UNIVERSIDAD
DE GRANADA

Escuela Técnica Superior de Ingerías Informática y de Telecomunicación

Máster Universitario en Ciencia de Datos e Ingeniería de Computadores

TRABAJO FIN DE MÁSTER

Estimación de la incertidumbre en deconvolución ciega de imágenes usando Deep Image Prior Bayesiano

Presentado por:
Alejandro Pérez Lara

Tutor:
Rafael Molina Soriano
Departamento de Ciencias de la Computación e Inteligencia Artificial

Francisco Miguel Castro Macías
Departamento de Ciencias de la Computación e Inteligencia Artificial

Curso académico 2023-2024

Estimación de la incertidumbre en deconvolución ciega de imágenes usando Deep Image Prior Bayesiano

Alejandro Pérez Lara

Alejandro Pérez Lara *Estimación de la incertidumbre en deconvolución ciega de imágenes usando Deep Image Prior Bayesiano*.

Trabajo Fin de Máster. Curso académico 2023-2024.

Responsable de tutorización

Rafael Molina Soriano
Departamento de Ciencias de la Computación e Inteligencia Artificial

Francisco Miguel Castro Macías
Departamento de Ciencias de la Computación e Inteligencia Artificial

Máster Universitario en Ciencia de Datos e Ingeniería de Computadores

Escuela Técnica Superior de Ingerías Informática y de Telecomunicación

Universidad de Granada

Declaración de originalidad

D./Dña. Alejandro Pérez Lara

Declaro explícitamente que el trabajo presentado como Trabajo de Fin de Máster (TFM), correspondiente al curso académico 2023-2024, es original, entendida esta, en el sentido de que no ha utilizado para la elaboración del trabajo fuentes sin citarlas debidamente.

En Granada a 6 de septiembre de 2024

Fdo: Alejandro Pérez Lara

Índice general

Resumen

La calidad de las imágenes es un aspecto crucial en numerosos campos que abarcan desde la medicina hasta la astronomía. Sin embargo, las imágenes capturadas a menudo sufren degradaciones significativas, como el emborronamiento, esto provoca que se reduzca la utilidad de las mismas. La Deconvolución Ciega de Imágenes (Blind Image Deconvolution, BID) busca recuperar la calidad original de una imagen sin conocer el proceso específico que causó la distorsión. Esto convierte al BID en un problema complejo de resolver pero de gran relevancia.

En los últimos años, los avances en el Aprendizaje Profundo han abierto nuevas posibilidades para abordar el problema del BID. En particular, el uso de modelos como Deep Image Prior (DIP) ha demostrado ser una herramienta eficaz para mejorar la calidad de las imágenes. Uno de los enfoques más exitosos en este campo ha sido el modelo SelfDeblur, que ha obtenido grandes resultados aplicando la misma idea que DIP al problema de BID de forma específica. Sin embargo, tanto la formulación original de DIP como la de SelfDeblur no contemplan la incertidumbre presente en las reconstrucciones, lo cual es crucial en áreas como la medicina, donde las decisiones basadas en imágenes pueden tener consecuencias críticas.

El presente trabajo tiene como objetivo abordar esta limitación mediante la incorporación de técnicas de modelado Bayesiano a la resolución del problema de BID. Para ello, se propone una extensión del modelo SelfDeblur, llamado Bayes-SelfDeblur, que permite no solo la restauración de imágenes degradadas, sino también la estimación de la incertidumbre asociada a dichas restauraciones. Esto proporciona un valor añadido al proceso, ya que permite cuantificar el grado de confianza en las imágenes reconstruidas.

Para lograr nuestro objetivo haremos uso de las Redes Neuronales Bayesianas (Bayesian Neural Networks, BNN). Al hacer uso de las BNN necesitaremos aproximar distribuciones de probabilidad complejas por lo que haremos uso de Inferencia Variacional. Concretamente, usaremos Variational Dropout, por su simplicidad y su eficiencia. Gracias al uso de las BNN seremos capaces de calcular tanto la incertidumbre aleatoria como la incertidumbre epistémica de nuestras reconstrucciones.

Concretamente, en este trabajo se realizan las siguientes contribuciones:

1. Proponemos un nuevo modelo, denominado Bayes-SelfDeblur. Este modelo extiende a SelfDeblur incorporando mecanismos para la cuantificación de la incertidumbre.

2. Evaluamos el modelo propuesto en un conjunto clásico de BID. Nuestro modelo obtiene mejores resultados que SelfDeblur, el método de partida. Además, estudiamos los mapas de incertidumbre aleatoria y epistémica que se obtienen para cada una de las imágenes.

Palabras clave: Deconvolución Ciega de Imágenes, Deep Image Prior, Inferencia Variacional, Variational Dropout, SelfDeblur, Bayes-SelfDeblur, incertidumbre aleatoria, incertidumbre epsitémica, Aprendizaje Automático, Aprendizaje Profundo, Redes Neuronales.

Summary

Image quality is a crucial aspect in numerous fields ranging from medicine to astronomy. However, captured images often suffer significant degradation, such as blurring, which reduces their usefulness. Blind Image Deconvolution (BID) seeks to restore the original quality of the photos, without knowing the specific process that caused the distortion. This makes BID a complex problem that is of great relevance.

In recent years, advances in Deep Learning have opened up new possibilities for addressing the BID problem. In particular, the use of models such as Deep Image Prior (DIP) has proven to be an effective tool for improving image quality. One of the most successful approaches in this field has been the SelfDeblur model, which has obtained great results by applying the same idea as DIP to the BID problem specifically. However, both the original DIP and SelfDeblur formulations do not take into account the uncertainty present in reconstructions, which is crucial in areas such as medicine, where image-based decisions can have critical consequences.

The present work aims to address this limitation by incorporating Bayesian modeling techniques to the resolution of the BID problem. To this end, an extension of the SelfDeblur model, called Bayes-SelfDeblur, is proposed, which allows not only the restoration of degraded images but also the estimation of the uncertainty associated with such restorations. This provides an added value to the process since it allows quantifying the degree of confidence in the reconstructed images.

To achieve our objective we will make use of Bayesian Neural Networks (BNN). When making use of BNN we will need to approximate complex probability distributions so we will make use of Variational Inference. Specifically, we will use Variational Dropout, due to its simplicity and efficiency. Thanks to the use of BNN we will be able to calculate both the random uncertainty and the epistemic uncertainty of our reconstructions.

Specifically, the following contributions are made in this work:

1. We propose a new model, called Bayes-SelfDeblur. This model extends SelfDeblur by incorporating mechanisms for uncertainty quantification.

2. We evaluate the proposed model on a classical set of BIDs. Our model obtains better results than SelfDeblur, the starting method. In addition, we study the random and epistemic uncertainty maps obtained for each of the images.

Keywords: Blind Image Deconvolution, Deep Image Prior, Variational Inference, Variational Dropout, SelfDeblur, Bayes-SelfDeblur, aleatoric uncertainty, epistemic uncertainty, Machine Learning, Deep Learning, Neural Networks.

1 Deconvolución Ciega de Imágenes.

Las imágenes son una fuenta invaluable de información en numerosos campos de la ciencia y la vida cotidiana. Al ser capturadas, frecuentemente sufren diversos tipos de degradaciones que afectan a su calidad. Entre los problemas más comunes podemos destacar el emborronamiento que puede estar causado por el movimiento de la cámara, el desenfoque del objetivo, el ruido que puede estar generado por la baja iluminación, la compresión de la imágen, o la baja calidad de ésta.

La Deconvolución Ciega de Imágenes (Blind Image Deconvolution, BID) busca recuperar una imagen de mayor calidad a partir de una imagen degradada de acuerdo a algún proceso de emborronamiento desconocido. La complejidad de este problema, a diferencia de la deconvolución tradicional, donde la función de emborronamiento es conocida, reside en que no tenemos conocimiento sobre el proceso de degradación, lo que hace que encontrar una solución sea difícil.

Con la resolución del problema BID podemos mejorar significativamente la calidad de las imágenes, revelando detalles que de otro modo no serían visibles como podemos apreciar en la figura 1.1. Es por ello que la solución de este problema tiene multitud de aplicaciones:

- **Microscopía:** BID se utiliza para mejorar la resolución de imágenes microscópicas, permitiendo la visualización de estructuras celulares y subcelulares con mayor precisión.
- **Astronomía:** En el campo de la astronomía, BID se emplea para eliminar la distorsión atmosférica de las imágenes astronómicas, permitiendo obtener imágenes más nítidas de estrellas, galaxias y otros objetos celestes.
- **Medicina:** En el caso de las imágenes médicas, BID se utiliza sobre tomografías computarizadas y resonancias magnéticas, para mejorar la calidad de estas y así facilitar el diagnóstico de enfermedades.
- **Fotografía:** Se utiliza para mejorar la nitidez de fotografías, especialmente en condiciones de poca luz o con movimiento de la cámara.
- **Restauración de imágenes:** La recuperación de detalles perdidos o deteriorados de imágenes antiguas o dañadas por el paso del tiempo.

1.1. Formulación matemática del problema

En [1] se propone modelar el proceso de emborronamiento de acuerdo a la siguiente ecuación:

$$g(x) = \sum_{s \in S_h} h(x,s) f(s) + n(x), \qquad x = (x_1, x_2) \in S_f, \qquad S_f, S_h \subset \mathbb{R}^2 \tag{1.1}$$

donde tenemos que $f(s)$ representa la imagen original, $g(x)$ es la imagen observada, $h(x,s)$ es el emborronamiento y $n(x)$ es el ruido observado que puede estar causado por múltiples factores.

Figura 1.1: Comparativa entre el antes y el después de aplicar SelfDeblur [10]

El modelo descrito en la ecuación anterior 1.1 suele ser representado en términos matriciales mediante la siguiente expresión:

$$g = Hf + n \tag{1.2}$$

donde g es la imagen observada, f la imagen original, H es la matriz de emborronamiento y n el ruido.

BID tiene como objetivo determinar la imagen original f y el desenfoque producido por h. Si la convolución cumple:

$$g(x) = \sum_{y} h(y)f(x-y) + n(x) \tag{1.3}$$

entonces el proceso de emborronamiento es lineal espacialmente invariante. Este nombre se debe a que todos los píxeles han sufrido el mismo emborronamiento. Este es el tipo de emborronamiento que consideramos aquí.

1.2. Enfoques clásicos para la resolución del problema

Los enfoques clásicos para la restauración de imágenes utilizan un enfoque analítico. Estas técnicas analíticas describen explícitamente un modelo de deconvolución, que decide los criterios para obtener una solución y elige un procedimiento de optimización. Normalmente estas técnicas optimizan una función de energía para cada nueva imagen y núcleo de emborronamiento. Dentro de las técnicas analíticas podemos separarlas en dos grupos, las técnicas determinísticas y las técnicas estocásticas [11].

1.2.1. Técnicas determinísticas

Estos métodos formulan el problema como una tarea de optimización para una determinada métrica. El objetivo suele ser minimizar la norma l_2 de g y Hf.

$$min||g - Hf||^2 \tag{1.4}$$

Además de la métrica l_2, para orientar la solución hacia una imagen más realista, se incorpora conocimiento sobre como esta fue tomada originalmente. Este conocimiento se tiene de forma previa y es propio de cada imagen y de las condiciones en las que esta fue tomada.

Este proceso para incorporar esta información adicional se conoce como regularización.

La regularización intenta añadir limitaciones para garantizar la suavidad y reducir el emborronamiento, un ejemplo de regularización es no penalizar en exceso las discontinuidades para preservar los bordes. Existen distintas técnicas de regularización, cada una con sus propias ventajas y limitaciones.

1.2.2. Técnicas estocásticas

Los métodos estocásticos tratan la imagen desconocida f y el emborronamiento h como variables aleatorias con sus propias distribuciones de probabilidad. Para encontrar las soluciones más probables se utilizan varias aproximaciones. Partimos de $p(g|f,h)$, $p(f)$ y $p(h)$

- Máxima verosimilitud: Es un método que se basa en maximizar la probabilidad de observar la imagen emborronada g dadas f y h. Es decir maximizamos en f y h la probabilidad $p(g|f,h)$.

- Máximo a posteriori (MAP): Es similar a la máxima verosimilitud pero también tenemos en cuenta la distribución a priori de la imagen y el emborronamiento a partir del conocimiento previo que tenemos. En este caso maximizamos en f y h la expresión $p(g|f,h)p(h)p(f)$.

- Marco totalmente Bayesiano: Un enfoque más completo que considera distribuciones de probabilidad para todas las incógnitas (imagen, desenfoque y cualquier hiperparámetro utilizado en el modelo). Esto permite un tratamiento más riguroso de la incertidumbre en el proceso de restauración. En este caso nuestro objetivo es calcular $p(h,f|g)$.

Como se puede deducir de los métodos introducidos anteriormente, dentro del escenario ciego en el que desconocemos el núcleo de emborronamiento, las técnicas analíticas suelen utilizar un enfoque iterativo para estimar el desenfoque y la imagen nítida latente. La mayoría de estas técnicas se concentran primero en obtener una estimación precisa del núcleo de emborronamiento h en una fase inicial, en la cual la extracción de una imagen clara se restringe sólo a las características más destacadas que ayudan a estimar el desenfoque, y así intentar obtener una imagen nítida. Posteriormente, se requiere un enfoque no ciego en el que ya tenemos una primera aproximación del núcleo para calcular la imagen nítida, lo que lleva a una estrategia de fase dual (inicialmente estimando el desenfoque, seguido de una Deconvolución No Ciega de Imágenes (Non Blind Image Deconvolution, NBID)).

