

6

MI MIRADA CON LA I. A. TRATAMIENTO DE IMÁGENES



En este capítulo usaremos la Inteligencia Artificial, una Red Neuronal Convolutacional, para crear nuevas imágenes.

El proceso recuerda lejanamente al que realiza un pintor para crear sus cuadros:

El cerebro del pintor percibe la realidad a través del reflejo de la luz en ella y de la interpretación que su cerebro da a los impulsos recibidos a través de la vista.

Con esa idea, el pintor transforma la realidad e interpreta el mundo a su estilo, con su sensibilidad. Como reacción a las formas que ve, a la luz que percibe genera su idea, una imagen mental.

Finalmente, con su habilidad y técnica pictórica, representa esa idea, esa imagen mental, en una imagen física contenida en los pigmentos de colores, que queda impresa en el papel o tela donde realiza su pintura.

De una forma que recuerda vagamente este mismo proceso, nuestro ordenador percibe una imagen fotográfica como una matriz de píxeles que identifican en cada uno de ellos un color, es la imagen original, una fotografía de la realidad. La Inteligencia Artificial, en una segunda fase, es capaz de interpretar y modificar esta matriz de píxeles, reinterpretándolos con un estilo y características que ha generado, a través de un proceso iterativo, sobre muchas imágenes, que comparan realidad con dibujo o pintura. Al final, el ordenador es capaz de generar una nueva imagen obtenida aplicando ese código de estilo a la imagen original.

En nuestro caso, yo he utilizado la imagen original que encabeza este capítulo.

Además, he utilizado en mi ordenador un programa en Python de Machine Learning que, en un primer momento, carga en su CPU la imagen fotográfica y la prepara para el proceso de creación.

En un segundo script se carga el código con un módulo de TensorFlow Hub. En nuestro caso el módulo es Arbitrary Image Stylization, lo veremos con detalle y en el anexo hay una explicación mayor, que se aplica a las variables de estilo cargadas, las imágenes de cuadros pintados o dibujados.

Con ello, se genera la transferencia de estilo y produce unas nuevas imágenes creadas aplicando al contenido original, la fotografía original, un traspaso de los rasgos y características de las pinturas utilizadas como estilos. Veamos los resultados:



Las imágenes de arriba son los estilos. El modelo de Red Neuronal con el algoritmo indicado generó las imágenes de abajo, aplicando los rasgos y características de los diferentes estilos a la imagen original. Estas imágenes son nuevas. Son creaciones del ordenador.

Imagen Final: 1



Imagen Final: 2



Imagen Final: 3



Imagen Final: 4



Imagen Final: 5



Imagen Final: 6



En una tercera fase, volvemos aplicar algoritmos para la transformación de estas imágenes creadas. En este ejemplo usamos la primera de la izquierda para estilizar y transformar las imágenes:



En este caso no hemos modificado el estilo, sino que hemos modificado características de la imagen, para suavizar y simplificar, dando el aspecto de una verdadera acuarela.

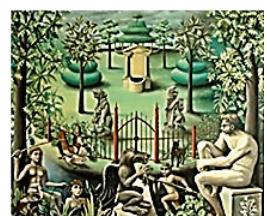
También, podemos transformar la imagen con el código, afectando al color y a los bordes, dando como resultado imágenes que parecen dibujos a tinta o a lápiz.



Por tanto, vemos la capacidad de la Inteligencia Artificial de crear nuevas imágenes a partir de una fotografía original, transformándola, según un estilo y características buscado, para crear nuevas imágenes con la apariencia de pinturas a la acuarela o dibujos de tinta o lápiz.

El ejemplo anterior lo realizamos con una fotografía personal, dando lugar al retrato o pintura a la acuarela de dicha persona, o a su esbozo o dibujo.

