	21.3842 / 0.4720
Amiens (STV)	**36.6841 / 0.9139**

Cuadro 4.4 Comparación del rendimiento del algoritmo STV y la red neuronal (NN) cuando se utilizan imágenes con ruido

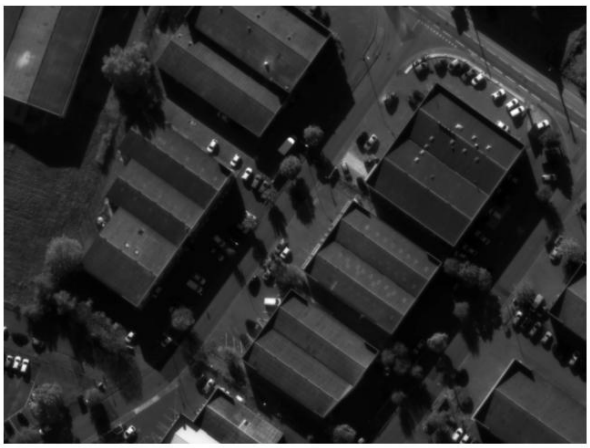
Prueba 2: Imágenes con ruido

Se busca determinar el comportamiento de la red cuando se utilizan imágenes con ruido aleatorio gaussiano (sigma = 8). Para ello primero se agrega ruido aleatorio a todas las imágenes del stack y luego se prueba la red y el algoritmo STV con dicho stack (Ver tabla 4.4).

En la figura 4.6 se observa la imagen luego de mejorar la resolución utilizando la red neuronal y la imagen resultado del algoritmo STV. Se puede observar que la red no elimina el ruido de la imagen, mientras que el algoritmo STV si. Los resultados tienen sentido dado que la red no se encuentra entrenada ni diseñada para imágenes con ruido. En los siguientes experimentos se utilizan entonces las imágenes con el ruido más bajo (SNR = 45dB).

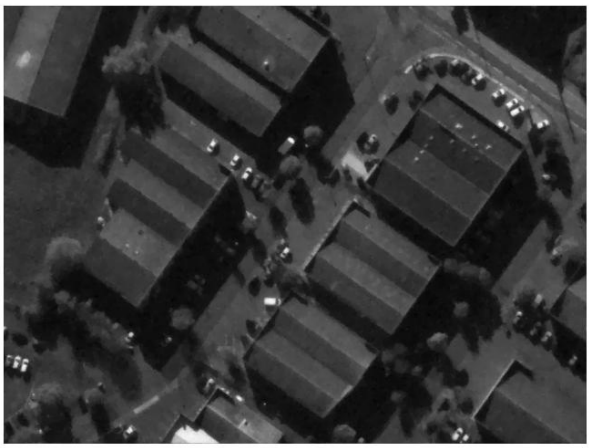
Prueba 3: Ruido en las traslaciones

El algoritmo STV supone se conocen las traslaciones, no así la red neuronal que ignora dicha información, calculándola dentro de la misma red. En este experimento se determina el rendimiento de la red neuronal en relación al método de referencia STV cuando las traslaciones entre las imágenes tienen ruido. El stack de imágenes con errores en el desplazamiento se crean como sigue: Sea e la magnitud de error en la traslación, agregamos al desplazamiento



(a) Imagen Original

(b) Resultado NN



(c) Resultado STV

Figura 4.6 Imágenes resultado de utilizar como entrada a los algoritmos imágenes con ruido (Prueba 2).

Método	Ruido e	PSNR
NN	-	**33.2**
STV	0	40.5
STV	0,1	35.6 ± 7
STV	0,3	28.8 ± 2

Cuadro 4.5 Resultado de la reconstrucción de las imágenes con el método NN y STV con error en las traslaciones de tamaño e, para una pila de 3 imágenes de Amiens SNR@L1 = 45 dB

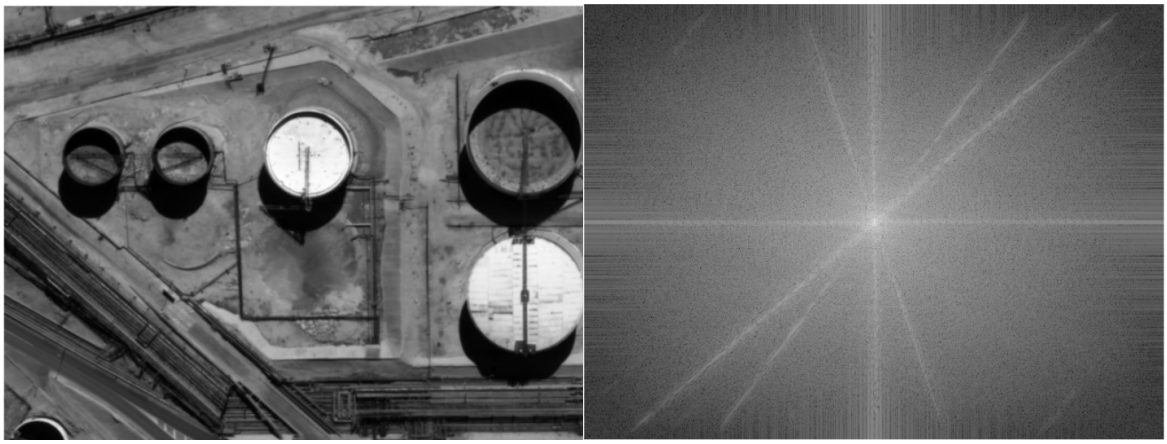
δ^j un error ε^j con una distribución uniforme en un círculo de radio e, es decir,

$$\varepsilon^j = (e\cos\varphi^j, e\sin\varphi^j),\ \varphi^j \sim \mathbb{U}[0,2\pi]. \tag{4.1}$$