Los errores cometidos durante la fase inicial de estimación del desenfoque pueden causar problemas importantes, como distorsiones, incluso cuando existen pequeñas imprecisiones en el núcleo. Esto resalta la necesidad de desarrollar métodos NBID que puedan resistir las imprecisiones del núcleo de emborronamiento si lo que se quiere es tomar esta aproximación dual al problema del BID. Actualmente las investigaciones sobre NBID que aborden específicamente los errores de estimación de desenfoque son muy limitadas.

Cabe señalar que los errores de estimación del núcleo no son el único tipo de valor atípico. También pueden producirse violaciones locales del modelo de convolución debido a píxeles saturados, regiones subexpuestas o píxeles calientes o muertos. Para abordar estos tipos de valores atípicos, se han sugerido varios métodos NBID robustos. Pese a ello estas correcciones suelen hacer que los detalles de la imagen estén demasiado suavizados.

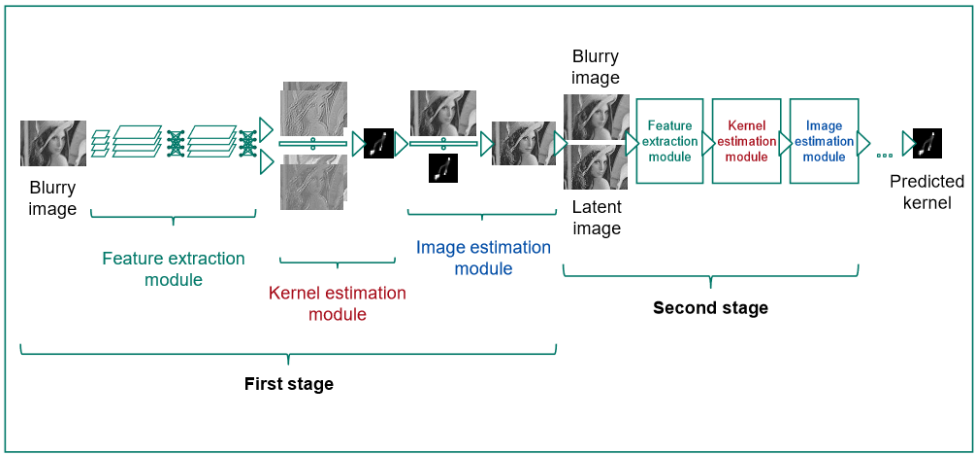

Figura 1.2: Imagen de la arquitectura de la red propuesta en *Learning to Deblur* [13].

1.3. Enfoques actuales para la resolución del problema

En la actualidad existen multitud de estudios que de una forma u otra aplican el Aprendizaje Profundo (Deep Learning, DL) al problema de BID siguiendo estrategias muy variadas que podrían dividirse en dos: desemborronamiento estático y dinámico. La diferencia entre estas dos estrategias reside en que el desemborronamiento estático explícitamente calcula el núcleo de emborronamiento mientras que el desemborronamiento dinámico calcula la imagen limpia directamente.

1.3.1. Desemborronamiento estático

Un ejemplo de desemborronameinto estático lo encontramos con la red neuronal propuesta en [13] que está diseñada para la estimación del núcleo de emborronamiento. En concreto, esta arquitectura de Red Neuronal Convolucional (Convolutional Neural Network, CNN) aborda el problema del BID en tres etapas como podemos apreciar en la figura 1.2:

1. Extracción de características: se computan representaciones de la imagen que son útiles para la estimación del núcleo de emborronamiento.

2. Estimación del núcleo: a partir de las características extraídas se estima el núcleo de emborronamiento.

3. Estimación de la imagen: tomando el núcleo de emborronamiento calculado en el paso anterior calculamos una aproximación de la imagen limpia.

Estas tres etapas se repiten con cada iteración de la red. Con la primera iteración únicamente extraemos las características de la imagen borrosa pero en las iteraciones siguientes disponemos de la imagen original latente. A partir de que se disponga de esta imagen se extraen las características de ella y a su vez las de la imagen borrosa inicial.

Otra solución al problema del BID sería la propuesta hecha en [14] que podemos apreciar en las figura 1.3. En este caso la estrategia es la de dividir la imágen en parches y a partir de estos estimar los núcleos de emborronamiento para cada uno de los parches utilizando

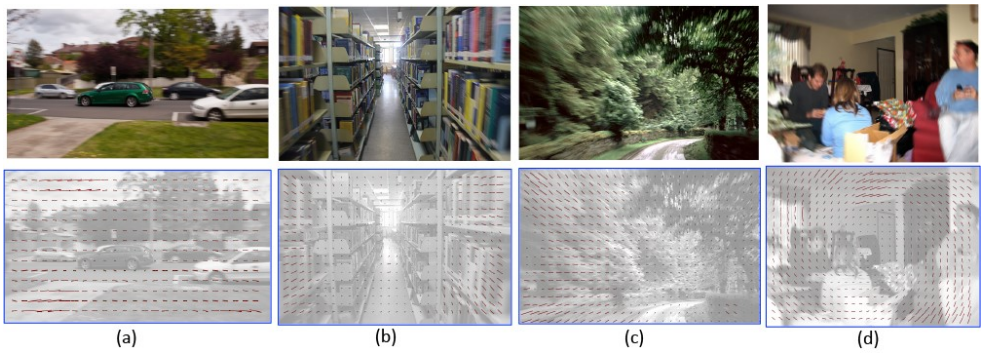

Figura 1.3: Imágenes emborronadas y sus correpondientes desenfoques de moviemiento, *Motion Blur* [14].

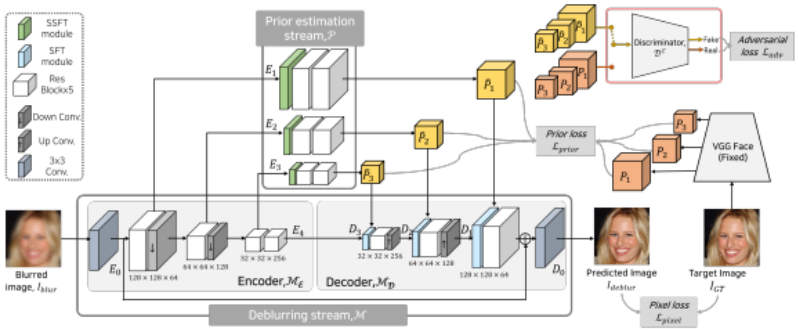

Figura 1.4: Arquitectura de la red DFGnet [7]

una CNN. A partir del núcleo de emborronamiento para cada parche se utilizó un modelo de campo aleatorio de Markov para fusionar todos los parches en un único núcleo de emborronamiento.

Para mejorar el rendimiento del modelo los autores rotaron las imágenes para calcular el núcleo de emborronamiento de los parches de las imágenes rotadas y como se conocía en ángulo de rotación a las que estas habían sido sometidas. Se utilizó esta información para obtener el emborronamiento de cada parche de una forma más precisa.

En los últimos años han surgido redes como la DFGnet [7] que se centra en eliminar el emborronamiento de imágenes de caras. Esta red se basa en dos flujos el de la estimación a priori y el desemborronamiento. Estos dos flujos se combinan al utilizar las Redes Generativas Adversarias (GAN, Generative Adversial Networks) donde una red preentrenada como puede ser VGGFace extrae las características de la imagen original sin emborronamiento y sus resultados se usan en la GAN para discriminar sobre las características de la imágenes son extraídas por la red encargada del desemborronamiento. En la figura 1.4 podemos apreciar la arquitectura de DFGnet.

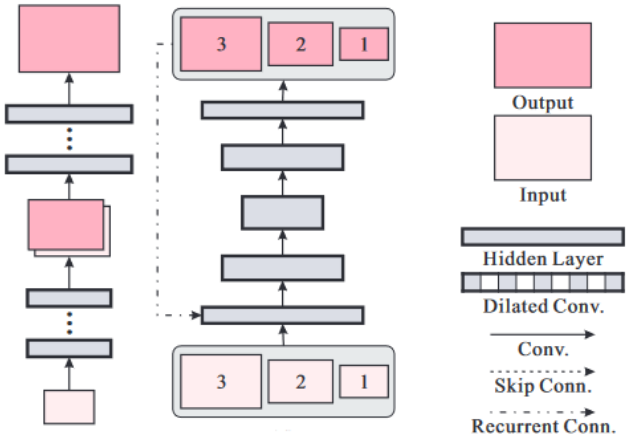

Figura 1.5: Comparativa entre la arquitectura simplificada de *multi-scale CNN* (izquierda) y *scale recurrent network* (derecha). Esquemas de [15]

1.3.2. Desemborronamiento dinámico

Dentro de las soluciones para el problema del BID que utilizan el desemborronamiento dinámico nos encontramos las CNN que toman como base diferentes resoluciones de la imágen para eliminar el emborronamiento. Esta idea ha sido utilizada en soluciones clásicas al problema del BID y ahora se ha aplicado a las CNN, donde tenemos la *multi-scale CNN* [12] y la *scale recurrent network* [15]. Aunque estas dos redes tengan una estrategia similar son diferentes en cuanto a su arquitectura como podemos ver en la figura 1.5.

En cuanto a su funcionamiento tenemos que la *multi-scale CNN* [12] toma como entrada lo que se conoce como una pirámide de imágenes borrosas. Esto significa que las imágenes reducen su resolución. Lo que hace que la red sea capaz de captar el emborronamiento en la imagen a partir de las múltiples resoluciones de la imagen.

Para ver la arquitectura de forma más detallada de *multi-scale CNN* disponemos de la figura 1.6. Como resultado de utilizar esta red lo que acabamos teniendo es una pirámide de imágenes latentes estimadas, que representa las estimaciones de las imágenes nítidas para cada escala. La información que se ha ido obteniendo en cada escala ha sido utilizada en las escalas posteriores para dar como resultado una reconstrucción de la imagen borrosa que toma como base las reconstrucciones en resoluciones inferiores.

En lo que se refiere a la *scale recurrent network* [15] se utiliza una red, que se va entrenando sobre las distintas resoluciones. Al hacer esto se introduce inestabilidad en la red puesto que se llega a un punto a que los parámetros se sobre ajustan para cada resolución en concreto. Pero si utilizamos la misma red con los mismo parámetros entrenables para cada resolución lo que tenemos es una simplificación de la misma lo que proporciona dos ventajas. La primera sería que el uso de las distintas resoluciones se puede llegar a ver como un aumento de datos. La segunda es la incorporación de módulos recurrentes como podemos apreciar en la figura 1.7.

A partir de lo que vemos en la figura 1.7 podemos apreciar que la imagen borrosa pasa a escalas inferiores para ir entrenando la red. La arquitectura de esta red se compone de bloques de codificación (encoder blocks, EBlocks) y de bloques de decodificación (decoder blocks, DBlocks). Los bloques residuales (residual blocks, ResBlocks) y las conexiones recurrentes juegan un papel clave en el rendimiento para el entrenamiento de la red y la reconstrucción

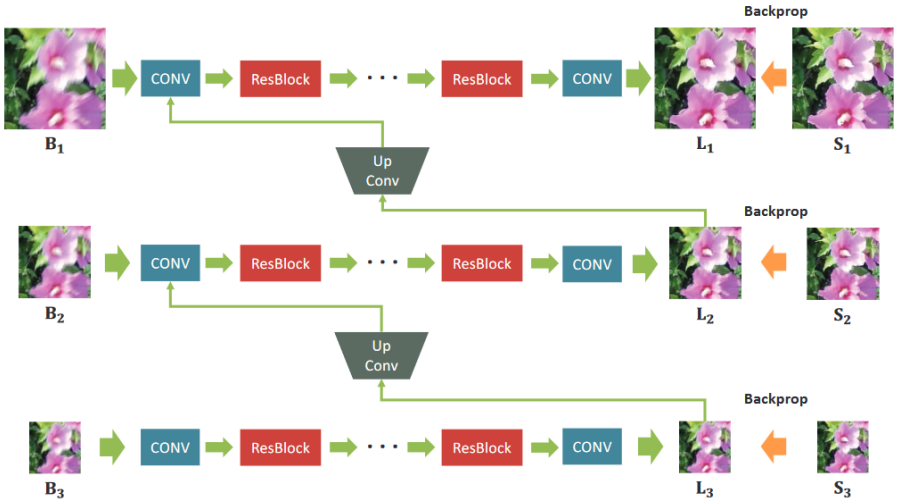

Figura 1.6: Esquema de funcionamiento de la arquitectura *multi-scale CNN* [12] donde podemos apreciar que cada nivel es una resolución distinta con la que se está trabajando para obtener una imagen latente sin emborronamiento a esa resolución.

de la imagen nítida.

Existe una técnica más moderna conocida como Deep Image Prior (DIP) [16, 6]. Este método reentrenan un modelo profundo para cada imagen y se ha utilizado en muchas tareas de visión de bajo de nivel, como la superresolución, el inpainting, el dehazing, la separación de transparencia, entre otras. Hablaremos de estas redes de forma más extensa en el capítulo 3.

Actualmente existen modelos como el BIRD (Blind Image Restoration via Fast Diffusion Inversion) [2] el cual optimiza conjuntamente los parámetros del modelo y la imagen restaurada haciendo uso de un modelo de difusión preentrenado del cual se extraen las características de las imágenes. En la figura 1.8 podemos apreciar la comparativa de múltiples modelos.

No hemos centrado específicamente en los modelos de DIP y de reescalado de resoluciones puesto que son la base del SelfDeblur [10] y nuestro trabajo se desarrollará sobre este para añadir información acerca de la incertidumbre.

