

Recuerdos a Color: Reviviendo Momentos Familiares con Pix2Pix y DnCNN

Antonio Santiago Tepsich

Universidad de San Andrés

Buenos Aires, Argentina

atepsich@udesa.edu.ar

Máximo Gubitosi

Universidad de San Andrés

Buenos Aires, Argentina

mgubitosi@udesa.edu.ar

Abstract—Se desarrollaron y aplicaron modelos de deep learning para colorizar imágenes en blanco y negro, específicamente empleando un DnCNN para la reducción de ruido y un generador de cGAN para la colorización. El estudio demostró que es posible no solo restaurar colores de forma convincente sino también mejorar la calidad visual de imágenes históricas, superando las limitaciones de su época. A pesar de ciertas limitaciones relacionadas con la generalización en fondos variados y la calidad de las imágenes del dataset propio, los modelos mostraron una capacidad significativa para producir imágenes de alta calidad.

I. INTRODUCCION

En la intersección de los recuerdos y la innovación tecnológica, las personas siempre buscamos mantener vivos distintos momentos. Las maneras fueron evolucionando, desde el uso de lienzos a las fotografías. Sin embargo, las limitaciones tecnológicas de épocas pasadas relegaron muchos de estos recuerdos a escalas de grises, y los colores del momento original quedaron olvidados.

Con el objetivo de devolver el color a nuestras imágenes familiares, se implementaron distintas metodologías. Con el fin de abarcar el mismo problema con distintos enfoques, se trabajó tanto en la parte de los datos, como en la implementación de distintos modelos. Cada uno de los modelos fue aumentando en complejidad y eso se ve reflejado en la cantidad de parámetros que tiene cada uno. Partiendo del Autoencoder Convolucional (CAE), seguido de una U-Net Residual y llegando a lo que es un modelo basado en Redes Adversariales, específicamente, Pix2Pix [2].

Como las imágenes reales pueden estar deterioradas por el paso del tiempo o debido a su baja calidad, se optó por hacer un modelo extra de pre-procesamiento. En el mismo se busca ajustar características importantes como las dimensiones, claridad, nitidez y ruido, que servirá posteriormente para optimizar la entrada a la segunda etapa, la colorización.

A lo largo de este informe se explicará, de manera resumida, la metodología llevada a cabo en cada modelo y set de datos, para luego mostrar los resultados obtenidos y hacer un análisis del mismo. Por último, se cierra con una conclusión, siguiéndole de propuestas a posibles mejoras a futuro y una sección de Anexo donde se podrán ver en mayor profundidad las arquitecturas y los resultados obtenidos.

II. METODOLOGÍA

A. Modelos

A.1) Autoencoder Convolucional:

El modelo de Autoencoder Convolucional(CAE), es una arquitectura específica para la tarea de reducción de dimensionalidad de datos mientras se preserva información espacial, lo que los hace ideales para aplicaciones en procesamiento de imágenes. En esencia, un CAE aprende a codificar las entradas en representaciones comprimidas y luego a reconstruir la entrada desde esa representación comprimida, lo cual puede ser útil para tareas como denoising, super-resolución, y más.

En nuestro caso, la estructura específica del modelo implementado, se presenta en la tabla I.

Tipo	W/H	C In	C Out	K	S	P
Conv2D	224	32	16	(3, 3)	1	1
MaxPool2d	112	16	16	-	2	0
Conv2D	112	16	8	(3, 3)	1	1
MaxPool2d	56	8	8	-	2	0
ConvTranspose2d	112	8	16	(2, 2)	2	0
ConvTranspose2d	224	16	2	(2, 2)	2	0

TABLE I: Especificaciones de la arquitectura CAE utilizada para la reconstrucción de imágenes

La eficiencia del modelo y su capacidad para reconstruir las imágenes originales dependen de la función de costo. En nuestro caso se utilizó la pérdida de error cuadrático medio MSE, definida a continuación:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (1)$$

A.2) U-Net:

Una U-Net [3] es una arquitectura de Red Neuronal Convolutiva que fue diseñada inicialmente para tareas de segmentación biomédica de imágenes. Estas, se han vuelto populares para realizar otro tipo de tareas, como lo es la colorización de imágenes.

La U-Net se caracteriza por su estructura en forma de "U", compuesta por un *Encoder* y un *Decoder* conectados por *Skip Conexions*, para no perder información en la reconstrucción de imágenes.

En nuestro caso probamos con la arquitectura detallada en la Tabla II y la función de costo MSE, mencionada previamente en la ecuación 1.