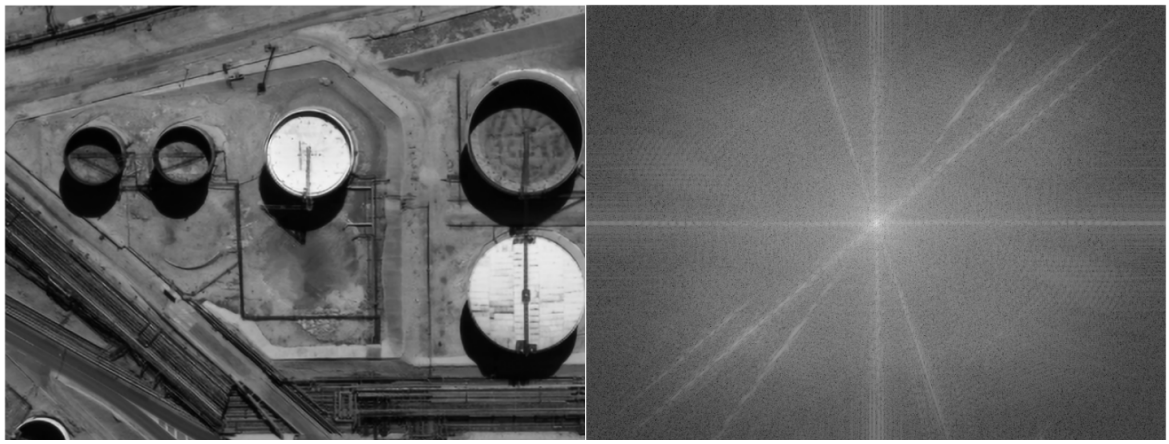
Se realiza la operación varias veces para obtener estadísticas del rendimiento del método STV para varios sorteos del error. Observando los resultados de la tabla 4.5 se ve que al agregarle ruido a las traslaciones mayor a 0.3 píxeles el resultado de la red es mejor para las tres imágenes que el resultado obtenido al utilizar el algoritmo STV.

En la figuras 4.7 y 4.8 se muestra el resultado de utilizar los dos métodos (STV y NN) para la imagen Amiens con los diferentes niveles de ruido en las traslaciones.

Los experimentos anteriores indican que la red utilizada funciona mejor cuando se utilizan solo tres imágenes de baja resolución. A su vez estos resultados sugieren que la red no admite imágenes de baja resolución en la entrada y que aunque el algoritmo STV presenta mejores resultados cuando las traslaciones se conocen de forma precisa, si el error en las traslaciones es suficiente grande (0.3 pixeles), los resultados obtenidos son mejores utilizando la red. Otro factor a destacar es que el tiempo de ejecución de la red es mucho más chico que el del algoritmo STV (3 segundos contra aproximadamente 245).

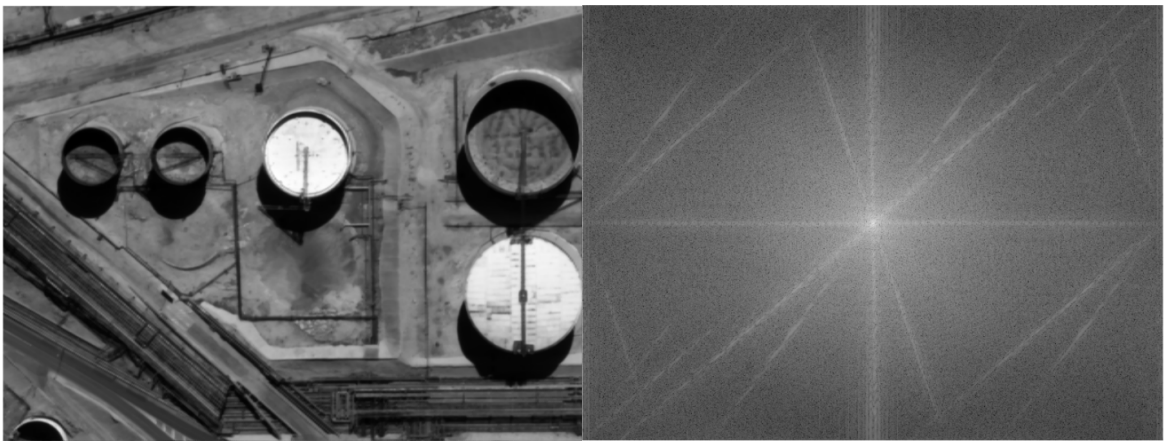


(a) Imagen Original (b) FFT de (a)