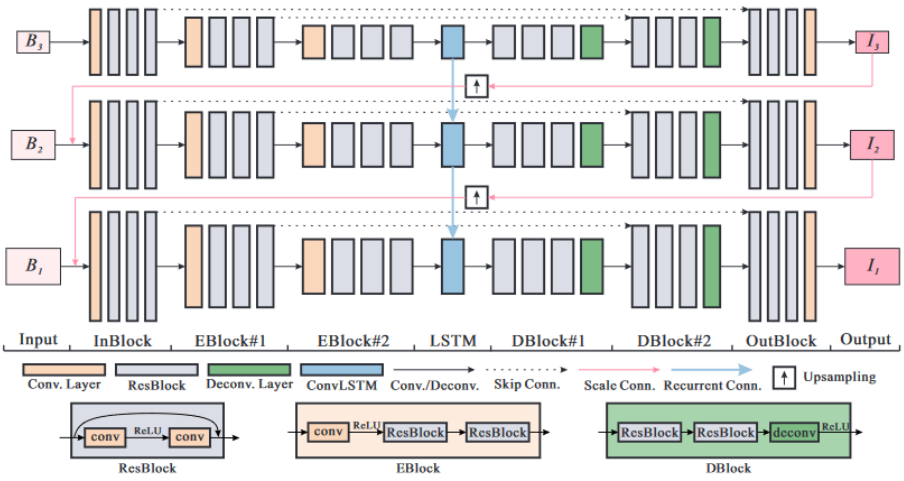

Figura 1.7: Esquema de funcionamiento de la arquitectura *scale recurrent network* [15].

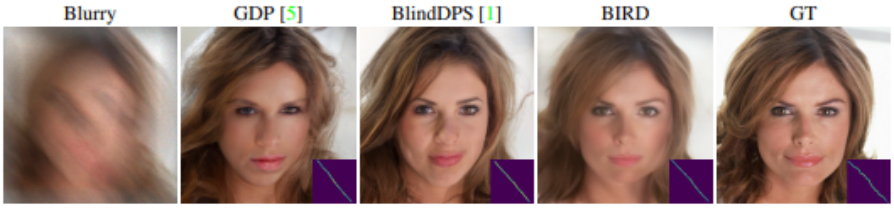

Figura 1.8: Comparativa entre múltiple modelos de desemborronamiento entre los que se encuentra el modelo BIRD [2].

2 Incertidumbre en Aprendizaje Profundo

Los modelos de aprendizaje profundo tienen un gran potencial para resolver tareas complejas al establecer relaciones entre datos de alta dimensionalidad y vectores de salida, pero estos a menudo carecen de la capacidad de cuantificar la incertidumbre de sus predicciones. Lo que puede llevar a resultados poco fiables, especialmente en tareas críticas como el diagnóstico médico o la conducción autónoma. Por lo tanto, es fundamental asociar un nivel de certeza a las salidas de estos modelos, permitiendo identificar y evitar aquellas predicciones en las que el modelo no tiene un alto grado de confianza.

En este capítulo vamos a explicar las redes neuronales bayesianas (Bayesian Neural Networks, BNN) y como éstas son capaces de capturar la incertidumbre aleatoria y epistémica de una red. Para ello primero vemos el modelado Bayesiano y como se utiliza este en las BNN.

2.1. Modelado Bayesiano

Dados unos ejemplos de entrada $X = \{x_1, x_2, \ldots, x_N\}$ con sus correspondientes etiquetas $Y = \{y_1, y_2, \ldots, y_N\}$ y fijada una clase de funciones f^ω parametrizadas por ω, el enfoque Bayesiano busca encontrar los parámetros ω de forma que nuestros datos hayan sido generados de acuerdo a $y = f^\omega(x)$ con una alta probabilidad. Para poder identificar los parámetros de estos resultados vamos a utilizar una distribución a priori $p(\omega)$ sobre el espacio de parámetros ω. Esta distribución $p(\omega)$ representa el grado de creencia previo sobre qué parámetros han generado nuestros datos antes de que estos hayan sido observados. A lo largo de esta sección seguiremos el trabajo [4].

Definimos la distribución a priori sobre ω y la distribución de probabilidad $p(y \mid X, \omega)$ que captura la información sobre que parámetros ω son más o menos probables a partir de los datos observados. Nuestro problema del BID es una regresión, puesto que lo podemos expresar según la ecuación 1.1, por lo que utilizaremos una distribución normal para ello [8]:

$$p(y \mid x, \omega) = \mathcal{N}(y \mid f^\omega(x), \tau^2) \tag{2.1}$$

donde τ es la varianza de nuestra distribución normal.

Dados el conjunto de datos (X, Y), podemos obtener la distribución a posteriori sobre el espacio de parámetros ω a través del teorema de Bayes:

$$p(\omega \mid X, Y) \propto p(Y \mid X, \omega) p(\omega) \tag{2.2}$$

donde $p(Y \mid X, \omega) = \prod_{n=1}^{N} p(y_n \mid x_n, \omega)$. Esta distribución captura los parámetros de función ω más probables dados nuestros datos observados al haberla expresado de forma que esta dependa del término $p(Y \mid X, \omega)$. Haciendo uso de esta distribución a posteriori podemos predecir la salida de un nuevo punto x^* integrando de la siguiente forma, este proceso se

conoce como inferencia:

$$p(y^* \mid x^*, X, Y) = \int p(y^* \mid x^*, \omega) p(\omega \mid X, Y) d\omega \tag{2.3}$$

En pocas situaciones, la distribución a posteriori $p(\omega \mid X, Y)$ puede evaluarse de forma analítica, para nuestro caso en el que tratamos con redes neuronales no es posible. Por ello como no podemos evaluarla, vamos a aproximarla por otra distribución que sí sabemos utilizar.

Comenzamos definiendo una distribución variacional aproximada $q_\theta(\omega)$, parametrizada por θ, que es fácil de evaluar. Nos gustaría que nuestra distribución $q_\theta(\omega)$ fuera lo más parecida posible a la distribución a posteriori obtenida a partir del modelo original. Para ello vamos a utilizar la divergencia de Kullback-Leibler (KL), que mide como de diferentes son dos distribuciones de probabilidad. Nuestro objetivo será el de minimizar la divergencia KL de las distribuciones $q_\theta(\omega)$ y $p(\omega \mid X, Y)$, para ello elegiremos la forma de la distribución $q_\theta(\omega)$ y en función de como de flexible sea ésta tendremos una aproximación mejor o peor de la distribución a posteriori:

$$KL(q_\theta(\omega), p(\omega \mid X, Y)) = \int q_\theta(\omega) \log \frac{q_\theta(\omega)}{p(\omega \mid X, Y)} d\omega \tag{2.4}$$

supondremos que esta integral está bien definida, para lo cual tendremos que asumir ciertas propiedades sobre $q_\theta(\omega)$, las cuales en la práctica siempre se cumplirán [19].

Al minimizar la divergencia de KL podemos aproximar la distribución a posteriori y haciendo uso de dicha aproximación sustituiremos en la ecuación 2.3 la distribución a posteriori $p(\omega \mid X, Y)$ por $q_\theta(\omega)$, quedando una expresión de la forma:

$$p(y^* \mid x^*, X, Y) \approx \int p(y^* \mid x^*, \omega) q_\theta(\omega) d\omega \tag{2.5}$$

La minimización de la divergencia de KL es equivalente a maximizar el límite inferior de evidencia (Evidence Lower Bound, ELBO), con respecto a los parámetros variacionales que definen a $q_\theta(\omega)$. La ELBO será la función de pérdida que utilizaremos más adelante para entrenar nuestra red:

$$\mathcal{L}_{VI}(\theta) := \int q_\theta(\omega) \log p(Y \mid X, \omega) d\omega - KL(q_\theta(\omega), p(\omega)) \leq \log p(Y \mid X) = \log \text{evidencia} \tag{2.6}$$

donde tenemos que el término de la divergencia $KL(q_\theta(\omega), p(\omega))$ de la expresión anterior es el regularizador y es el encargado de penalizar a las distribuciones que no se asemejan a la distribución a priori. Es por ello que a la hora de obtener la distribución $q_\theta(\omega)$ no se tiene en cuenta la complejidad de la distribución de la misma si no su parecido con la distribución $p(\omega \mid X, Y)$.

El procedimiento anteriormente descrito se conoce como inferencia variacional (variational inference, VI), y es una técnica estándar en la modelización bayesiana. La VI sustituye una distribución que no podemos evaluar analíticamente por otra que sea lo más parecida a esta. En comparación con los enfoques de optimización que se utilizan a menudo en el aprendizaje profundo, en este caso optimizamos distribuciones en lugar de estimaciones puntuales.

El enfoque de la VI conserva muchas de las ventajas de la modelización bayesiana (como el equilibrio entre modelos complejos y modelos que explican bien los datos), y da lugar a modelos probabilísticos que captan la incertidumbre del modelo. Aunque este procedimiento

hace que la inferencia sea analítica para una gran clase de modelos, sigue teniendo muchas carencias como el no ser escalable a grandes conjuntos de datos (evaluar 2.3 requiere cálculos sobre todo el conjunto de datos), y que la aproximación no se adapta a modelos complejos (modelos en los que esta última integral no puede evaluarse analíticamente).

2.2. Aprendizaje Bayesiano Profundo

Nuestro objetivo es capturar la incertidumbre de una red neuronal. Para lograrlo, suponemos que la función f^ω del apartado anterior es una red neuronal que estará compuesta por un total de L capas, cada una de ellas parametrizada por $\omega_i \in \mathbb{R}^{K_i K_{i-1}}$. Recogiendo estos parámetros en un vector $\omega = (\omega_1, \ldots, \omega_L)$, la distribución a priori que usaremos será $\omega \sim \mathcal{N}(0, I)$. De esta forma obtenemos una red neuronal Bayesiana (Bayesian Neural Network, BNN). Dependiendo de la estrategia que elijamos para aproximar la distribución predictiva $p(\omega \mid X, Y)$ obtendremos una variante diferente.

En nuestro caso la aproximación que vamos a utilizar es la de Variational Dropout [5]. Para ello vamos a definir nuestra distribución variacional aproximada como $q_\theta(\omega) = \prod_{i=1}^{L} q_{\theta_i}(\omega_i)$, donde $q_{\theta_i}(\omega_i)$ se define para cada una de las capas de la siguiente forma:

$$\begin{aligned} (\omega_i)_j &= (\theta_i)_j z_{ij}, \\ z_{ij} &\sim \text{Bernoulli}(p_i), \quad j \in \{1, \ldots, K_i K_{i-1}\}, \end{aligned} \tag{2.7}$$

donde $(a)_j$ denota al elemento j-ésimo de a, p_i es la probabilidad con la que se aplica dropout en la capa i-ésima y θ_i son los parámetros de la capa i-ésima.

Para obtener la función de pérdida con la que entrenar la red neuronal debemos aproximar los dos sumandos de la ecuación 2.6 ya que ninguno admite una expresión cerrada para la distribución variacional que hemos elegido. Para el primero vamos a utilizar la integración de Monte Carlo sobre el parámetro W, obteniendo un estimador sin sesgo de la ELBO,

$$\widehat{\mathcal{L}}_{VI}(\theta) = \frac{1}{T} \sum_{n=1}^{N} \sum_{t=1}^{T} \log p(y_n \mid x_n, \widehat{\omega}_t) - KL(q_\theta(\omega), p(\omega)), \tag{2.8}$$

donde $\widehat{\omega}_t \sim q_\theta(\omega)$. Para el segundo sumando usamos la aproximación propuesta en [5], obteniendo

$$KL(q(\omega), p(\omega)) \approx \sum_{i=1}^{L} \frac{p_i}{\tau^2 2} \|\theta_i\|^2 = \lambda \sum_{i=1}^{L} \|\theta_i\|^2, \tag{2.9}$$

donde en la segunda igualdad hemos escrito $\lambda = 0.5 p_i \tau^{-2}$, con $p = p_i$ para cada i. Esto se corresponde a usar la misma probabilidad de dropout en cada capa, lo cual suele ocurrir en la práctica. Finalmente, obtenemos la siguiente aproximación de la ELBO,

$$\mathcal{L}(\theta) \approx \frac{1}{T} \sum_{n=1}^{N} \sum_{t=1}^{T} \log p(y_n \mid x_n, \omega_t) - \lambda \sum_{i=1}^{L} \|\theta_i\|^2 = \tag{2.10}$$

$$= -\frac{1}{T} \sum_{n=1}^{N} \sum_{t=1}^{T} \left\{ \tau^{-2} \|y_n - f^\omega(x_n)\|^2 + \log \tau^2 \right\} - \lambda \sum_{i=1}^{L} \|\theta_i\|^2 \tag{2.11}$$

donde $\omega_t \sim q_\theta(\omega)$ y T es el número total de muestras que hemos tomado. Obsérvese que tomar una muestra de $q_\theta(\omega)$ consiste en aplicar dropout en cada capa de la red. Los pesos

de esta capas son θ_i y la variable binaria $z_{i,j} = 0$ se corresponderá con que la unidad j en la capa $i-1$ no esté activa. Este ELBO nos servirá, salvo el signo, como la función de pérdida a minimizar para nuestra BNN.

La distribución predictiva, dada por la ecuación 2.5, tampoco admite una expresión cerrada. Podemos aproximarla aplicando nuevamente integración de Monte Carlo:

$$p(y^* \mid x^*, X, Y) \approx \frac{1}{T} \sum_{t=1}^{T} p(y^* \mid x^*, \widehat{\omega}_t) \tag{2.12}$$

donde $\widehat{\omega}_t \sim q_\theta(\omega)$ y T es el número total de muestras de la red que hemos tomado para aproximar la distribución a posteriori.