Tipo	W/H	C In	C Out	K	S	P
Conv2D	256	1	32	(4, 4)	2	1
Conv2D	128	32	64	(4, 4)	2	1
Conv2D	64	64	128	(4, 4)	2	1
Conv2D	32	128	256	(4, 4)	2	1
T Conv2D	16	256	128	(4, 4)	2	1
Concat	32	256	256	-	-	-
T Conv2D	32	256	64	(4, 4)	2	1
Concat	64	128	128	-	-	-
T Conv2D	64	128	32	(4, 4)	2	1
Concat	128	64	64	-	-	-
T Conv2D	128	64	2	(4, 4)	2	1
Concat	256	2	3	-	-	-
Conv2D	256	3	2	(3, 3)	1	1

TABLE II: Arquitectua U-Net Residual

A.3) Pix2Pix:

El modelo Pix2Pix, es una arquitectura de Red Neuronal que transforma una imagen de entrada en una imagen de salida deseada, usando técnicas de aprendizaje supervisado junto con Redes Adversarias Condicionales (cGANs) [4]. Este modelo es especialmente efectivo para tareas donde la relación entre la entrada y la salida puede ser aprendida y modelada, como lo es en nuestro caso con la colorización de imágenes.

Esta arquitectura se basa en la implementación de dos redes neuronales distintas, donde una es una Red Generadora y otra una Red Discriminadora. Por la parte del Generador, intenta crear imágenes que parezcan reales a partir de imágenes de entrada, y mientras tanto, el Discriminador tiene la tarea de determinar si la imagen de entrada a la red es una imagen original del dataset, o si fue generada por el generador. De esta manera el modelo se convierte en adversario, donde las redes van compitiendo entre sí a lo largo del entrenamiento. Es así como la Red Generadora es capaz de aprender a crear imágenes lo suficientemente realista como para no distinguirla del dataset original.

Por parte de la Red Generadora, esta construida con un U-Net, que fue explicada en la sección II-A2, pero esta vez se usa como Red Generadora con la arquitectura detallada en el Anexo A, particularmente la Figura 13. La manera en que entrena, se ve reflejada de la siguiente manera:

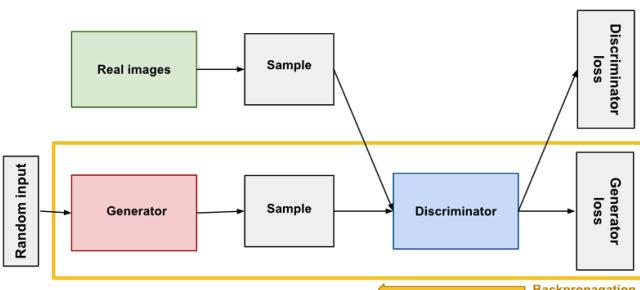


Fig. 1: Esquema de Backpropagation de la Red Generadora

Por parte de la Red Discriminadora, está implementada con un red denominada PathGAN [5], que en lugar de decidir si la imagen parece real, genera un mapa de segmentación donde separa las áreas de la imagen que parecen reales de las que no. Su arquitectura también esta detallada en el Anexo A, Figura 14 y entrena así:

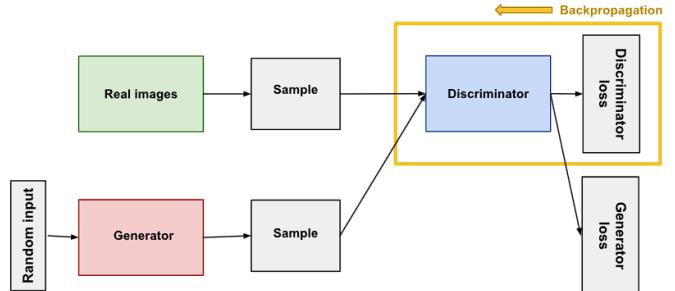


Fig. 2: Esquema de Backpropagation de la Red Discriminadora

En cuanto a la función de costo, tanto la Red Generadora como la Red Discriminadora tienen distinta Loss, pero ambas usan BinaryCrossEntropy.

En cuanto a Loss de la Red Generadora (L_G) incluye la pérdida GAN y la pérdida L1:

$$L_G = -\mathbb{E}_z \log(D(G(z))) + \lambda \cdot \mathbb{E}_z \|y - G(z)\|_1 \quad (2)$$

Por el lado de la Loss de la Red Discriminadora (L_D) combina las pérdidas de imágenes reales y generadas:

$$L_D = -[\mathbb{E}_{x \sim p_{\text{data}}(x)} \log(D(x)) + \mathbb{E}_z \log(1 - D(G(z)))] \quad (3)$$

donde $D(x)$ y $D(G(z))$ son las predicciones del discriminador para imágenes reales x y generadas $G(z)$, respectivamente, y es la imagen objetivo, y λ es el factor de balance para la pérdida L1.

A.4) Pre-Procesamiento de Imágenes:

Previo a la etapa de colorización se pretende estandarizar las imágenes, para que el modelo obtenga una entrada de características ya predeterminadas. Para lograr esto las imágenes deben tener las mismas dimensiones, espacio de color y escala de valores de los píxeles. Se modificaron las imágenes para quedar de 256×256 en escala de grises, con valores de píxeles entre 0 y 255. Con el fin de garantizar mejores resultados aún, se desarrolló también un modelo previo para reducir el ruido presente en las imágenes. Dado que las imágenes son viejas y fueron digitalizadas, se encuentran en baja resolución y algunas se encuentran algo deterioradas.