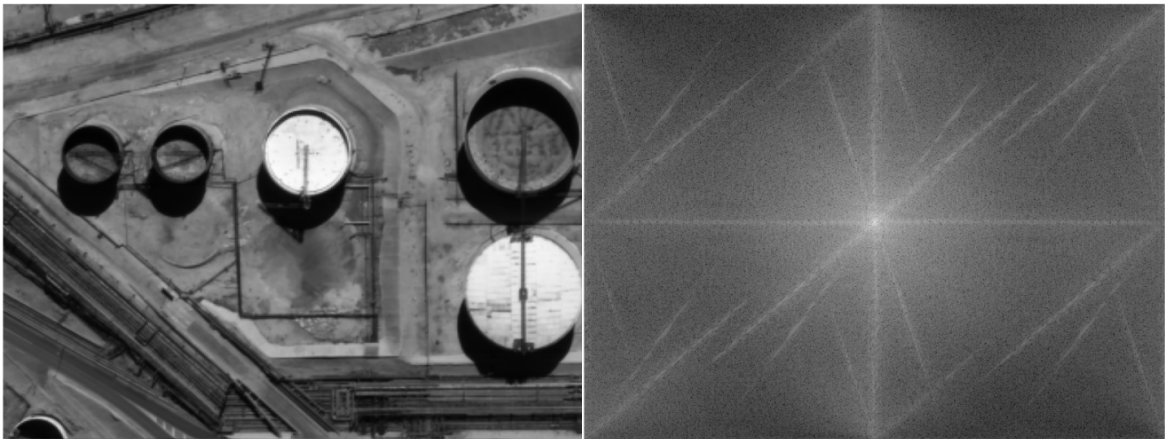


(c) Imagen reconstruida (NN)- PSNR 33.1 dB (d) FFT de (c)

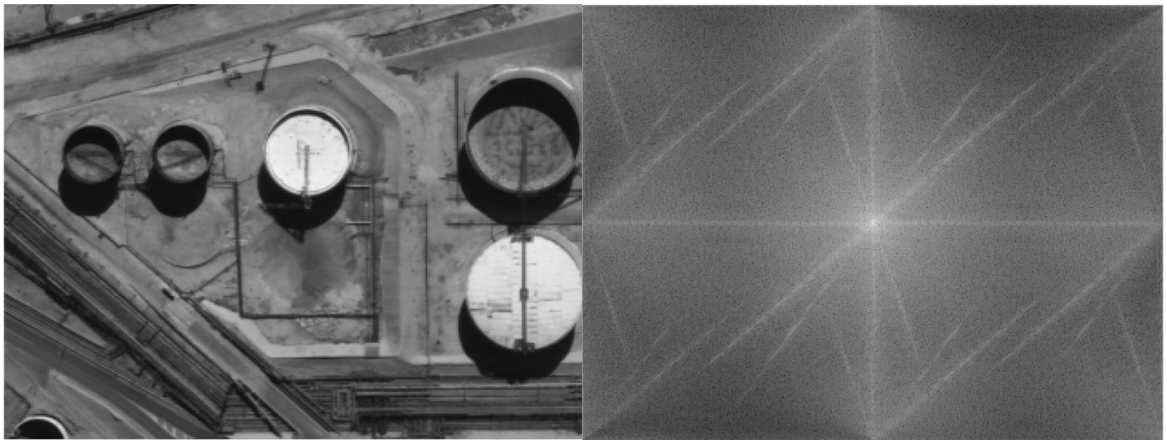
Figura 4.7 Súper resolución para el método NN a partir de tres imágenes de baja resolución de la pila Amiens con SNR = 45dB. Se muestra la imagen y su FFT.



(a) STV, e = 0, PSNR 40,5 dB (b) FFT de (a)



(c) STV, e = 0,1 - PSNR 35,6 dB (d) FFT de (c)



(e) STV, e = 0,3 - PSNR 28,8 dB (f) FFT de (e)

Figura 4.8 Súper-resolución para el método STV a partir de tres imágenes de baja resolución de la pila Amiens con SNR = 45dB

4.4. Comparación de las redes

En esta sección se estudia como se comporta la red mono-imagen en comparación con la red multi-imagen. Primero se presentan las características utilizadas para cada red y luego se presentan los resultados.

Características de las redes

Dado que la red multi-imagen [42] se encuentra entrenada utilizando la interpolación bicubica y dado que las redes presentan un mejor rendimiento cuando el núcleo utilizado para entrenar la red (R y P) es el mismo que el utilizado para aumentar la escala (P) de la imagen a la que se busca mejorar la resolución, se utilizará la interpolación bicubica para aumentar la resolución de la imagen de entrada y se compararán las redes entrenadas con dicha interpolación.
Se realizan experimentos creando pilas de tres imágenes con desplazamiento sub-pixel entre si y otros utilizando las pilas de imágenes satélitales brindadas por el CNES.

Resultados

Se evalúa el funcionamiento de las dos redes, la red multi-imagen (MI) y la red mono-imagen (SI) para la imagen Funchal (Figura 4.9).

Figura 4.9 Imagen Funchal

La red multi-imagen admite como entrada únicamente imágenes de tamaño 240×320, por lo que se utiliza una sección de la imagen. Los desplazamientos entre imágenes se crean de forma aleatoria, creando varias pilas para obtener la estadística del rendimiento. En la

tabla 4.6 se observan los resultados numéricos, y en la figura 4.10 se muestran las imágenes resultado así como los detalles.

Red	PSNR/SSIM
Mono-Imagen	27.37 / 0.94
Multi-Imagen	$30{,}67 \pm 0{,}27$ / $0{,}96 \pm 0{,}0$

Cuadro 4.6 Resultados de las redes para la imagen Funchal. Para la red multi-imagen la prueba se hace con pilas de 3 imágenes con translaciones aleatorias.

Otro ejemplo se puede ver con la imagen Pato, la cual se aumenta la resolución también mediante las dos redes.