Los resultados de esta sección nos permiten aproximar la distribución a posteriori de los pesos de la red que usaremos ahora para cuantificar la incertidumbre de nuestra red neuronal.

2.3. Tipos de incertidumbre: epistémica y aleatoria

La incertidumbre es un elemento fundamental de los modelos de aprendizaje profundo y esta debe de ser comprendida y gestionada de forma adecuada. Existen principalmente dos tipos de incertidumbre: la aleatoria y la epistémica. La incertidumbre epistémica, representa la variabilidad inherente al modelo que estamos utilizando. Por otro lado, la incertidumbre aleatoria, muestra la variabilidad presente en los datos. Ambos tipos de incertidumbre son críticos en los modelos, y su correcta identificación y manejo son clave para la integridad y confiabilidad de los resultados obtenidos.

A lo largo de esta sección vamos a ver como los dos tipos de incertidumbres se pueden obtener a través de la descomposición de la varianza de la distribución predictiva del modelo. Para ello vamos a empezar denotando como $q(y^* \mid x^*)$ a la distribución predictiva aproximada que tenemos en la ecuación 2.5,

$$q(y^* \mid x^*) = \int q_\theta(\omega) p(y^* \mid x^*, \omega) d\omega \tag{2.13}$$

A continuación vamos a obtener la varianza de esta distribución, la cual cuantifica la incertidumbre presente en las predicciones obtenidas. Nuestro objetivo será el de expresar la varianza en función de dos términos diferentes que representen a cada uno de los tipos de incertidumbre. La varianza se define como

$$\mathbb{V}_{q(y^* \mid x^*)}\left[y^*\right] = \mathbb{E}_{q(y^* \mid x^*)}\left[\|y^* - \mu^*\|^2\right] \tag{2.14}$$

donde $\mu^* = \mathbb{E}_{q(y^* \mid x^*)}\left[y^*\right]$. Siguiendo [3], obtenemos la siguiente descomposición,

$$\mathbb{V}_{q(y^* \mid x^*)}\left[y^*\right] = \underbrace{\mathbb{V}_{q_\theta(\omega)}\left[\mathbb{E}_{p(y^* \mid x^*, \omega)}\left[y^*\right]\right]}_{A} + \underbrace{\mathbb{E}_{q_\theta(\omega)}\left[\mathbb{V}_{p(y^* \mid x^*, \omega)}\left[y^*\right]\right]}_{B}. \tag{2.15}$$

El término A corresponde a la incertidumbre epistémica y el término B corresponde a la incertidumbre aleatoria.

La incertidumbre epistémica surge de la falta de conocimiento completo sobre la realidad que se modela y las limitaciones del modelo en sí. Esta incertidumbre se puede reducir al

utilizar una cantidad mayor datos para entrenar nuestro modelo puesto que estos aumentan la información y esto reduce la aleatoriedad del propio modelo haciendo que el espacio de búsqueda para la distribución a posteriori $q(\omega)$ se reduzca.

La incertidumbre aleatoria se asocia con el ruido presente en los datos y la aleatoriedad del proceso que los genera. En este trabajo nos centraremos en el escenario heterocedástico, en el que asumimos que el ruido depende de cada punto de entrada [8]

Ahora que hemos visto como se descompone la varianza de la distribución $q(y^*|x^*)$, ver 2.15 vamos a mostrar cómo cuantificar los dos tipos de incertidumbre que acabamos de presentar para la BNN que hemos definido en el apartado anterior. Para ello vamos a seguir el trabajo [8] en el que modifica la salida de la red de forma que estime una media y una varianza obteniendo la siguiente expresión:

$$f^{\omega}(x) = [\mu^{\omega}(x), \sigma^{\omega}(x)] \tag{2.16}$$

El modelo de observación lo definiremos como la siguiente distribución:

$$p(y \mid x, \omega) = \mathcal{N}(y \mid \mu^{\omega}(x), \sigma^{\omega}(x)^2 I) \tag{2.17}$$

Para la distribución variacional usamos Variational Dropout. Un razonamiento análogo a la sección anterior nos conduce a la aproximación de la correspondiente,

$$\mathcal{L}(\theta) \approx \frac{1}{T} \sum_{n=1}^{N} \sum_{t=1}^{T} \log p(y_n \mid x_n, \omega_t) - \lambda \sum_{i=1}^{L} \|\theta_i\|^2 = \tag{2.18}$$

$$= -\frac{1}{T} \sum_{n=1}^{N} \sum_{t=1}^{T} \left\{ \sigma^{\omega}(x_n)^{-2} \|y_n - \mu^{\omega}(x_n)\|^2 + \log \sigma^{\omega}(x_n)^2 \right\} - \lambda \sum_{i=1}^{L} \|\theta_i\|^2, \tag{2.19}$$

donde $\omega_t \sim q_\theta(\omega)$ y T es el número total de muestras que hemos tomado.

A continuación, buscamos obtener expresiones para cada uno de los sumandos de la ecuación 2.15. Cada muestra $\omega_t \sim q(\omega)$ de la red nos da una salida $\left[\mu^{\omega_t(x)}, \sigma^{\omega_t(x)}\right]$ que vamos a abreviar como $[\mu_t, \sigma_t]$. Siguiendo [8], aproximamos los sumandos de la siguiente forma:

$$A \approx \frac{1}{T} \sum_{t=1}^{T} \|\mu_t - \overline{\mu}\|^2, \tag{2.20}$$

$$B \approx \frac{1}{T} \sum_{t=1}^{T} \sigma_t^2 \tag{2.21}$$

Todo lo desarrollado en este capítulo lo aplicaremos ahora en el siguiente sobre la red neuronal SelfDeblur, haciendo que esta sea una BNN y que haga uso de Variational Dropout. Con esto aparte de obtener las reconstrucciones de las imágenes, tendremos la incertidumbre sobre cada punto de la reconstrucción que nos ofrece la red.

3 Deep Image Prior: arquitectura como distribución a priori

El proceso de restauración de imágenes, especialmente en BID y eliminación de ruido, ha sido un área central en la visión por computadora y el procesamiento de imágenes. Con la evolución de las CNN, se ha logrado un avance significativo en estas tareas, basándose en su capacidad para capturar características intrínsecas de las imágenes.

Tradicionalmente, se ha asumido que la eficacia de las CNN en la restauración de imágenes es un resultado directo de un entrenamiento exhaustivo sobre grandes conjuntos de datos etiquetados. Sin embargo, el trabajo [16] ha revelado que la estructura misma de las CNN posee una capacidad inherente para capturar las características de las imágenes (priors) en la restauración de imágenes.

En este capítulo, analizaremos la arquitectura de DIP como una herramienta para BID de imágenes. También veremos como se utiliza DIP en la arquitectura de SelfDeblur y por último utilizaremos Variational Dropout en la red de SelfDeblur para estudiar la incertidumbre de la red.

3.1. Deep Image Prior

El DIP se basa fundamentalmente en que la propia red sin entrenamiento previo va a ser la encargada de capturar la información a priori sobre la imagen. Esta red va a ser entrenada para una sola imagen y se escogerá su arquitectura para el problema en concreto que se quiera resolver puesto que el DIP no ha sido diseñado en exclusiva para resolver el problema de BID, pero en nuestro caso vamos a enfocar todo el desarrollo de DIP en dicho problema.

Partimos de una observación con emborronamiento x_0, en el caso de las CNN generativas tradicionales intentaríamos producir una imagen nítida x^* a partir de x_0 de la forma $x^* = f_\theta(x_0)$. Para ello la red f_θ tendría que haber sido entrenada previamente para establecer dicha relación. Pero en el caso de las redes DIP lo que vamos a resolver el problema de forma inversa, tratando de recuperar una imagen lo más nítida posible x^* utilizando únicamente la información proporcionada por la observación corrupta x_0.

A continuación vamos a ver como la propia arquitectura de la red captura la información implícita de la entrada sin que esta red haya sido entrenada previamente. Lo que haremos será definir la parametrización $x = f_\theta(z)$ para una imagen $x \in \mathbb{R}^{CHW}$ (con H siendo la altura de la imagen, W el ancho y C el número canales) donde $z \in \mathbb{R}^{C'H'W'}$ es un vector fijo que se utiliza como entrada para la red.

Para entender como funciona esta parametrización sobre los problemas de eliminación del ruido de una imagen vamos a expresar dicho problema como la minimización de una energía:

$$x^* = \min_x E(x; x_0) + R(x) \tag{3.1}$$

donde $E(x; x_0)$ es una término dependiente de los datos que se define en función del proble-

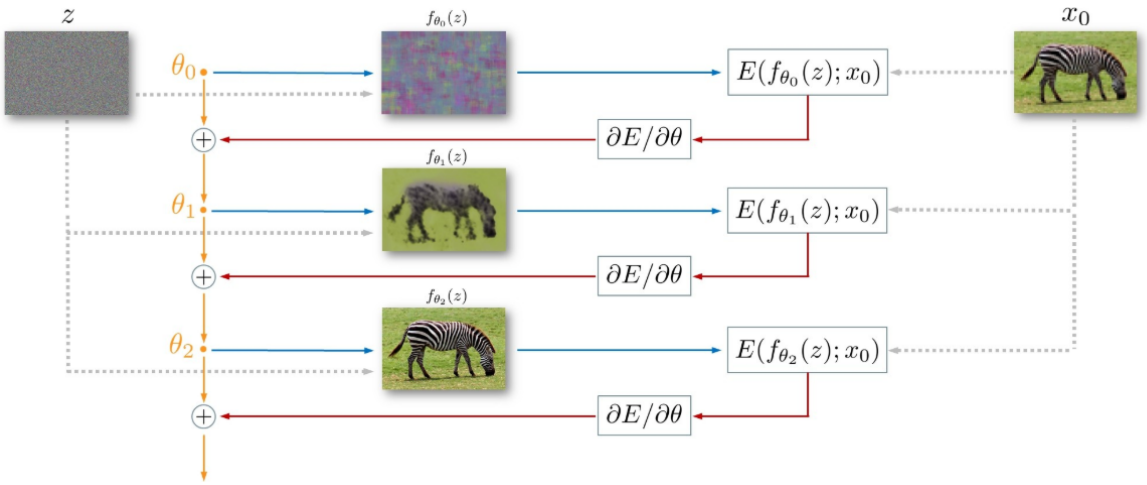

Figura 3.1: Imagen del funcionamiento del DIP, [16].

ma, x_0 es la imagen emborronada, x la imagen limpia y $R(x)$ es el regularizador que captura los prior de la imagen.

El término $E(x, x_0)$ se diseña para el problema concreto que se está tratando, en nuestro caso será la eliminación del ruido de la imagen. La parte complicada de la ecuación 3.1 será determinar el regularizador $R(x)$. Este puede ser tan sencillo como utilizar una norma (por ejemplo, la Variación Total) de la imagen o una CNN entrenada sobre un gran conjunto de datos.

En primer lugar observamos que podemos transformar la ecuación 3.1 utilizando una función g sobreyectiva de tal forma que tengamos la siguiente expresión:

$$x^* = \min_{\theta} E(g(\theta); x_0) + R(g(\theta)) \tag{3.2}$$

Al introducir esta g en la expresión 3.1, el espacio de búsqueda de la imagen cambia radicalmente, puesto que nuestro objetivo pasa a ser el de optimizar la función g variando θ. Ahora vamos a definir la función $g(\theta)$ para que tenga la forma $f_\theta(z)$, donde f_θ será nuestra red neuronal con parámetros θ y z será un parámetro de entrada fijo.

Si nuestra red f_θ es lo suficientemente buena podemos asumir que el término $R(g(\theta)) = 0$ puesto que la información de los priors será capturada por la propia red y nos quedaría la siguiente ecuación:

$$\theta^* = \text{argmin}_\theta E(f_\theta(z); x_0) \tag{3.3}$$

Ahora el foco de nuestro problema pasa a ser la obtención de θ^* se haría de manera iterativa y haciendo uso de un optimizador como el gradiente descendiente. Como podemos apreciar en la figura 3.1 comenzaríamos sobre una inicialización aleatoria θ_0^* de los pesos, donde iterativamente los actualizaríamos para minimizar la ecuación 3.3. Cada iteración t de los pesos, θ_t^* se asignarían a una imagen $x_t^* = f_{\theta_t^*}(z)$ para un z fijo como hemos mencionado anteriormente.