La propuesta del modelo de reducción de ruido consiste de una arquitectura DnCNN, *Denoising Convolutional Neural Network* [1], que recibe imágenes en escala de grises con ruido y devuelve la misma reconstruida, eliminando el ruido. La clave de este modelo es que en lugar de aprender directamente la imagen limpia, la red aprende la diferencia (o residuo) entre la imagen ruidosa y la limpia. Para entrenar el modelo se le

agregó ruido Gaussiano a las imágenes del dataset CelebA, obteniendo de esta manera imágenes 'limpias' y 'ruidosas'.

La función de pérdida que se utilizó es el MSE, ecuación 1 entre el ruido real y el ruido estimado. Y la arquitectura de este modelo se visualiza en la Figura 3:

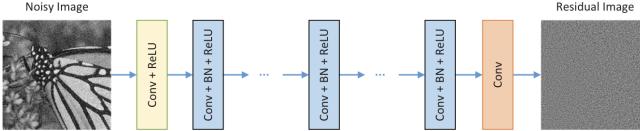


Fig. 3: Arquitectura simplificada del modelo DnCNN

B. Datasets

Encontrar los datasets para entrenar los modelos resultó ser un desafío. Esto se debe a que no era posible entrenar con un dataset propio, ya que no contábamos con suficientes imágenes y solo teníamos las imágenes en blanco y negro.

En principio buscábamos imágenes que compartieran dimensiones, el espacio de colores y que fueran variadas en cuanto a su contenido. Es por eso que el primer dataset, que consiste en una reducción del **MIRFLICKR25k** [6]. Sin embargo, al empezar a entrenar los modelos, no se lograron buenos resultados e identificamos que la varianza en el conjunto de datos era un problema. Además, considerando que la mayoría de las imágenes del dataset propio son en su mayoría retratos, buscamos un dataset mas similar, el cual fue el **CelebA Dataset** [7].

A la hora de entrenar los modelos definitivos se consideraron distintos subconjuntos del dataset CelebA. Para el modelo de denoising se tomaron primero los 500 elementos iniciales del dataset para definir los hiperparámetros empleando la estrategia de cross validation con 5 folds. Una vez definidos los hiperparámetros se entrenó el modelo final con 1000 imágenes de test y 300 de evaluación. De forma similar, para el modelo colonizador se emplearon 800 imágenes, y dada la escala del modelo se probaron pocas combinaciones de hiperparámetros usando un 80% de entrenamiento y 10% y 10% para testeо y validación. Basados en los hiperparametros de Pix2Pix, llegamos a los valores óptimos, y se entrenó el modelo principal con 2500 muestras, manteniendo las mismas proporciones.

B.1) Datasets Benchmarks:

Dataset propio: Este conjunto tiene archivos de diversos tamaños y resoluciones que corresponden a imágenes antiguas en blanco y negro que fueron digitalizadas de distintas maneras. Considerando esto, las imágenes deben procesarse para estar todas en las mismas condiciones.

MIRFLICKR25k Dataset reducido: Consiste de 25.000 imágenes en espacio de color LAB, que fueron seleccionadas para realizar la tarea de colorización. Las imágenes son variadas en estilo, están en la misma escala y tienen iguales dimensiones.

CelebA Dataset: Consiste de 202.599 imágenes de las caras de 10.177 celebridades. Los archivos se encuentran en formato

jpg, en el espacio RGB, además están todas en la misma escala y tienen dimensiones equivalentes.

B.2) Data Augmentation:

Con el fin de que el modelo tenga un set de imágenes lo suficientemente diferentes a las imágenes originales como para que el modelo no haga overfitting pero sin hacer que las imágenes originales sean inentendibles, se utilizaron una serie de algoritmos. Estos son Resize por medio de interpolacion, Cropping y Random Jittering.

B.3) LAB* vs RGB:

Un aspecto clave en el procesamiento de imágenes es el espacio de color. Al convertir imágenes de blanco y negro a color, se pasa de un espacio de un canal a uno de tres. La diferencia radica en el esfuerzo requerido por el modelo: en el espacio LAB, el canal *L* corresponde a la escala de grises, por lo que el modelo solo predice los otros dos canales. En cambio, en el espacio RGB, deben predecirse los tres canales.

C. Evaluación

Evaluar la calidad de las imágenes generadas es difícil de cuantificar debido a la naturaleza subjetiva de la percepción visual. A pesar de esto, nos basamos en dos técnicas para medir cuán buenos son los modelos, que se explicaran a continuación.

C.1) Score Humano:

El score manual aprovecha la institución de los observadores, lo que lo hace un punto de partida para determinar que tan buenas son las imágenes generadas. Aunque tiene la desventaja de ser subjetivo y potencialmente inconsistente, aporta una medida de referencia para probar distintos modelos.