Los desplazamientos entre imágenes se crean de forma aleatoria, creando varias pilas para obtener la estadística del rendimiento. En la tabla 4.7 se observan los resultados numéricos, y en la figura 4.11 se muestran las imágenes resultado así como los detalles de la misma.

Red	PSNR / SSIM
Mono-Imagen	33.75 / 0.97
Multi-Imagen	$40{,}55 \pm 0{,}53$ / $0{,}95 \pm 0{,}0$

Cuadro 4.7 Resultados de las redes para la imagen Pato. Para la red multi-imagen la prueba se hace con pilas de 3 imágenes con translaciones aleatorias.

Se evalúan los stacks de imágenes satelitales brindadas por el CNES, para tres imágenes en cada pila. En este caso además de presentar los resultados utilizando igual interpolación para entrenar la red y para aumentar el tamaño de la imagen de test para la red mono-imagen, se muestran los resultados cuando la red se entrenó utilizando otra interpolación, y cuando se utiliza otro núcleo de interpolación, diferente a la bicubica, para aumentar el tamaño de imagen de entrada. En la tabla 4.8 se muestran los resultados numéricos de los experimentos y en las figuras 4.12 y 4.13 se muestran ejemplos de las imágenes resultado.

Red	SI				MI
Train (R y P)	Bicubica	Bicubica	Shannon	Shannon	Bicubica
Test (P)	Bicubica	Shannon	Bicubica	Shannon	Bicubica
Amiens	33.08	34.05	34.43	35.25	**36.54**
Balma	37.87	**40.9**	39.66	42.14	39.37
Pdbouc	33.51	37.2	33.37	**38.57**	32.83
Promedio	$34{,}82 \pm 2{,}65$	$37{,}38 \pm 3{,}42$	$35{,}82 \pm 3{,}36$	$38{,}65 \pm 3{,}44$	$36{,}24 \pm 3{,}28$

Cuadro 4.8 Resultados para las tres imágenes satelitales dadas por el CNES (FTM=0.3, SNR=45). Se muestran los resultados para la red multi-imagen (MI) y para la red mono-image (SI). Se presentan los resultados para ambas redes utilizando los modelos entrenados con la interpolación bicubica y aumentando la dimensión utilizando la misma interpolación, así como también se presentan los resultados para la interpolación de Shannon para la red mono-imagen.

(a) Referencia

(b) Detalles de (a)

(c) LR

(d) Detalles de (c)

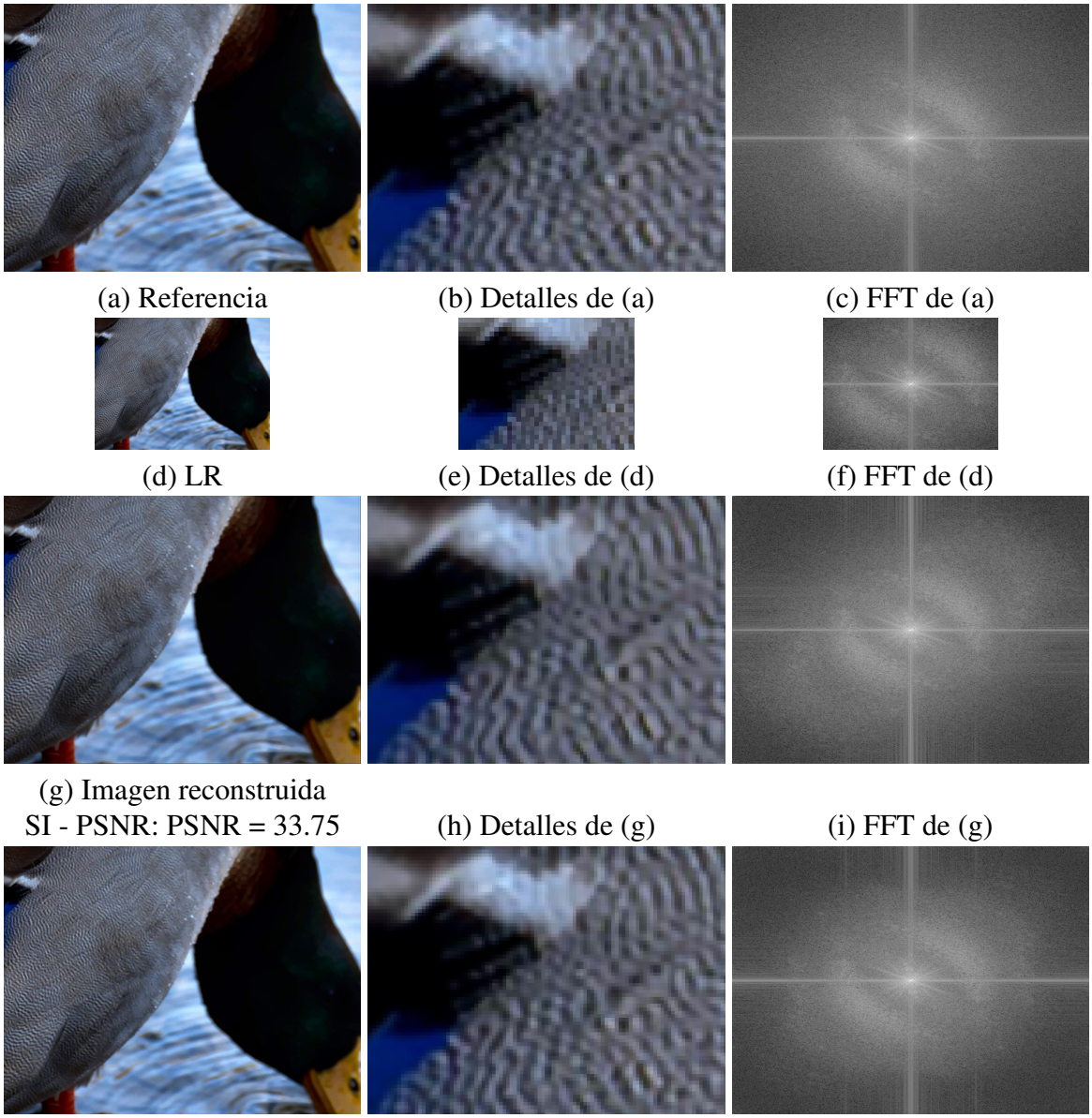
(e) Imagen reconstruida
SI - PSNR: PSNR = 27.36