Una vez que el proceso termina habremos encontrado un θ^* para el cual podremos obtener la imagen restaurada x^* a partir de la siguiente ecuación $x^* = f_{\theta^*}(z)$. En caso de que

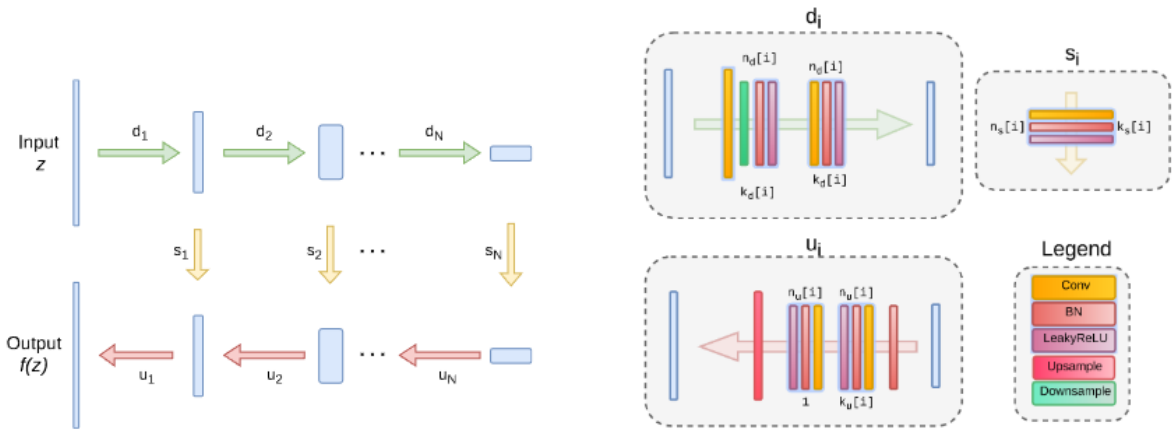

Figura 3.2: Imagen de la arquitectura DIP, [16].

apliquemos DIP para la resolución del problema BID definiendo una energía $E(\cdot,\cdot)$ adecuada estaremos obteniendo una imagen nítida pero no tendremos el núcleo de emborronamiento.

La arquitectura que se utiliza para el DIP es la podemos ver en la figura 3.2 y se conoce como reloj de arena (hourglass). La arquitectura utiliza una estructura de "decoder-encoder" y comienza con una imagen de entrada que pasa por varias capas de submuestreo (encoder), denotadas por $d_1, d_2, \ldots, d_k$), seguidas de varias capas de sobremuestreo (decoder) denotadas por $u_1, u_2, \ldots, u_k$.

Cada capa de submuestreo consta de una convolución (Conv), normalización por lotes (BN) y Función de activación mediante la Rectified Linear Unit (ReLU). Las capas de sobremuestreo consisten en sobremuestreo por el vecino más cercano seguido de capas de convolucón y funciones de activación ReLU.

3.2. SelfDeblur: Deep Image Prior para Deconvolución Ciega de Imágenes

Tomando como punto de partida el DIP [16], surgió el modelo del SelfDeblur que se centra en exclusiva en la resolución del problema del BID. La filosofía detrás de SelfDeblur es el uso de dos redes generativas, en la cual una de ella, $\mathcal{G}_x$ se encargará de capturar información a priori sobre la imagen x y la otra red $\mathcal{G}_k$ capturará información sobre del núcleo de emborronamiento k. A lo largo de esta sección seguiremos el trabajo [10].

Haciendo uso de la ecuación 1.1 que define el problema del BID vamos a definir la función de energía $E(\cdot;\cdot)$ que utilizaremos en la ecuación 3.1 de la sección anterior de la siguiente forma:

$$E(x;x_0) = ||k \otimes x - x_0||^2 \tag{3.4}$$

Siguiendo con el desarrollo de la ecuación 3.1 tenemos que para nuestro caso tenemos dos términos regularizadores, $\phi(x)$ como regularizador de la imagen y $\varphi(k)$ como regularizador del núcleo de emborronamiento k. Así, la ecuación 3.1 toma la siguiente forma:

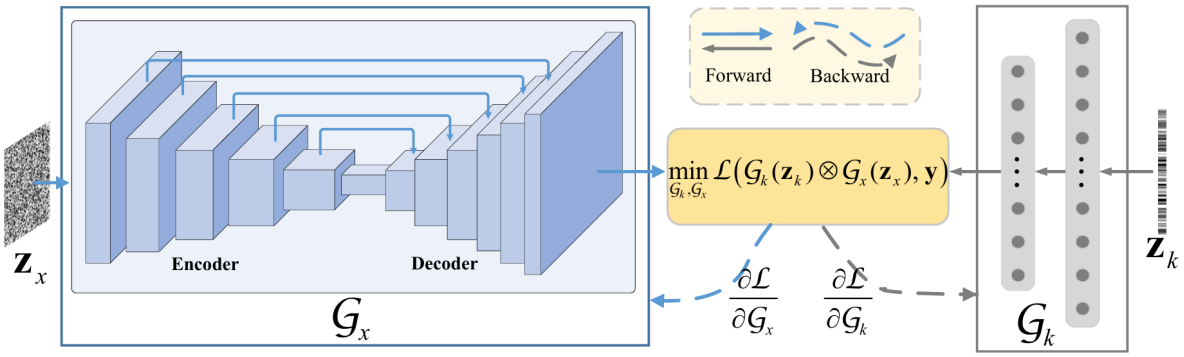

Figura 3.3: Imagen de la arquitectura SelfDeblur, [10].

$$(x^*, k^*) = \arg\min_{(x,k)} ||k \otimes x - x_0||^2 + \phi(x) + \varphi(k) \qquad 0 \leq x_i \leq 1, \forall i, \qquad k_j \geq 0 \quad \sum_j k_j = 1 \quad \forall j \tag{3.5}$$

donde x^* es la reconstrucción de la imagen x con valores de los píxeles limitados en el intervalo $[0, 1]$ y k^* la reconstrucción del núcleo de emborronamiento k con restricción de que todos los píxeles sean no negativos y que su suma sea la unidad.

Siguiendo un desarrollo similar al del DIP en la ecuación 3.3 tomamos las redes generativas $\mathcal{G}_x$ y $\mathcal{G}_k$ que sustituirán a la imagen x y al núcleo de emborronamiento k. Al igual que ocurría en el DIP podemos suponer que nuestras redes $\mathcal{G}_x$ y $\mathcal{G}_k$ son lo suficientemente buenas como para capturar los priors de x y k por lo que podemos prescindir de los términos de regularización $\phi(x)$ y $\varphi(k)$ quedando la siguiente expresión:

$$\min_{(\mathcal{G}_x,\mathcal{G}_k)} ||\mathcal{G}_k(z_k) \otimes \mathcal{G}_x(z_x) - x_0||^2 \qquad \begin{array}{c} 0 \leq (\mathcal{G}_x(z_x))_i \leq 1, \quad \forall i, \\ (\mathcal{G}_k(z_k))_j \geq 0, \quad \sum_j (\mathcal{G}_k(z_k))_j = 1, \quad \forall j \end{array} \tag{3.6}$$

donde z_x y z_k son muestras de una distribución uniforme, $(\cdot)_i$ y $(\cdot)_j$ son los elementos i y j de las etiquetas de las redes neuronales. La ecuación 3.6 establece la función objetivo de SelfDeblur. Ahora vamos a ver cual es la estructura de las redes utilizadas.

3.2.1. Red generativa de la imagen nítida

La red generativa $\mathcal{G}_x$ es la que utiliza la misma red que en DIP y esta debe de tener la capacidad de modelar con precisión las imágenes limpias latentes, que a menudo contienen estructuras complejas y texturas detalladas. Esta capacidad de modelado es esencial para reconstruir imágenes de alta calidad a partir de datos borrosos.

La arquitectura de $\mathcal{G}_x$ que podemos ver en la figura 3.3 se beneficia de la estructura de autoencoder, que permite una codificación eficiente de las características de la imagen y una posterior decodificación para reconstruir la imagen objetivo. Las conexiones de salto, por otro lado, ayudan a preservar la información espacial que podría perderse durante el proceso de codificación, lo que es fundamental para mantener la fidelidad de las texturas y los detalles finos en las imágenes restauradas.

3.2.2. Red generativa del núcleo de emborronamiento

La red generativa $\mathcal{G}_k$, se utiliza para modelar el núcleo de desenfoque k de manera efectiva, puesto que la red $\mathcal{G}_x$ se utiliza para capturar los priors de las imágenes pero su capacidad para modelar el núcleo de desenfoque de manera efectiva es limitada lo que lleva a la necesidad de una red generativa más especializada para esta tarea.

El núcleo de desenfoque k es significativamente más simple en términos de información requerida en comparación con la imagen limpia latente x. Esto se debe a que el núcleo de desenfoque, tiene una estructura y complejidad mucho menores pese a ser un componente crítico. Por lo tanto, una red Fully Connected Network (FCN) es suficiente para generar el núcleo de desenfoque, simplificando así el proceso de generación sin comprometer la calidad del resultado final.

La FCN $\mathcal{G}_k$, que podemos ver en la figura 3.3 está diseñada para ser simple pero efectiva. Acepta un vector de ruido z_k como entrada, lo que proporciona la variabilidad necesaria para la generación del núcleo. La red cuenta con una capa oculta, lo que permite una representación rica y detallada del núcleo de desenfoque. La salida consta de K^2 nodos, correspondientes a los elementos del núcleo de desenfoque que se desea generar.

Para asegurar que el núcleo de desenfoque generado cumpla con las restricciones físicas necesarias, como la no negatividad y la suma unitaria (que garantiza que el total de los pesos del núcleo sea igual a uno) de la ecuación 3.6, se aplica la función de activación SoftMax en la capa de salida. Esto transforma los valores de salida en una distribución de probabilidad, asegurando que todos los valores sean positivos y que su suma sea igual a uno.

Finalmente, la salida de la red se reorganiza en una estructura matricial para formar el núcleo de desenfoque $K \times K$. Este paso es crucial, ya que convierte la salida lineal de la red núcleo de emborronamiento. La simplicidad de la FCN $\mathcal{G}_k$, combinada con la aplicación de restricciones a través de la no linealidad SoftMax, demuestra ser una solución elegante y eficiente para la generación de núcleos de desenfoque en el proceso de SelfDeblur.

3.2.3. Algoritmo de optimización

El proceso de optimización de la ecuación 3.6 puede explicarse como un aprendizaje auto-supervisado, en el que las redes generativas $\mathcal{G}_k$ y $\mathcal{G}_x$ se entrenan utilizando únicamente una imagen de prueba (es decir, la imagen borrosa x_0) y no se dispone de una imagen limpia. Este proceso se caracteriza por un modelo de optimización conjunta que actualiza los parámetros de las redes generativas $\mathcal{G}_k$ y $\mathcal{G}_x$ de manera simultánea.

Tras alcanzar un número predefinido de iteraciones T, el algoritmo es capaz de generar tanto el núcleo de desenfoque estimado como la imagen limpia latente, utilizando las representaciones aprendidas por las redes $\mathcal{G}_k$ y $\mathcal{G}_x$. Donde el núcleo de desenfoque estimado y la imagen limpia latente tendrán la forma $k^* = \mathcal{G}_k^T(z_k)$ y $x^* = \mathcal{G}_x^T(z_x)$, respectivamente. A continuación vemos el algoritmo de SelfDeblur:

La implementación de SelfDeblur ha demostrado ser efectiva en la generación de imágenes desenfocadas visualmente plausibles y ha logrado ganancias cuantitativas notables en comparación con los métodos de deconvolución ciega del estado del arte. Además, la flexibilidad de esta arquitectura permite su aplicación en diversos escenarios de imágenes borrosas, desde fotografías tomadas en condiciones de baja luz hasta imágenes afectadas por movimiento.

Lo más notable es que la red SelfDeblur no requiere de imágenes limpias para el entrenamiento, lo que representa una ventaja significativa sobre otros métodos que dependen de

Algorithm 1 Algoritmo de optimización de la red SelfDeblur

Require: Imagen borrosa x_0
Ensure: Núcleo de emborronamiento k y la imagen limpia x
1: Muestra z_x y z_k de una distribución uniforme con semilla 0
2: **for** $t = 1$ hasta T **do**
3: Actualizamos la función de pérdida 3.6
4: Calculamos los gradientes con respecto a los parámetros de $\mathcal{G}_k$ y $\mathcal{G}_x$
5: Actualizamos los parámetros de $\mathcal{G}_x^\omega$ y $\mathcal{G}_k$
6: **end for**
7: $x = \mathcal{G}_x(z_x), \quad k = \mathcal{G}_k(z_k)$

grandes conjuntos de datos de imágenes nítidas para el aprendizaje supervisado.

3.3. Bayes-SelfDeblur: Deep Image Prior probabilístico para Deconvolución Ciega de Imágenes

En la sección anterior hemos presentado SelfDeblur, un método basado en DIP que permite estimar tanto el núcleo de emborronamiento como la imagen limpia. A pesar de que obtiene buenos resultados, puede ocurrir que en algunos casos la imagen original no se pueda recuperar correctamente [10]. En estos escenarios es necesario cuantificar la incertidumbre presente en nuestro modelo, para saber cuándo, y donde se está equivocando. En esta sección vamos a presentar Bayes-SelfDeblur, una extensión de SelfDeblur que hemos desarrollado en este Trabajo Fin de Máster, que es capaz de informar sobre la incertidumbre presente en las reconstrucciones producidas.