C.2) Frechet Inception Distance (FID):

Esta técnica mide la similitud entre dos distribuciones de imágenes, una generada por nuestro modelo y la otra es la imagen real. Se calcula utilizando la distancia de Frechet entre las capas de características del modelo "Inception v3" activadas por las imágenes reales y generadas. Se calcula de la siguiente manera:

$$FID = \|\mu_{z_r} - \mu_{z_g}\|^2 + Tr(\Sigma_{z_r} + \Sigma_{z_g} - 2(\Sigma_{z_r} \Sigma_{z_g})^{\frac{1}{2}}) \quad (4)$$

donde z_r representan las imágenes reales y z_g las imágenes generadas. Un FID bajo indica que las dos distribuciones son similares, reflejando imágenes generadas de alta calidad yrealismo.

III. RESULTADOS Y ANÁLISIS

A. Pre-Procesamiento de Imágenes:

Una vez que las imágenes se encuentran en las dimensiones, espacio de color y escala de color correctas el siguiente paso es crear el dataset de imágenes 'ruidosas'. Se tomó el dataset CelebA y se le aplicó ruido gaussiano con un factor de escala seleccionado visualmente de 7,65, $(0,03 \times 255)$, acorde a lo observado en la Figura 4.

El siguiente paso fue el entrenamiento del modelo de denoising. En esta etapa, primero se desarrollaron dos variantes

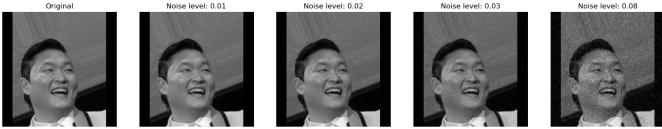


Fig. 4: Comparación entre aplicar distintos niveles de intensidad de ruido

de Autoencoders Variacionales, las cuales fueron descartados por resultados inaceptables y se terminó optando por la DnCNN. Tras entrenar este modelo por 200 épocas sobre un subconjunto de 1000 y 300 imágenes para entrenar y evaluar respectivamente, se obtuvieron los resultados presentados en la Figura 5 y el error evolucionó como se ilustra en la figura 6.



Fig. 5: Comparación de imagen con ruido, sin ruido y la predicción del modelo (*imagen limpia*)

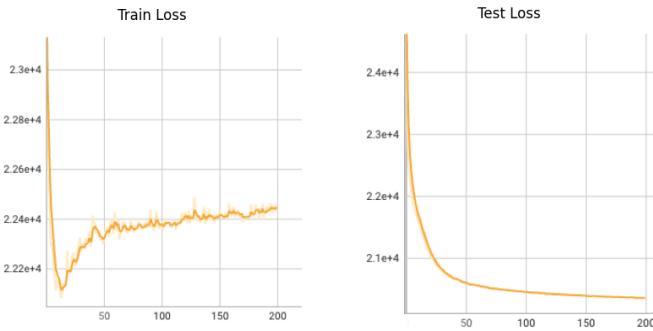


Fig. 6: Evolución de la Loss de entrenamiento y validación respecto de la época de la DnCNN

En la Figura 6 se puede observar como el Loss en entrenamiento cae al principio y luego aumenta lentamente. Por otro lado, en validación el Loss presenta una curva monotónicamente decreciente. Considerando ambas curvas, resulta conveniente tomar el modelo tras las 200 épocas, donde el Test Loss se encuentra en un mínimo. El hecho de que el Train Loss continúa aumentando levemente se debe a la presencia de técnicas de regularización en el entrenamiento del modelo.

B. Colorizador:

Por poder llegar a los resultados obtenidos, los mejores outputs se obtuvieron con un Learning Rate muy chico, un dataset de 2500 imágenes y con el optimizador Adam. Los

siguientes resultados son de imágenes que los modelos no vieron en ningún momento.



Fig. 7: Comparación entre Input, Real e Imagen Generada por Pix2Pix

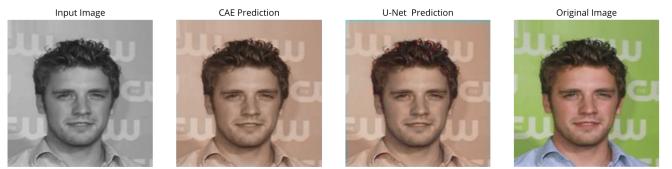


Fig. 8: Comparación entre Input, Real e imágenes generadas por el CAE y la U-Net

En la Figura 8 se puede observar que las colorizaciones de estos modelos son mucho mas primitivas que las del modelo adversarial. El CAE parecería simplemente aplicar un filtro a la imagen y la U-Net presenta colores difusos.

Y las distintas Losses de los modelos son las siguientes:

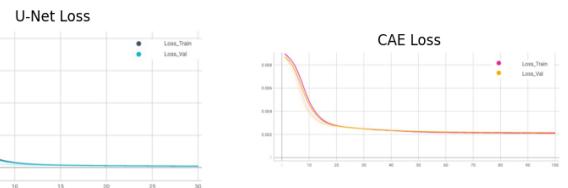


Fig. 9: Evolución de la Loss de entrenamiento y validación respecto de la época de Pix2Pix

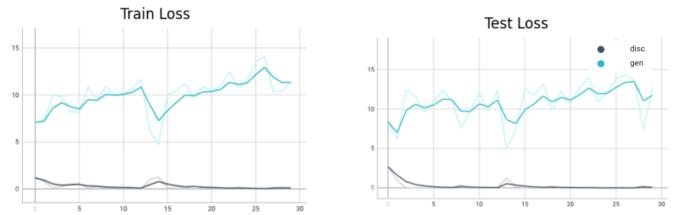


Fig. 10: Evolución de la Loss de entrenamiento y validación respecto de la época de CAE y U-Net

Por otro lado, si observamos la Figura 7, utilizando la metodología de evaluación II-C1, el ultimo modelo es el que obtiene mejores resultados, haciéndolo difícil de identificar con la imagen verdadera.