(f) Detalles de (e)

(g) Imagen reconstruida
MI - PSNR: PSNR = 30.79

(h) Detalles de (g)

Figura 4.10 Resultados de aumentar la resolución de una sección de la imagen Funchal, en particular de una pila de tres imágenes con desplazamiento sub-pixel, utilizando la red multi-imagen (MI) y la red mono-imagen (SI), para esta última se utiliza una única imagen.



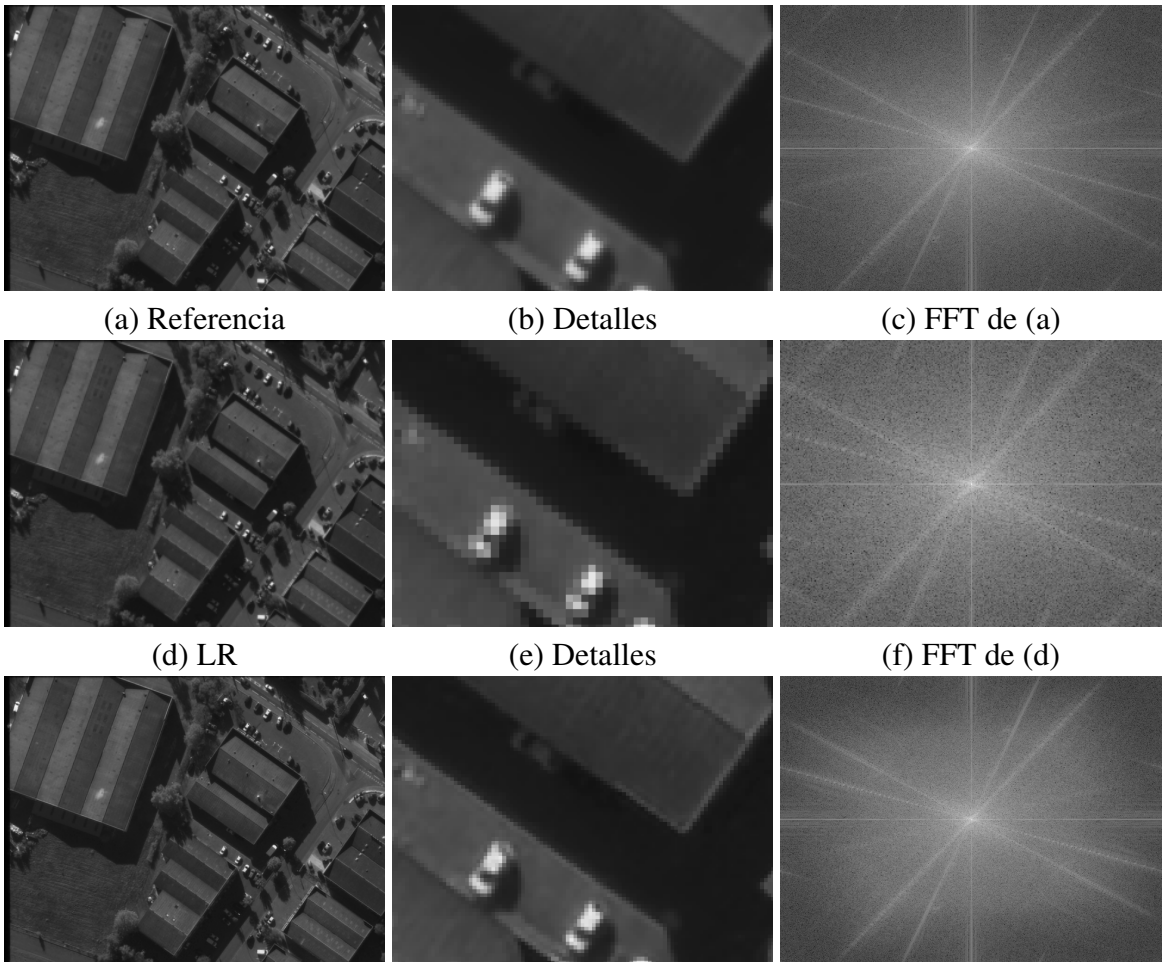
(a) Referencia (b) Detalles de (a) (c) FFT de (a)

(d) LR (e) Detalles de (d) (f) FFT de (d)

(g) Imagen reconstruida
SI - PSNR: PSNR = 33.75 (h) Detalles de (g) (i) FFT de (g)

(j) Imagen reconstruida
MI - PSNR: PSNR = 40.86 (k) Detalles de (j) (l) FFT de (j)

Figura 4.11 Resultados de aumentar la resolución de la imagen Pato, en particular de una pila de tres imágenes con desplazamiento sub-pixel, utilizando la red multi-imagen y la red mono-imagen, para esta última se utiliza una única imagen.