Nuestro objetivo será convertir la red neuronal $\mathcal{G}_x$ en una BNN siguiendo un proceso análogo al presentado en el capítulo 2. Para simplificar la notación, en lo que sigue vamos a omitir la dependencia respecto de z_x y z_k. Modificamos su salida para que estime una media y una matriz de covarianzas,

$$\mathcal{G}_x^\omega = [\mu^\omega, \Sigma^\omega], \tag{3.7}$$

donde $\mu^\omega \in \mathbb{R}^{CHW}$ es la media y $\Sigma^\omega = \text{diag}\left(\sigma_1^\omega, \ldots, \sigma_{CHW}^\omega\right) \in \mathbb{R}^{CHW \times CHW}$, siendo ω sus parámetros, con distribución a priori $p(\omega) = \mathcal{N}\left(\omega \mid 0, \tau^2 I\right)$. La salida de $\mathcal{G}_x^\omega$ parametrizará la distribución sobre la imagen limpia x, que ahora será una variable aleatoria,

$$p(x \mid \omega) = \mathcal{N}\left(x \mid \mu^\omega, \Sigma^\omega\right). \tag{3.8}$$

El modelo de observación del problema de deconvolución continúa siendo

$$p(x_0 \mid x) = \mathcal{N}\left(x \mid \mathcal{G}_k \otimes x, \gamma^2 I\right), \tag{3.9}$$

donde $\gamma^2 > 0$ es la varianza del ruido. Las propiedades de la distribución Gaussiana nos permiten concluir que

$$p(x_0 \mid \omega) = \mathcal{N}\left(x_0 \mid \mathcal{G}_k \otimes \mu^\omega, \gamma^2 I + S^\omega\right), \tag{3.10}$$

donde $S^\omega = \overline{\mathcal{G}_k} \Sigma^\omega \overline{\mathcal{G}_k}^\top$, siendo $\overline{\mathcal{G}_k}$ la matriz de convolución asociada a $\mathcal{G}_k$. La matriz S^ω es

de dimensiones $HCW \times HCW$, por lo que en la práctica no podremos trabajar con ella. Para solventar este problema, en este trabajo recurrimos a la aproximación $S^{\omega} \approx \Sigma^{\omega}$.

Desafortunadamente, no podemos calcular la distribución a posteriori $p(\omega \mid x_0)$ en forma cerrada. Por ello, vamos a proceder de forma análoga al capítulo 2. Como allí, vamos a aproximar tal distribución por una distribución variacional $q_\theta(\omega)$. Concretamente, usando Variational Dropout y un razonamiento análogo al del capítulo 2, obtenemos la siguiente aproximación del ELBO,

$$\mathcal{L}(\theta) \approx \frac{1}{T}\sum_{t=1}^{T}\{\log p(x_0 \mid \omega_t)\} - KL(q_\theta(\omega), p(\omega)) = \tag{3.11}$$

$$= -\frac{1}{T}\sum_{t=1}^{T}\left\{\sum_{i=1}^{CHW}\frac{((x_0)_i - (\mathcal{G}_k \otimes \mu^{\omega_t})_i)^2}{\gamma^2 + (\sigma_i^{\omega_t})^2} + \log\left(\gamma^2 + (\sigma_i^{\omega_t})^2\right)\right\} - \lambda\left\|\omega\right\|^2, \tag{3.12}$$

donde θ son los parámetros variacionales de $\mathcal{G}_x$, el subíndice i denota la i-ésima componente de cada vector, $\omega_t \sim q_\theta(\omega)$, T es el número de muestras. La expresión anterior puede ser maximizada respecto a θ y a los parámetros de $\mathcal{G}_k$, lo cual conduciría a una solución válida del problema de deconvolución.

Obsérvese que el modelo propuesto generaliza a SelfDeblur de forma probabilística. SelfDeblur trata como variables deterministas tanto a la imagen limpia x como a los parámetros ω. En el modelo presentado, esto consiste en hacer colapsar $p(x \mid \omega)$ y $p(\omega)$ a sus medias, ignorando sus varianzas. Esto se traduce en que si en la ecuación (3.11) tomamos $\sigma_i^{\omega} = 0$ y $\tau = 0$ recuperamos, salvo una constante, la función objetivo de SelfDeblur dada por la ecuación (3.6).

Una vez hayamos obtenido los parámetros variacionales óptimos, podemos recurrir a la distribución predictiva para estimar la imagen limpia y la incertidumbre en la misma. Esta viene dada por

$$p(x \mid x_0) = \int p(x \mid \omega)p(\omega \mid x_0)d\omega \approx \int p(x \mid \omega)q_\theta(\omega)d\omega. \tag{3.13}$$

Su media y su varianza pueden ser aproximadas usando T muestras de la distribución variacional,

$$\mathbb{E}_{p(x|x_0)}[x] \approx \frac{1}{T}\sum_{t=1}^{T}\mu^{\omega_t}, \tag{3.14}$$

$$\mathbb{V}_{p(x|x_0)}[x] \approx \frac{1}{T}\sum_{t=1}^{T}|\mu^{\omega_t} - \overline{\mu}|^2 + \frac{1}{T}\sum_{t=1}^{T}(\sigma^{\omega_t})^2, \tag{3.15}$$

donde $\overline{\mu} = T^{-1}\sum_{t=1}^{T}\mu^{\omega_t}$.

El proceso de optimización asociado a Bayes-SelfDeblur se recoge en el algoritmo 3.3. Partiendo de la imagen borrosa, los parámetros de $\mathcal{G}_k$ y $\mathcal{G}_x^{\omega}$ se ajustan para maximizar el objetivo dado por la ecuación (3.11). Cuando este proceso acaba, obtenemos las estimaciones de la imagen limpia y la incertidumbre en la misma usando las ecuaciones (3.14) y (3.15).

Algorithm 2 Algoritmo de optimización de la red Bayes-SelfDeblur

Require: Imagen borrosa x_0, número de muestras T
Ensure: Núcleo de emborronamiento k, imagen limpia x, y mapa de incertidumbre v
Muestra z_x y z_k de una distribución uniforme con semilla 0
for $t = 1$ hasta T **do**
Actualizamos la función de pérdida 3.6
Calculamos los gradientes con respecto a los parámetros de $\mathcal{G}_k$ y $\mathcal{G}_x$
Actualizamos los parámetros de $\mathcal{G}_x^{\omega}$ y $\mathcal{G}_k$
end for
$k = \mathcal{G}_k(z_k)$, $x = T^{-1}\sum_{t=1}^{T}\mu^{\omega_t}(z_x)$, $v = T^{-1}\sum_{t=1}^{T}|\mu^{\omega_t}(z_x) - \overline{\mu}|^2 + T^{-1}\sum_{t=1}^{T}\sigma^{\omega_t}(z_x)^2$

4 Experimentación

En el capítulo anterior hemos presentado tres modelos progresivamente más complejos: DIP, SelfDeblur, y Bayes-SelfDeblur. Este último es el nuevo método que proponemos en este trabajo. Consiste en una extensión de SelfDeblur capaz de estimar la incertidumbre en el proceso de desemborronamiento.

El objetivo de este capítulo es comparar el modelo propuesto con el modelo original. Para evaluar los modelos vamos a hacer uso de las métricas Structural Similarity Index Measure (SSIM) y Peak Signal-to-Noise Ratio (PSNR), además en el caso de nuestro modelo analizaremos la incertidumbre proporcionada por la varianza que vimos en el ecuación 3.7 del capítulo anterior.

4.1. Metodología y datos de la experimentación

Para llevar a cabo la experimentación hemos utilizado el conjunto de datos Levin [9] que se encuentra formado por un total de 4 imágenes que están tomadas haciendo uso de un trípode para evitar los movimientos en el eje z que se producen al coger la cámara con la manos. Al hacer esto tenemos que los ejes x e y quedan totalmente libres para aplicar el núcleo de emborramiento.

Para realizar el emborramiento de las fotografías se han tomado un total de 8 núcleos de emborronamiento que se han aplicado a las 4 imágenes. A partir de las imágenes emborronadas estas se recortaron para que tuvieran un tamaño de 255×255 píxeles dando como resultado un total de 32 imágenes sobre las que realizaremos nuestros experimentos. En la figura 4.1 tenemos las 4 fotografías y los 8 núcleos.

A partir de cada imagen del conjunto de Levin obtenemos una estimación de la imagen limpia, del núcleo de emborronamiento, y un mapa de incertidumbre usando el algoritmo 3.3 con $T = 100$. Compararemos estas estimaciones con las generadas por SelfDeblur usando el algoritmo 3.2.3.

La metodología que seguimos en la experimentación se basa en estudiar las métricas SSIM y PSNR para comprobar la calidad de nuestra reconstrucción con respecto a la imagen original sin emborronar, estas métricas se desarrollarán más a fondo en la siguiente sección.

En los experimentos vamos a estudiar el efecto que tiene el dropout en las reconstrucciones obtenidas. Para ello consideraremos dos variantes: aplicar el mismo dropout en cada capa (dropout uniforme), o aplicar un dropout distinto dependiendo de si la capa es encoder o decoder (dropout mixto). Sobre estas dos estrategias para aplicar dropout, haremos diversas pruebas variando los valores de dropout que estarán comprendidos entre 0.01 y 0.3. Con esta información determinaremos cual es la mejor configuración de dropout para nuestra propuesta de Bayes-SelfDeblur.

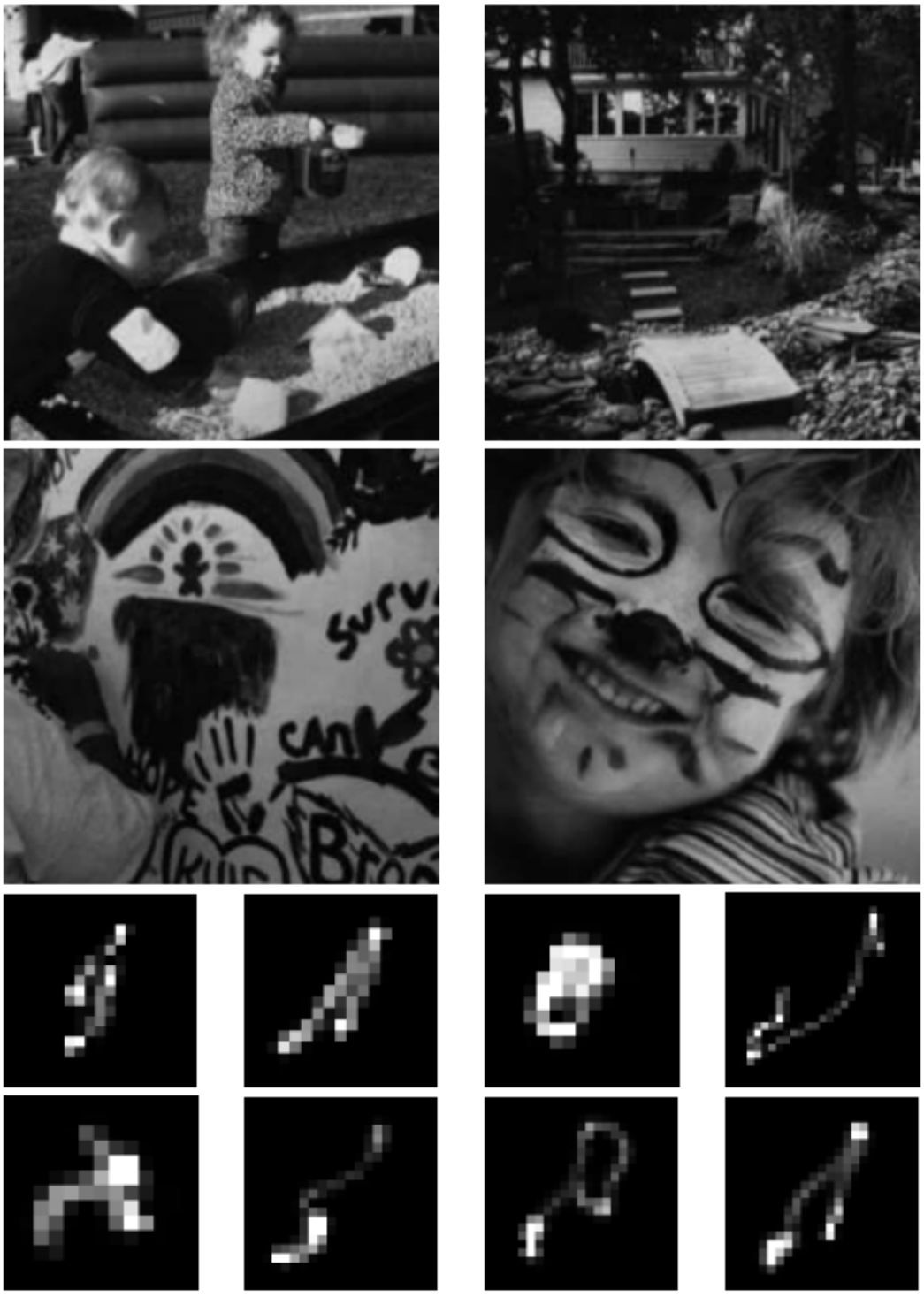

Figura 4.1: Imagen de las 4 fotografías y los 8 núcleos de emborronamiento que conforman el conjunto de datos Levin, [9].

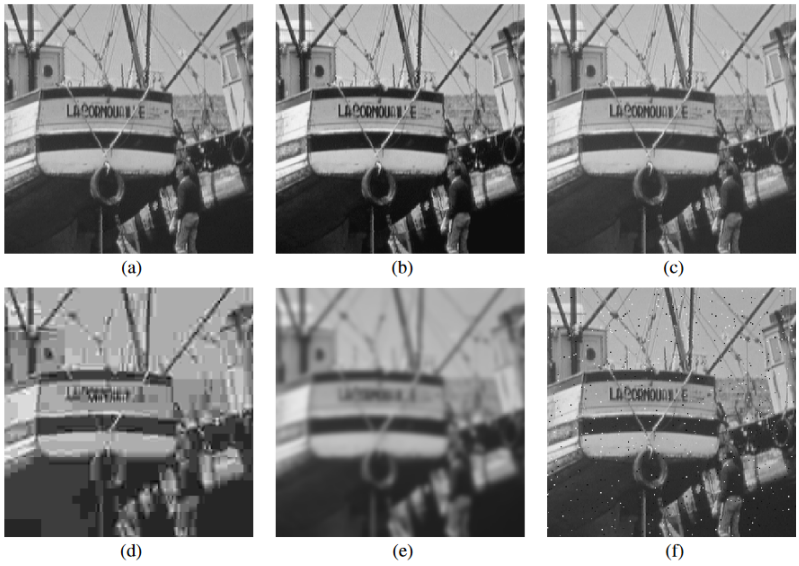

Figura 4.2: Comparación de 8 imágenes con diferentes tipos de distorsión y todas ellas con $MSE = 210$, [9].