En cuanto a metodología FID II-C2, por parte de la CAE obtuvimos 0.3091, por parte de la U-Net 0.2342 y por ultimo, el modelo Pix2Pix con los hiperparametros previamente mencionados llegamos a obtener un score de 0.0163. Esto nos vuelve a validar nuestro criterio previo ya que al ser un valor bajo, indica que tanto las imagenes reales como las generadas tienen distribuciones similares, reflejando imágenes generadas de alta calidad y realismo.

A modo extra, se pueden ver mas resultados de las imágenes del set de Test del mejor modelo en Anexo B. Y por ultimo, probamos con un dataset el cual usa colores muy fuertes con contrastes muy oscuros. Es por eso que en el Anexo C, van a poder ver los increíbles resultados con este modelo en ese benchmark particular.

C. Pruebas en Conjunto:

El fin de este trabajo se centra en esta ultima sección, en la cual mostramos los resultados finales pero ahora con las imágenes de nuestras familias. Estas son imágenes que por el tiempo guardadas o por la tecnología del momento, son imágenes que estan un poco dañadas y en distintos tonos de grises. Es por eso que lo primero que hacemos es realizar Denoising y resizing explicado en la sección II-A4, obteniendo los siguientes resultados: Acá podemos ver como el daño que

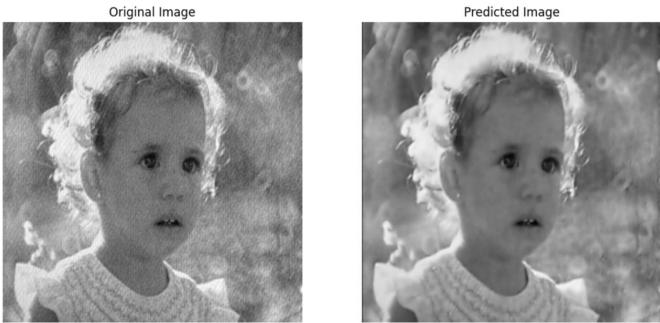


Fig. 11: Resultado de Imagen Real Recortada y Restaurada

hay en el centro de la imagen original prácticamente se elimina y que aquellas irregularidades por la resolución de la fotografía se ve suavizada. Esto sin dudas ayudara al modelo generativo que le dará color a la imagen ya que podemos intuir que tendrá un grado mas de similitud al dataset con el que se lo entreno.

Por ultimo, utilizando el mejor modelo obtenido, usamos únicamente la Red Generadora de la sección II-A3 y obtenemos los siguientes resultados:

En el Anexo D podrán ver más resultados reales.

IV. CONCLUSIONES

Este trabajo demostró cómo la aplicación de técnicas avanzadas de aprendizaje automático y procesamiento de imágenes puede servir como un puente entre la preservación de recuerdos históricos y la innovación tecnológica. Logramos el objetivo de colorear imágenes en blanco y negro empleando una combinación de modelos de deep learning: una DnCNN para reducir el ruido en las imágenes del dataset propio y el generador de la cGAN.

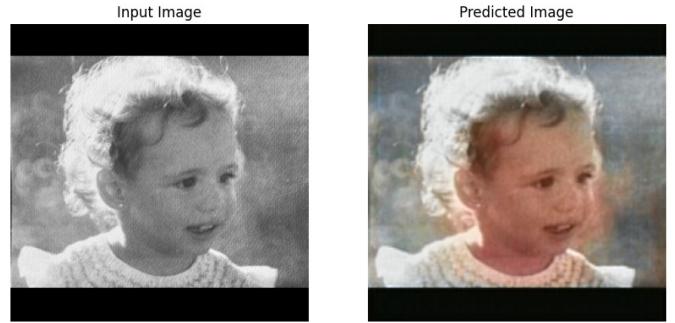


Fig. 12: Resultado de Imagen Real Recortada y Restaurada

Nuestros resultados muestran que es factible no solo restaurar colores de forma convincente, sino también mejorar la calidad de imágenes históricas. La integración del DnCNN para el denoising antes de la colorización mejoró la calidad de las imágenes. Aunque hay margen para optimizar el modelo de reducción de ruido, el foco principal de este estudio fue el modelo de colorización. Futuras mejoras a este modelo podrían incluir una búsqueda más exhaustiva de hiperparámetros, mayor duración del entrenamiento, un incremento en el número de imágenes y la implementación de técnicas como el early stopping.