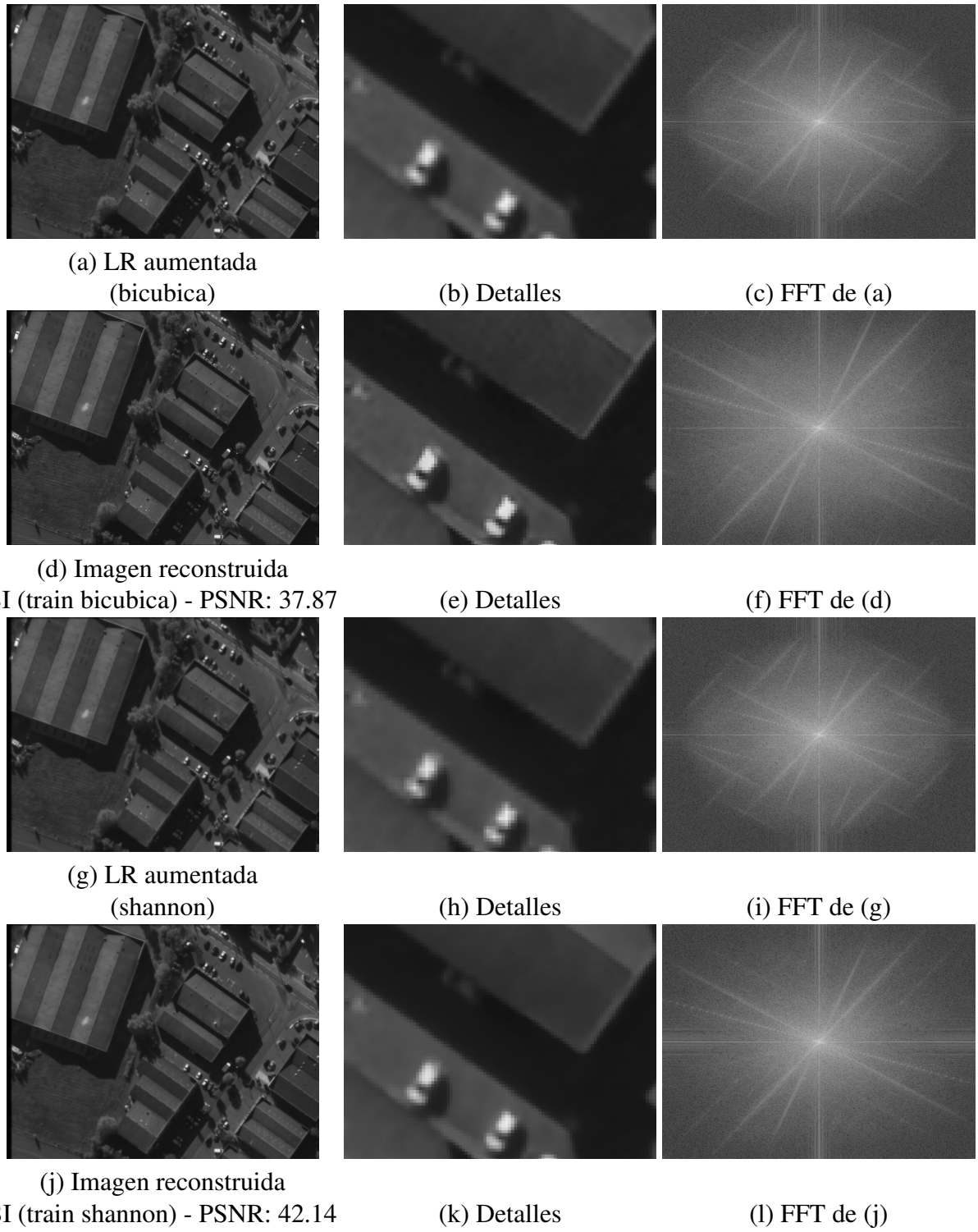


(a) Referencia (b) Detalles (c) FFT de (a)

(d) LR (e) Detalles (f) FFT de (d)

(g) Imagen reconstruida
MI - PSNR: 39.37 (h) Detalles (i) FFT de (g)

Figura 4.12 Resultado de aplicar a la pila balma de tres imágenes la red multi-imagen (MI) (FTM=0.3, SNR=45).



(a) LR aumentada (bicubica) (b) Detalles (c) FFT de (a)

(d) Imagen reconstruida SI (train bicubica) - PSNR: 37.87 (e) Detalles (f) FFT de (d)

(g) LR aumentada (shannon) (h) Detalles (i) FFT de (g)

(j) Imagen reconstruida SI (train shannon) - PSNR: 42.14 (k) Detalles (l) FFT de (j)

Figura 4.13 Resultado de aplicar a la primera imagen de la pila balma de tres imágenes la red mono-imagen (FTM=0.3, SNR=45), utilizando diferentes núcleos para aumentar el tamaño de la imagen antes de ingresar a la red (P) y diferentes modelos de red entrenada (bicubica y shannon).

Capítulo 5

Discusión y conclusiones

A continuación se presentan las conclusiones que se desprenden de este trabajo y las principales líneas de investigación para seguir a partir de lo aquí realizado.