4.2. Métricas utilizadas

En el ámbito del BID, la evaluación de la calidad de las imágenes recuperadas es un paso crucial para validar la efectividad de los métodos aplicados. Dos de las métricas más comúnmente utilizadas para este propósito son el Structural Similarity Index Measure (SSIM) y el Peak Signal-to-Noise Ratio (PSNR). Estas métricas permiten cuantificar la similitud y la fidelidad de una imagen procesada en comparación con una imagen de referencia, proporcionando una evaluación objetiva de la calidad de la restauración.

SSIM es una métrica que considera cambios en la estructura, luminancia y contraste entre las imágenes, y es especialmente útil para evaluar cómo percibe la calidad el sistema visual humano. PSNR, por otro lado, se basa en la diferencia cuadrática media entre los valores de la imagen original y la procesada, siendo una métrica ampliamente aceptada en el campo del procesamiento de imágenes por su simplicidad y relación directa con la distorsión en la imagen.

En esta sección, se describirán estas métricas en detalle y se explicará cómo se han aplicado en el contexto de este trabajo para evaluar los resultados obtenidos a partir de la aplicación del SelfDeblur Bayesiano en BID.

4.2.1. Structural Similarity Index Measure

Tradicionalmente la calidad de una imagen se media por las diferencias visibles entre una imagen de referencia y una distorsionada como el Error Cuadrático Medio (MSE). Sin embargo esto no siempre se corresponde con la percepción de la calidad por parte de un ser humano, un ejemplo lo podemos ver en la figura 4.2. Es por esto que surgen métricas más modernas como el SSIM [18] que se centran en como una imagen distorsionada preserva las estructuras de la imagen original.

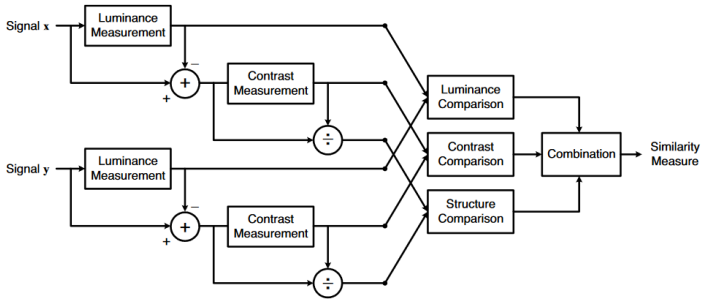

Figura 4.3: Esquema del funcionamiento de SSIM, [18].

En la figura 4.3 se muestra el diagrama del sistema de evaluación de la calidad del SSIM. En él tenemos dos señales de imágenes no negativas x e y que se han alineado entre sí (por ejemplo, parches espaciales extraídos de cada imagen). Si consideramos que una de las señales tiene una calidad perfecta, la medida de la similitud puede servir como medida cuantitativa de la calidad de la segunda señal. El sistema separa la tarea de medir la similitud en tres medidas de comparación más simples:

- **Luminancia:** Es una medida fotométrica que describe la intensidad de luz emitida, reflejada o transmitida por una superficie en una dirección específica, por unidad de área, y en función del ángulo desde el cual se observa. Es fundamental en la percepción visual del brillo y la calcularemos haciendo uso de la siguiente ecuación:

$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \tag{4.1}$$

  donde μ_x y μ_y son la media de de cada imagen, y $c_1 = (k_1 L)^2$ donde normalmente $k_1 = 0.01$ por defecto y $L = 2^b - 1$ donde b es el número de bits por píxel.

- **Contraste:** Es la diferencia en luminosidad o color que hace que un objeto o área se distinga visualmente de otros en una imagen o entorno. Para calcularlo utilizaremos la siguiente ecuación:

$$c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \tag{4.2}$$

  donde σ_x y σ_y son la varianza de cada imagen, y $c_2 = (k_2 L)^2$ donde normalmente $k_2 = 0.03$ por defecto.

- **Estructura:** aquellos atributos que representan la organización de los objetos de la escena, independientemente de la luminancia y el contraste. En nuestro caso lo mediremos con la siguiente ecuación:

$$s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3} \tag{4.3}$$

  donde σ_{xy} es la covarianza de las imágenes x e y, y $c_3 = c_2/2$ donde normalmente $k_2 = 0.03$ por defecto.

Una vez que tenemos definidas las tres medidas obtenemos el SSIM como una combinación ponderada de éstas:

$$\text{SSIM}(x, y) = l(x, y)^{\alpha} \cdot c(x, y)^{\beta} \cdot s(x, y)^{\gamma} \tag{4.4}$$

en concreto si tomamos $\alpha = \beta = \gamma = 1$ tenemos la expresión con la que calcularemos el SSIM:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{4.5}$$

4.2.2. Peak Signal-to-Noise

El PSNR es una métrica ampliamente utilizada para evaluar la calidad de imágenes y videos, comparando una versión comprimida o alterada de una imagen con su versión original. Esta métrica mide la relación entre la señal máxima posible (el valor máximo de un píxel) y el ruido, entendido como las diferencias entre la imagen original y la distorsionada.

Matemáticamente, el PSNR se expresa en decibelios (dB) y se calcula a partir del MSE, que cuantifica las diferencias promedio entre los píxeles correspondientes de ambas imágenes. El PSNR se define como:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (x - y)^2 \tag{4.6}$$

$$PSNR = 10 \log_{10} \left(\frac{L^2}{MSE} \right) \tag{4.7}$$

donde L es el valor máximo de la imagen (por ejemplo, 255 para imágenes de 8 bits).

Un PSNR más alto indica que la imagen alterada es más similar a la original, es decir, tiene menos distorsión, un ejemplo de esto lo vemos en figura 4.4. Típicamente, un PSNR de 30 dB o más se considera que indica una buena calidad de imagen, aunque esto depende del contexto específico y de la percepción visual humana. El PSNR es útil porque es fácil de calcular y proporciona una medida cuantitativa de la fidelidad de la imagen. Sin embargo, como mencionamos en la sección anterior tiene limitaciones, ya que no siempre se correlaciona bien con la percepción visual humana; pequeñas diferencias en áreas de alta frecuencia (bordes y detalles) pueden no reflejarse adecuadamente en el valor de PSNR.

4.3. Resultados y discusión

En esta sección vamos a presentar los resultados de los experimentos que hemos realizado. Comenzamos estudiando los valores medios de SSIM y PSNR de las imagenes reconstruidas. Con esta información podemos determinar cual es la forma más óptima de aplicar dropout para Bayes-SelfDeblur sobre el conjunto de datos Levin. Por último, mostraremos las imágenes de los dos tipos de incertidumbre junto con la incertidumbre total haciendo uso de lo que vimos en el capítulo 2 2.

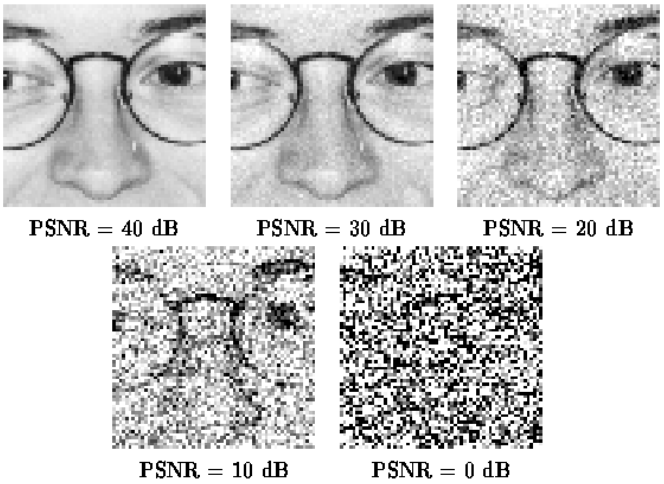

Figura 4.4: Comparación de una misma imagen según su PSNR [17].

4.3.1. Estimación de la imagen limpia

En este apartado estudiaremos cual es la mejor forma de aplicar dropout a nuestro modelo para obtener los mejores resultados. Posteriormente, compararemos el modelo obtenido con SelfDeblur.

La estrategia que hemos seguido para aplicar dropout tiene dos enfoques, por un lado hemos aplicado un dropout uniforme a todas las capas de la red y por otro lado hemos aplicado un dropout mixto que diferencia el dropout entre las capas encoder y decoder. En ambos casos los valores de dropout oscilarán entre 0.01 y 0.3. Con estos resultados vamos a comparar el modelo obtenido con SelfDeblur.

Para comparar el modelo original de SelfDeblur con los diferentes valores de dropout aplicados en nuestro modelo Bayes-SelfDeblur utilizaremos la tabla 4.1. Esta tabla se compone de 4 columnas:

- **Nombre del modelo:** donde especificamos si se trata de SelfDeblur o una de las dos variantes de nuestro modelo, Bayes-SelfDeblur Uniforme (modelo que aplica el dropout uniforme) o Bayes-SelfDeblur Mixto (modelo que aplica el dropout mixto).

- **Dropout:** donde especificamos la cantidad de dropout. Esta columna estará formada por un único número para los casos de SelfDeblur y Bayes-SelfDeblur Uniforme. Mientras que para el caso de Bayes-SelfDeblur Mixto estará formado por una pareja de valores (x, y) donde x especifica el dropout para las capas encoder e y especifica el dropout para las capas decoder.

- **SSIM:** esta columna a su vez se subdivide en otras dos, la de la métrica en si misma que se representa con el SSIM junto con la desviación típica y el rango que nos da el orden de los mejores resultados del SSIM.

- **PSNR:** al igual columna en esta contamos con la métrica que en este caso es el PSNR junto con su desviación típica y el rango de la misma.

Nombre del modelo	Dropout	SSIM		PSNR	
		Métrica	Rango	Métrica	Rango
SelfDeblur	0	0.575097 ± 0.099617	7	21.204795 ± 1.686948	7
Bayes-SelfDeblur Uniforme	0.01	0.640370 ± 0.061392	4	21.787695 ± 1.041697	5
	0.02	0.633201 ± 0.073589	5	21.862028 ± 0.930043	4
	0.05	$\mathbf{0.640743} \pm 0.064406$	3	$\mathbf{22.079172} \pm 1.039399$	3
	0.1	0.582138 ± 0.061592	6	21.350542 ± 0.373096	6
	0.15	0.500928 ± 0.110419	9	20.543515 ± 1.069418	9
	0.2	0.396141 ± 0.113800	11	19.128900 ± 1.097328	11
	0.25	0.323401 ± 0.077860	12	18.115718 ± 0.976739	12
	0.3	0.290741 ± 0.055940	13	17.484460 ± 0.542443	13
Bayes-SelfDeblur Mixto	(0.01, 0.1)	0.546890 ± 0.104015	8	20.638285 ± 1.073620	8
	(0.02, 0.15)	0.457349 ± 0.113864	10	19.374816 ± 1.215992	10
	(0.1, 0.01)	$\mathbf{0.670147} \pm 0.054326$	1	22.481286 ± 0.757867	2
	(0.15, 0.02)	0.661804 ± 0.053230	2	$\mathbf{22.593089} \pm 0.890038$	1

Tabla 4.1: Tabla con los resultados para el conjunto de datos Levin.

A partir de los resultados de la tabla 4.1 observamos que modelo original de SelfDeblur se ve superado por ambas variaciones de nuestro modelo. A continuación analizamos las dos variaciones de Bayes-SelfDeblur.

En la tabla 4.1 se aprecia que el dropout uniforme obtiene los mejores resultados al aplicar un dropout bajo que se encuentre comprendido entre 0.01 y 0.05. En concreto el mejor resultado tanto para SSIM como para PSNR se obtiene para un dropout de 0.05. Comparando la variante de dropout unifome de Bayes-SelfDeblur con el modelo original tenemos que para dropouts inferiores a 0.1 nuestros resultados son mejores tanto en SSIM como en PSNR para el conjunto de datos seleccionado.

Los peores resultados del dropout uniforme se obtienen para un valor de dropout igual o superior a 0.15 a partir de la información de la tabla 4.1 donde se aprecia claramente como la calidad de la reconstrucción empeora conforme aumenta el dropout que aplicamos. Esto se debe a que un dropout muy elevado hace que se pierda la información a priori que extrae la propia red. Esta información es clave a la hora de reconstruir la imagen por lo que tener un dropout elevado da peores resultados.

Cuando analizamos el caso del dropout mixto tenemos que los mejores resultados tanto para SSIM como para PSNR se dan cuando aplicamos un dropout alto a la capa encoder y uno bajo a la capa decoder. Dando así como resultado que las mejores reconstrucciones se obtienen al tener 0.1 de dropout en el encoder y 0.01 en el decoder.

Los peores resultados los tenemos al aplicar un dropout mixto en el que hay un dropout elevado en el decoder y un dropout bajo en el encoder. Aunque estos resultados siguen siendo superiores a los de aplicar un dropout uniforme de 0.2 no son lo suficientemente buenos como para batir el modelo original de SelfDeblur. Esto se debe a que la propia arquitectura de la red es la encargada de capturar la información a priori para la reconstrucción pierde demasiada información en las capas encoder que son las encargadas de generar la imagen limpia.