Los resultados del modelo de colorización muestran su capacidad para producir imágenes de alta calidad, especialmente cuando estas son bien definidas y similares a las usadas en el entrenamiento, como retratos. No obstante, el modelo enfrenta dificultades para reproducir los colores de los fondos que difieren notablemente de los del entrenamiento. Aunque el desempeño es excelente tanto en el conjunto de entrenamiento como en pruebas, reflejando la homogeneidad del dataset CelebA, las limitaciones del cGAN empleado se manifiestan en algunas imágenes del dataset propio que presentan elementos distintos o de baja resolución.

En conclusión, se logró colorizar satisfactoriamente las imágenes del dataset propio, obteniendo algunos ejemplos notables. Sin embargo, la calidad de la colorización fluctúa considerablemente dependiendo del estilo de cada imagen, como se mencionó previamente. Entre las principales limitaciones del sistema desarrollado, destaca la **similitud entre las imágenes** de entrenamiento. Existe un balance necesario entre la variedad y la cantidad de imágenes utilizadas y los recursos computacionales requeridos. El uso del dataset CelebA nos permitió obtener resultados sobresalientes en un tipo específico de imágenes, pero al extender la aplicación del modelo a imágenes más variadas de nuestro dataset propio, observamos una disminución notable en el rendimiento. Por otro lado, la **calidad de las imágenes** en el dataset propio era comparativamente inferior a la del dataset de entrenamiento, lo cual limitó la capacidad del modelo para segmentar adecuadamente las muestras. Aunque el modelo de denoising implementado ayudó a mitigar el impacto del ruido, es importante destacar que su función se centra en la reducción del ruido y no en el aumento de la resolución de las imágenes.

V. TRABAJO A FUTURO

Finalmente, este proyecto se basó en una extensa investigación, implementación, pruebas y errores. Es por eso que a lo largo de este trabajo fuimos tomando nota de las mejoras que se podrían hacer a futuro.

- Utilizar LAB* en el modelo Pix2Pix para reducir tiempo de entrenamiento.
- Remplazar la arquitectura cGAN por un que reduzca errores de convergencia como la CycleGAN o WGAN.
- Mejorar las generaciones del background de las imágenes entrenando adicionalmente con el dataset específico de Paisajes.
- Utilizar un modelo Pre-entrenado con ImageNet (ej: ResNet16) y entrenarlo como generador.
- Utilizar además de la DnCNN para denoising, un modelo de aumento de resolución, como una SRGAN, ESRGAN o RCAN.

REFERENCES

- [1] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising," *arXiv preprint arXiv:1608.03981*, August 2016.
- [2] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," *Berkeley AI Research (BAIR) Laboratory, UC Berkeley*, November 2018.
- [3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," *Computer Science Department and BIOSS Centre for Biological Signalling Studies, University of Freiburg, Germany*, May 2015.
- [4] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," *Departamento d'informatic et de recherche opérationnelle, Université de Montréal, Montreal, QC H3C 3J7, Canada and Flickr/Yahoo Inc., San Francisco, CA 94103, USA*, November 2014.
- [5] M. Assens, X. Giro-i-Nieto, K. McGuinness, and N. E. O'Connor, "PathGAN: Visual Scanpath Prediction with Generative Adversarial Networks," *Dublin City University and Universitat Politècnica de Catalunya*, September 2018.
- [6] S. Kumar, "Image Colorization Dataset," *Kaggle*, Available: <https://www.kaggle.com/datasets/shravankumar9892/image-colorization/data>.
- [7] J. Li, "CelebA Dataset," *Kaggle*, Available: <https://www.kaggle.com/datasets/jessicali9530/celeba-dataset/data>.

APPENDIX A

ARQUITECTURAS CGAN:

Red Generadora:

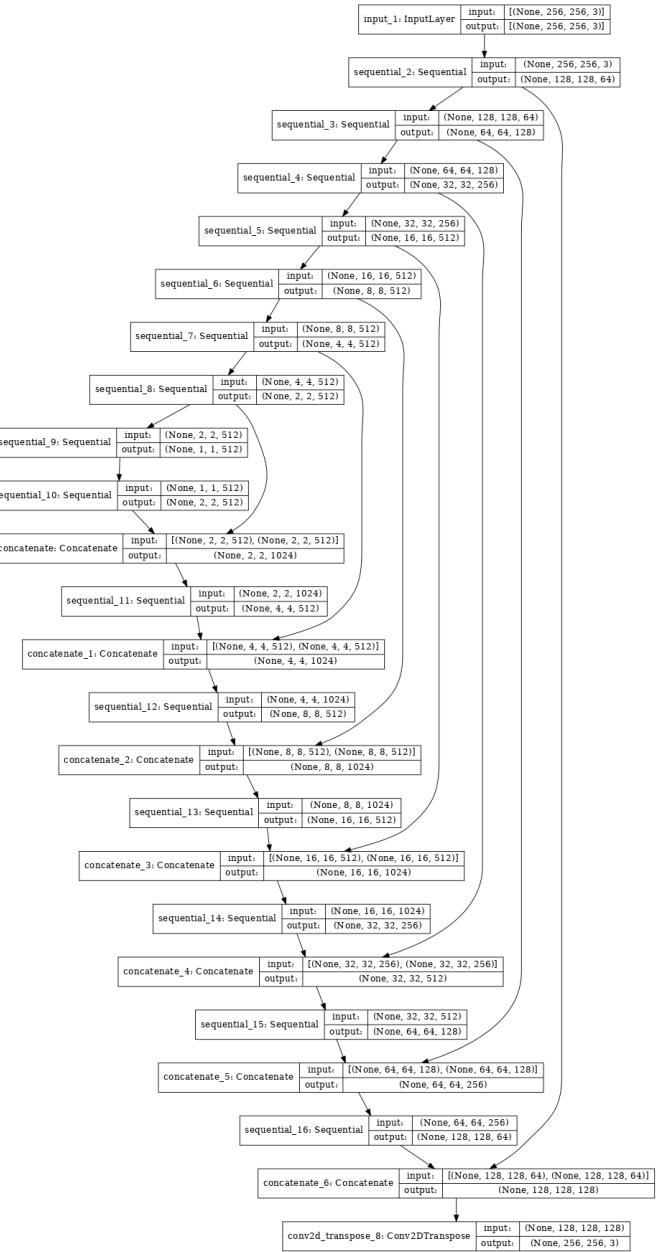


Fig. 13: Arquitectura de la Red Generativa implementada

Red Discriminadora:

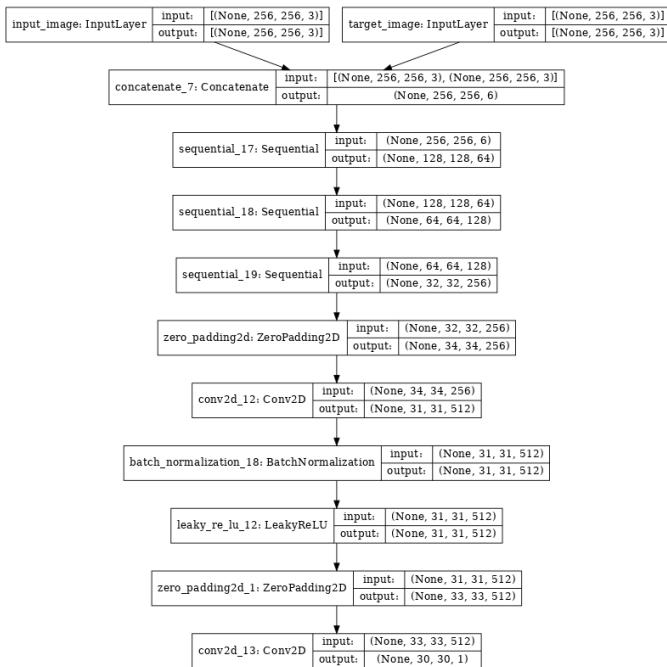


Fig. 14: Arquitectura de la Red Discriminadora implementada

APPENDIX B TEST DE PIX2PIX:

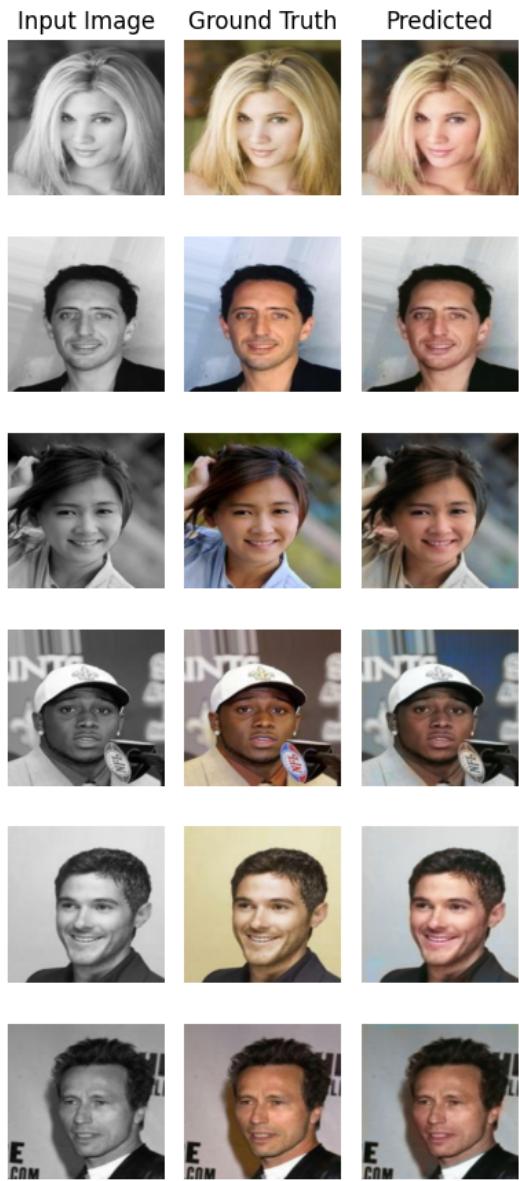


Fig. 15: Algunos resultados de las generaciones del set de Test

APPENDIX C
PICTURES BENCHMARK:

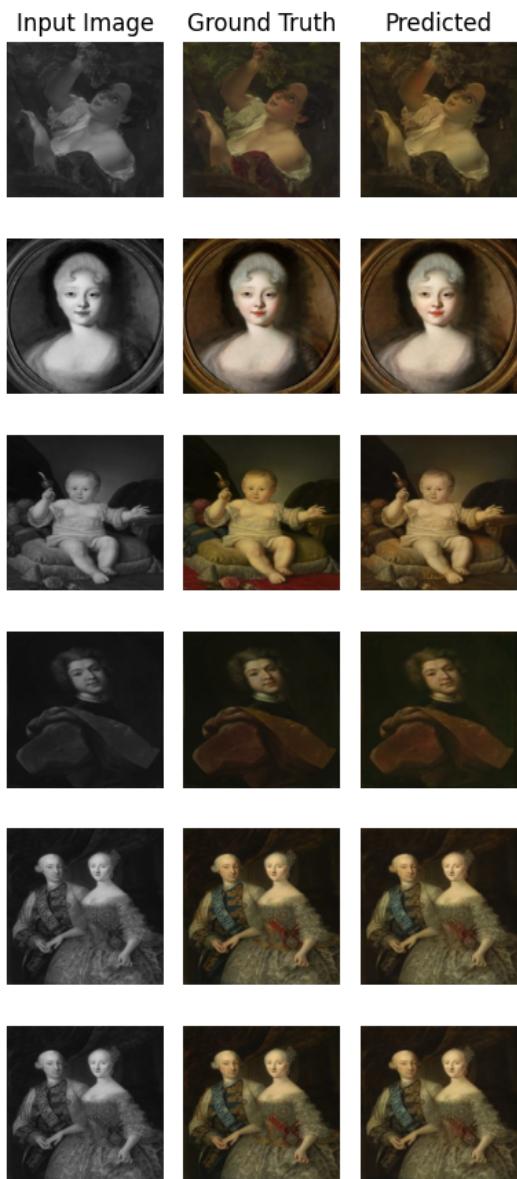


Fig. 16: Algunos resultados con generador Pix2Pix en benchmark de cuadros

APPENDIX D
IMÁGENES PROPIAS:

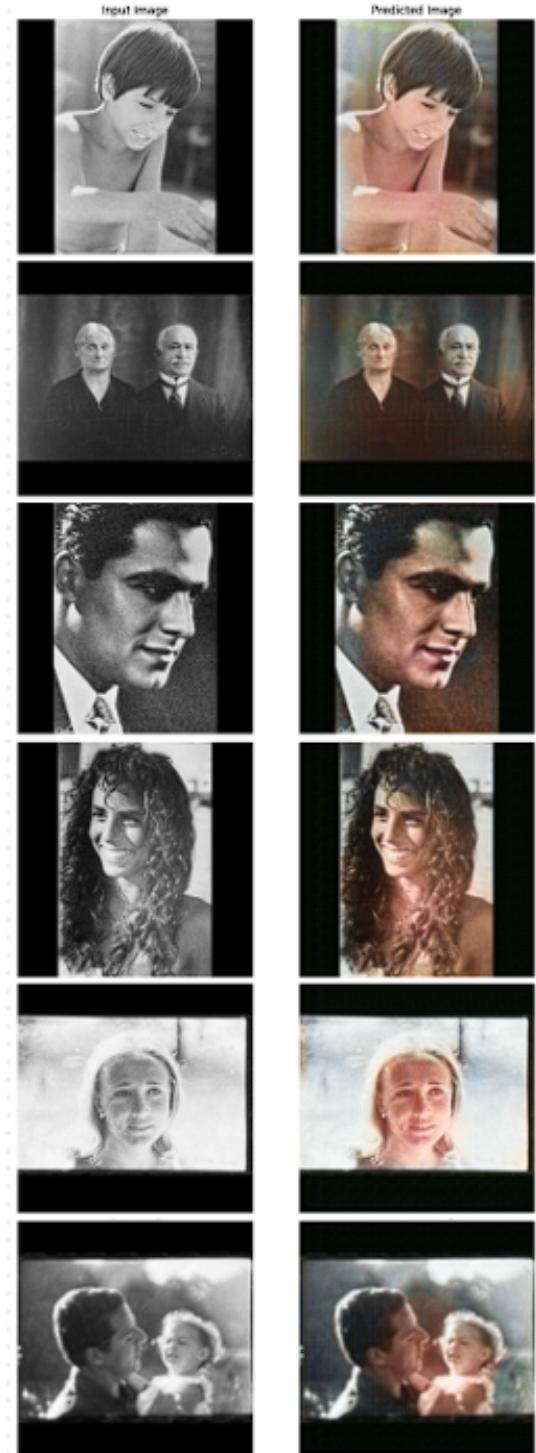


Fig. 17: Algunos resultados con generador Pix2Pix en imágenes propias