En este trabajo se estudia el comportamiento de las redes neuronales profundas para el problema de super-resolución de imágenes. Se presenta el estado del arte para dichos algoritmos, utilizando tanto los que utilizan una única imagen de entrada como los que utilizan varias imágenes de entrada para generar una imagen de alta resolución.

Se comparan los métodos basados en redes neuronales profundas con un método variacional propuesto recientemente, STV. En particular, se estudia el impacto que tiene la incertidumbre en la estimación de la traslación de las imágenes de baja resolución para las diferentes estrategias de super-resolución. Se determina que cuando se cuenta con las traslaciones entre las imágenes con alta precisión, los algoritmos convencionales presentan mejor rendimiento que los algoritmos basados en redes neuronales. En caso contrario, que el desplazamiento tenga un error mayor, los algoritmos basados en redes neuronales profundas presentan un mejor desempeño.

Para entrenar los métodos basados en redes neuronales es necesario generar muestras sintéticas a partir de imágenes de alta resolución. Se evalúa como la elección de los métodos de submuestreo y sobremuestreo afecta el aprendizaje de las redes. Ademas, se cuantifica el impacto de entrenar y testear una red utilizando datos generados con diferentes modelos de degradación.

Se experimenta con datos satelitales reales, asumiendo desconocido el modelo de distorsión. Se determina en este caso, que los mejores resultados se dan cuando se entrena la red utilizando la interpolación de Shannon para generar la baja resolución de las imágenes y para aumentar el tamaño de la imagen al comienzo de la red.

En el caso de que las imágenes tengan ruido, es importante contar con una red entrenada especialmente con dicho fin, o contar con un filtro previo al ingreso de la red.

En cuanto al costo computacional, el tiempo de ejecución de las redes es significativamente inferior que para el algoritmo STV (3 segundos contra aproximadamente 245).

Por ultimo, se provee como resultado adicional de esta tesis, una aplicación de código abierto para aumentar la resolución de una imagen, permitiendo seleccionar la red que se quiere utilizar y el método de interpolación utilizado en el entrenamiento de la red. La interfaz puede ser utilizada para reproducir los resultados aquí presentados.

A continuación se presentan dos trabajos que quedaron pendientes.

En este trabajo se utilizaron diferentes algoritmos para realizar la super-resolución. El algoritmo variacional STV utiliza como entrada los desplazamientos de las imágenes, mientras la red neuronal multi-imagen calcula el desplazamiento de las imágenes de baja resolución. Resulta de interés investigar si al aumentar la resolución de una imagen, utilizando una red de super-resolución mono-imagen, eliminando de esta manera el aliasing, permite realizar una estimación más precisa del movimiento entre las imágenes y mejorar el resultado obtenido al realizar el registrado posterior.

Resulta también de gran interés poder contar con una red neuronal profunda para realizar súper-resolución multi-imagen, dado que las existentes no tienen el código disponible en su totalidad.

Glosario

CNES	centro nacional de estudios espaciales de Francia.
CNN	red neuronal convolucional.
HR	alta resolución.
LR	baja resolución.
MSE	error cuadrático medio.
PSF	función de dispersión del sistema.
PSNR	peak signal to noise ratio.
ResNet	residual network [18].
RNN	red neuronal recurrente.
SR	super-resolución.
SSD	mínimo de la suma de la diferencia al cuadrado.
SSIM	structural similarity index measure.

Bibliografía

[1] Abergel, R. and Moisan, L. (2017). The Shannon Total Variation. *Journal of Mathematical Imaging and Vision*, 59(2):341–370.

[2] Barjatya, A. (2004). Block matching algorithms for motion estimation. *IEEE Transactions Evolution Computation*, 8(3):225–239.

[3] Brown, L. G. (1992). A survey of image registration techniques. *ACM computing surveys (CSUR)*, 24(4):325–376.

[4] Caballero, J., Ledig, C., Aitken, A., Acosta, A., Totz, J., Wang, Z., and Shi, W. (2017). Real-time video super-resolution with spatio-temporal networks and motion compensation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[5] Chambolle, A. and Pock, T. (2011). A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of mathematical imaging and vision*, 40(1):120–145.

[6] Chang, G., Pan, T., Qiao, F., Clark Jr, J. W., and Mawlawi, O. R. (2009). Comparison between two super-resolution implementations in pet imaging. *Medical physics*, 36(4):1370–1383.

[7] Demirel, H. and Anbarjafari, G. (2011). Discrete wavelet transform-based satellite image resolution enhancement. *IEEE transactions on geoscience and remote sensing*, 49(6):1997–2004.

[8] Di Martino, M. and Facciolo, G. (2018). An analysis and implementation of multigrid poisson solvers with verified linear complexity. *Image Processing On Line*, 8:192–218.

[9] Dong, C., Loy, C. C., He, K., and Tang, X. (2016). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307.

[10] Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., and Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766.

[11] Drulea, M. and Nedevschi, S. (2011). Total variation regularization of local-global optical flow. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 318–323. IEEE.

[12] Freeman, W. T., Jones, T. R., and Pasztor, E. C. (2002). Example-based super-resolution. *IEEE Computer graphics and Applications*, 22(2):56–65.

[13] Glasner, D., Bagon, S., and Irani, M. (2009). Super-resolution from a single image. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 349–356. IEEE.

[14] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

[15] Goshtasby, A. A. (2005). *2-D and 3-D image registration: for medical, remote sensing, and industrial applications*. John Wiley & Sons.