Como hemos dicho las mejores reconstrucciones se obtienen con 0.1 de dropout en el encoder y 0.01 en el decoder. Que este modelo sea el mejor es debido a que si tenemos un dropout un poco más elevado en la capa encoder la variabilidad de la información a priori que captura el encoder será mayor y gracias a un dropout no muy elevado en la capa decoder podemos mantener unos resultados consistentes y que van mejoran gracias a que la información proporcionada por el encoder es mayor al tener un dropout de 0.1.

Si nos fijamos en el SSIM de la tabla 4.1 tenemos que para varias elecciones de dropout nuestro modelo Bayes-SelfDeblur obtiene una mejor reconstrucción en media que el modelo original de SelfDeblur. Esto implica que en media nuestro modelo visualmente ofrece una imagen más nítida que la de SelfDeblur. Además la varianza que tiene Bayes-SelfDeblur es inferior a la SelfDeblur.

En cuanto al PSNR podemos apreciar que nuestro modelo vuelve a ser superior para varias elecciones de dropout respecto a SelfDeblur por lo que las reconstrucciones tendrán menos ruido que las ofrecidas por el modelo original. Esto también se refleja en la disminución de las imperfecciones, y al igual que ocurre con el SSIM la varianza vuelve a ser inferior para nuestro modelo lo que nos dará resultados más consistentes.

4.3.2. Estimación de la incertidumbre

En esta sección vamos a analizar las reconstrucciones obtenidas junto con los mapas de incertidumbre para cada una de las imágenes del conjunto de datos. Para calcular el mapa de incertidumbre epistémica hemos hecho uso con la ecuación 2.20 y para el cálculo del mapa de la incertidumbre aleatoria hemos utilizado la ecuación 2.21. Los resultados se muestran en la figure 4.5.

En la imagen de la primera fila observamos que la mayor cantidad de incertidumbre aleatoria se encuentra en los dos niños y en el cajón de arena que tienen a sus pies. Esto se debe a que como podemos ver en 4.1 las texturas de la ropa y el cajón, al igual que la complejidad de la cara pierden mucha información al ser emborronadas. En cuanto a la incertidumbre epistémica es superior en las siluetas de los niños y sus sombras. En concreto destacamos el parche en el brazo del niño y la silueta de la niña que se encuentra en un plano más alejado.

En la imagen de la segunda fila tenemos que la incertidumbre epistémica se da especialmente en los contornos de la casa como las ventanas y el puente que se encuentra en la parte inferior. La incertidumbre aleatoria resalta en las zonas de la parte inferior derecha de la imagen al igual que la casa del fondo. Si nos fijamos en la fotografía original 4.1 la zona inferior derecha tiene texturas complejas de la vegetación y el puente, además la zona de la casa tiene una sombra por lo que hace que la textura no sea uniforme.

En la imagen de la tercera fila no se aprecia ninguna zona en la que resalte la incertidumbre aleatoria. En cuanto a la incertidumbre epistémica las zonas que más destacan son los bordes de los elementos que componen la imagen. Si nos fijamos en la imagen original de 4.1 tenemos que hay varios elementos con una textura bastante uniforme por lo que el emborronamiento no las difumina pero sí se difuminan las fronteras de dichos elementos, por esto en la reconstrucción el modelo, señala los bordes como las zonas de mayor incertidumbre.

En la imagen de la cuarta fila tenemos que la incertidumbre epistémica vuelve a ser elevada en el contorno de las líneas del rostro del niño y en las rallas de la camiseta de este. La incertidumbre es elevada en la parte inferior derecha. Si nos fijamos en fotografía original 4.1 la parte inferior derecha se corresponde con el pelo del niño que tiene una textura más compleja.

A partir de esto mapas hemos visto que la incertidumbre aleatoria es mucho mayor en las zonas que tienen una textura compleja puesto que cuando se lleva a cabo el emborronamiento perdemos mucha información en dichas zonas. Mientras que para la incertidumbre epistémica nos encontramos el modelo señala zonas de contorno de los distintos elementos de la fotografía debido a que estas líneas de contorno quedan difuminadas tras el emborronamiento.

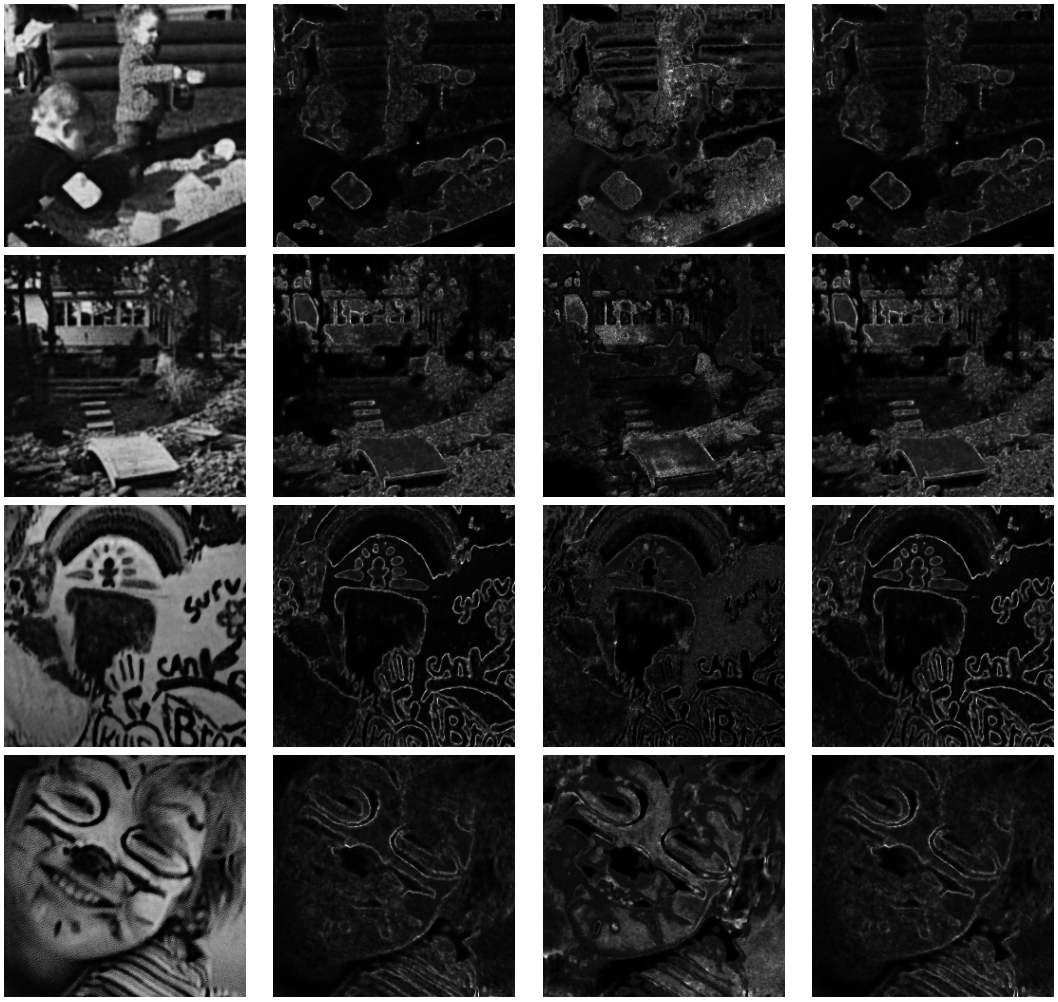

Figura 4.5: De izquierda derecha: imagen reconstruida, incertidumbre epistémica, incertidumbre aleatoria, suma de ambas.

5 Conclusiones y trabajo futuro

El objetivo de este capítulo es sintetizar las lecciones aprendidas a lo largo del trabajo y explorar las posibles direcciones que podrían guiar investigaciones futuras. Comenzamos con un resumen de los problemas abordados y las conclusiones obtenidas tras su resolución. Luego, identificamos áreas clave donde se podrían enfocar esfuerzos adicionales de cara al posible trabajo futuro.

5.1. Conclusiones

Comenzamos este trabajo hablando sobre la importancia que tienen las fotografías en distintos aspectos de nuestra vida y como son cruciales en ciertas ramas de la ciencia como puede ser la medicina para el diagnóstico de enfermedades. Para abordar este problema nos hemos centrado en resolver el problema del BID para recuperar una imagen de la mayor calidad posible a partir de una imagen degradada por un proceso de emborronamiento desconocido.

Para resolver el problema del BID nos hemos centrado en las estrategias que hacen uso del Aprendizaje Profundo. En concreto los modelos de DIP y SelfDeblur los cuales no necesitan de un entrenamiento previo y se sirven de la propia arquitectura de la red para eliminar el emborronamiento. Pero estos modelos tienen un problema debido a que no cuantifican la incertidumbre de las imágenes desemborronadas, lo que es un problema en ámbitos como el de la medicina, de aquí surge la motivación para este trabajo.

El desarrollo del modelo Bayes-SelfDeblur presentado en este proyecto pretende resolver el problema anterior y supone una importante contribución al problema del BID. Al incorporar un enfoque Bayesiano en el marco del Deep Image Prior (DIP), no solo se ha logrado una mejora en la calidad de las imágenes reconstruidas, sino también la capacidad de estimar la incertidumbre asociada a cada reconstrucción. Esto se ha conseguido con la inclusión de BNNs en el modelo de SelfDeblur, en concreto hemos utilizados Variational Dropout. Gracias a esto el modelo Bayes-SelfDeblur es capaz de capturar tanto la incertidumbre epistémica, relacionada con el modelo, como la incertidumbre aleatoria, asociada a los datos. Esto añade un valor significativo en aplicaciones donde la confianza en las imágenes procesadas es crítica, como en el ámbito médico o en la astronomía.

En la última parte del trabajo hemos realizado una experimentación para comparar el modelo Bayes-SelfDeblur con el modelo base de SelfDeblur. En concreto hemos creado dos variantes del modelo una con un dropout uniforme sobre todas las capas y otra con un dropout mixto que varía el dropout en función de si se trata de una capa encoder o de una capa decoder. En los experimentos hemos conseguido mejores resultados que el modelo base cuando el dropout uniforme no es muy elevado y cuando el dropout mixto tiene valores altos para la capa encoder y bajos para la decoder. Estos resultados nos confirman que los modelos basados en el marco de DIP extraen las características para reconstruir las imágenes a partir de la propia arquitectura de la red, lo que da lugar a malos resultados cuando el dropout es elevado puesto que se pierde la información de la propia red.

En comparación con el modelo SelfDeblur original, la implementación de Bayes-SelfDeblur ha mostrado mejores resultados en términos de precisión de las imágenes restauradas. Pero la

principal ventaja del nuevo modelo radica en su capacidad para proporcionar una estimación de la incertidumbre, lo cual es crucial en escenarios donde la toma de decisiones depende de la fiabilidad de la imagen obtenida.

5.2. Trabajo Futuro

A partir de este trabajo se pueden abrir otras líneas de investigación que tomen de base el modelo Bayes-SelfDeblur. A continuación hablamos de algunas de estas líneas investigación a futuro:

1. A partir de los valores que nos proporciona la incertidumbre epistémica se podría realizar un postprocesado de la imagen reconstruida en el que se crearían múltiples reconstrucciones para las zonas en las que hay una incertidumbre epistémica elevada.
2. Nuestro modelo Bayes-SelfDeblur hace uso de la misma arquitectura que DIP y SelfDeblur. Esta arquitectura es óptima para Redes Neuronales no Bayesianas pero podrían existir otras arquitecturas que se beneficiaran en mayor medida de las BNN.

Bibliografía

Las referencias se listan por orden alfabético. Aquellas referencias con más de un autor están ordenadas de acuerdo con el primer autor.

[1] P. Campisi and K. Egiazarian. *Blind Image Deconvolution: Theory and Applications*. CRC Press, 2017.

[2] Hamadi Chihaoui, Abdelhak Lemkhenter, and Paolo Favaro. Blind image restoration via fast diffusion inversion, 2024.

[3] Stefan Depeweg, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning, 2018.

[4] Yarin Gal. Uncertainty in deep learning, Sep 2016.

[5] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Appendix, 2016.

[6] Yossi Gandelsman, Assaf Shocher, and Michal Irani. "double-dip": Unsupervised image decomposition via coupled deep-image-priors, 2018.

[7] Soo Hyun Jung, Tae Bok Lee, and Yong Seok Heo. Deep feature prior guided face deblurring. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 884–893, 2022.

[8] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision?, 2017.

[9] Anat Levin, Yair Weiss, Fredo Durand, and William T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1964–1971, 2009.

[10] Peidong Liu, Joel Janai, Marc Pollefeys, Torsten Sattler, and Andreas Geiger. Self-supervised linear motion deblurring. *IEEE Robotics and Automation Letters*, 5(2):2475–2482, April 2020.

[11] Santiago López-Tapia, Javier Mateos, Rafael Molina, and Aggelos K. Katsaggelos. Learning moore-penrose based residuals for robust non-blind image deconvolution. *Digital Signal Processing*, 142:104193, 2023.

[12] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring, 2018.

[13] Christian J. Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf. Learning to deblur. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7):1439–1451, 2016.

[14] Jian Sun, Wenfei Cao, Zongben Xu, and Jean Ponce. Learning a convolutional neural network for non-uniform motion blur removal, 2015.

[15] Xin Tao, Hongyun Gao, Yi Wang, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring, 2018.

[16] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. *International Journal of Computer Vision*, 128(7):1867–1888, March 2020.

[17] Todd Veldhuizen. Measures of image quality. https://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/VELDHUIZEN/node18.html, 1998. [Accessed 01-09-2024].

[18] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.

[19] Andrew Gordon Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization, 2022.