[16] Greenspan, H. (2008). Super-resolution in medical imaging. *The Computer Journal*, 52(1):43–63.

[17] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

[18] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[19] Hemker, P. (1990). On the order of prolongations and restrictions in multigrid procedures. *Journal of Computational and Applied Mathematics*, 32(3):423–429.

[20] Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. *Artificial intelligence*, 17(1-3):185–203.

[21] Huang, J.-B., Singh, A., and Ahuja, N. (2015a). Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206.

[22] Huang, T. and Yang, J. (2010). Image super-resolution: Historical overview and future challenges. In *Super-resolution imaging*, pages 19–52. CRC Press.

[23] Huang, Y., Wang, W., and Wang, L. (2015b). Bidirectional recurrent convolutional networks for multi-frame super-resolution. In *Advances in Neural Information Processing Systems*, pages 235–243.

[24] Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2462–2470.

[25] Jo, Y., Wug Oh, S., Kang, J., and Joo Kim, S. (2018). Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3224–3232.

[26] Kappeler, A., Yoo, S., Dai, Q., and Katsaggelos, A. K. (2016). Video super-resolution with convolutional neural networks. *IEEE Transactions on Computational Imaging*, 2(2):109–122.

[27] Kennedy, J. A., Israel, O., Frenkel, A., Bar-Shalom, R., and Azhari, H. (2006). Super-resolution in pet imaging. *IEEE transactions on medical imaging*, 25(2):137–147.

[28] Kim, J., Kwon Lee, J., and Mu Lee, K. (2016a). Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654.

[29] Kim, J., Kwon Lee, J., and Mu Lee, K. (2016b). Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1637–1645.

[30] Lappe, M. and Rauschecker, J. P. (1993). A neural network for the processing of optic flow from ego-motion in man and higher mammals. *Neural Computation*, 5(3):374–391.

[31] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690.

[32] Li, F.-F., Johnson, J., and Yeung, S. (2018). Cs231n: Convolutional neural networks for visual recognition. *Stanford University CS231n: Convolutional Neural Networks for Visual Recognition*.

[33] Lin, F. C., Fookes, C. B., Chandran, V., and Sridharan, S. (2005). Investigation into optical flow super-resolution for surveillance applications.

[34] Lucas, B. D., Kanade, T., et al. (1981). An iterative image registration technique with an application to stereo vision.

[35] Mitchell, T. (1997). Machine learning, mcgraw-hill higher education. *New York*.

[36] Montufar, G. F., Pascanu, R., Cho, K., and Bengio, Y. (2014). On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932.

[37] Peled, S. and Yeshurun, Y. (2001). Superresolution in mri: application to human white matter fiber tract visualization by diffusion tensor imaging. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 45(1):29–35.

[38] Romano, Y., Isidoro, J., and Milanfar, P. (2017). Raisr: rapid and accurate image super resolution. *IEEE Transactions on Computational Imaging*, 3(1):110–125.

[39] Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

[40] Sood, R., Topiwala, B., Choutagunta, K., Sood, R., and Rusu, M. (2018). An application of generative adversarial networks for super resolution medical imaging. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 326–331. IEEE.

[41] Szeliski, R. (2010). *Computer vision: algorithms and applications*. Springer Science & Business Media.

[42] Tao, X., Gao, H., Liao, R., Wang, J., and Jia, J. (2017). Detail-revealing deep video super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy*, pages 22–29.

[43] Tatem, A. J., Lewis, H. G., Atkinson, P. M., and Nixon, M. S. (2001). Super-resolution target identification from remotely sensed images using a hopfield neural network. *IEEE Transactions on Geoscience and Remote Sensing*, 39(4):781–796.

[44] Tatem, A. J., Lewis, H. G., Atkinson, P. M., and Nixon, M. S. (2002). Super-resolution land cover pattern prediction using a hopfield neural network. *Remote Sensing of Environment*, 79(1):1–14.

[45] Thornton, M. W., Atkinson, P. M., and Holland, D. (2006). Sub-pixel mapping of rural land cover objects from fine spatial resolution satellite sensor imagery using super-resolution pixel-swapping. *International Journal of Remote Sensing*, 27(3):473–491.

[46] Timofte, R., Agustsson, E., Van Gool, L., Yang, M.-H., Zhang, L., Lim, B., Son, S., Kim, H., Nah, S., Lee, K. M., et al. (2017). Ntire 2017 challenge on single image super-resolution: Methods and results. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1110–1121. IEEE.

[47] Wang, Z., Bovik, A. C., Sheikh, H. R., Simoncelli, E. P., et al. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612.

[48] Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., and Woo, W.-c. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810.

[49] Yang, J., Wright, J., Huang, T. S., and Ma, Y. (2010). Image super-resolution via sparse representation. *IEEE transactions on image processing*, 19(11):2861–2873.

[50] Yue, L., Shen, H., Li, J., Yuan, Q., Zhang, H., and Zhang, L. (2016). Image super-resolution: The techniques, applications, and future. *Signal Processing*, 128:389–408.

[51] Zhang, L., Zhang, H., Shen, H., and Li, P. (2010). A super-resolution reconstruction algorithm for surveillance images. *Signal Processing*, 90(3):848–859.

[52] Zhang, Y., Tian, Y., Kong, Y., Zhong, B., and Fu, Y. (2018). Residual dense network for image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[53] Zhu, S. and Ma, K.-K. (2000). A new diamond search algorithm for fast block-matching motion estimation. *IEEE transactions on Image Processing*, 9(2):287–290.