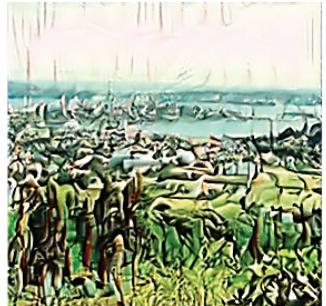
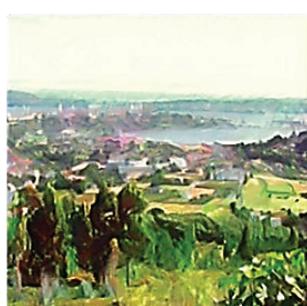
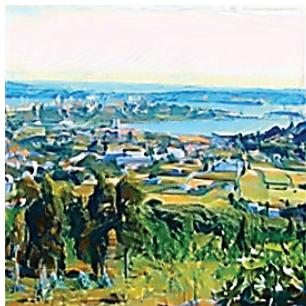
Naturalmente, también podemos aplicarlo a un paisaje, a la fotografía de un espacio, de una vista, como vemos ahora.



He utilizado para este ejemplo, una fotografía de la vista de un paisaje de la ría de Ferrol y la de una fotografía que tomé de un campo siendo recolectado por un agricultor.

Los estilos que utilizo son las fotografías de un cuadro de un paisaje de mi tío Antonio Tenreiro, de un cuadro con una vista mía y de un cuadro de mi primo Jaime D. Tenreiro.

Aplicando el anterior algoritmo de la red neuronal, que usamos para los retratos obtenemos las siguientes imágenes:



Vemos como, efectivamente, se pueden reconocer en cada nueva imagen el estilo que la generó.

También, podemos comprobar cómo la Inteligencia Artificial, más concretamente este algoritmo, genera imágenes raras, inserta rarezas como árboles que parecen tener piernas. Pero en general, el resultado es bueno. En todo caso, como ya veremos en los ejemplos, los podemos eliminar con técnicas tradicionales.

Naturalmente se podría mejorar, pero son imágenes nuevas, creadas por Inteligencia Artificial y sugerentes, capaces de transmitir algo.

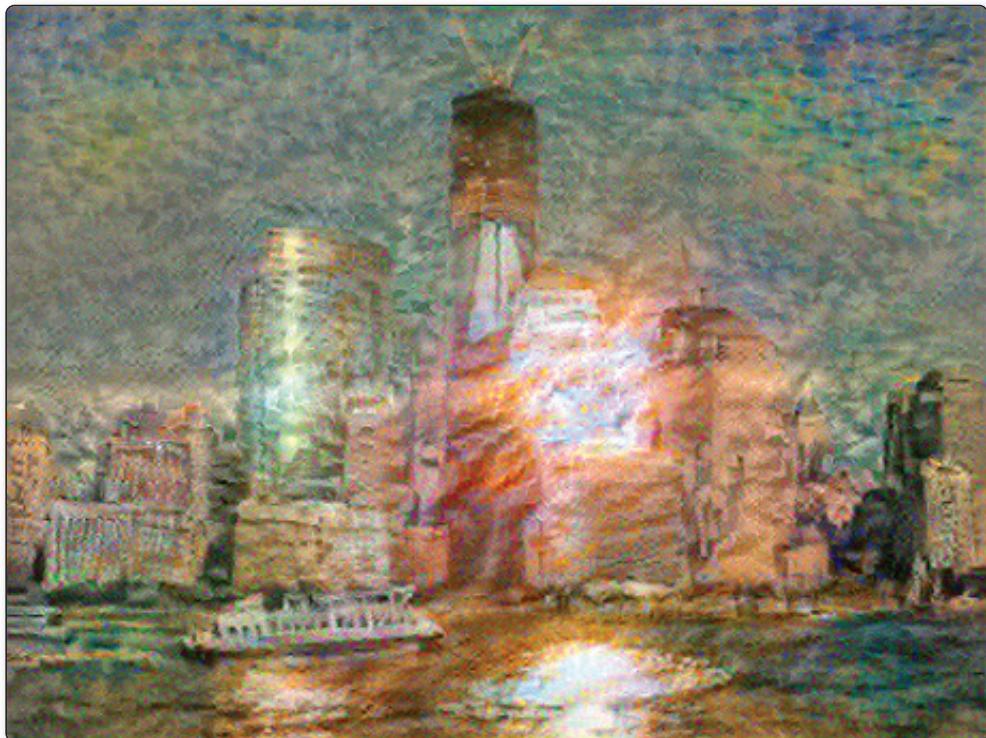
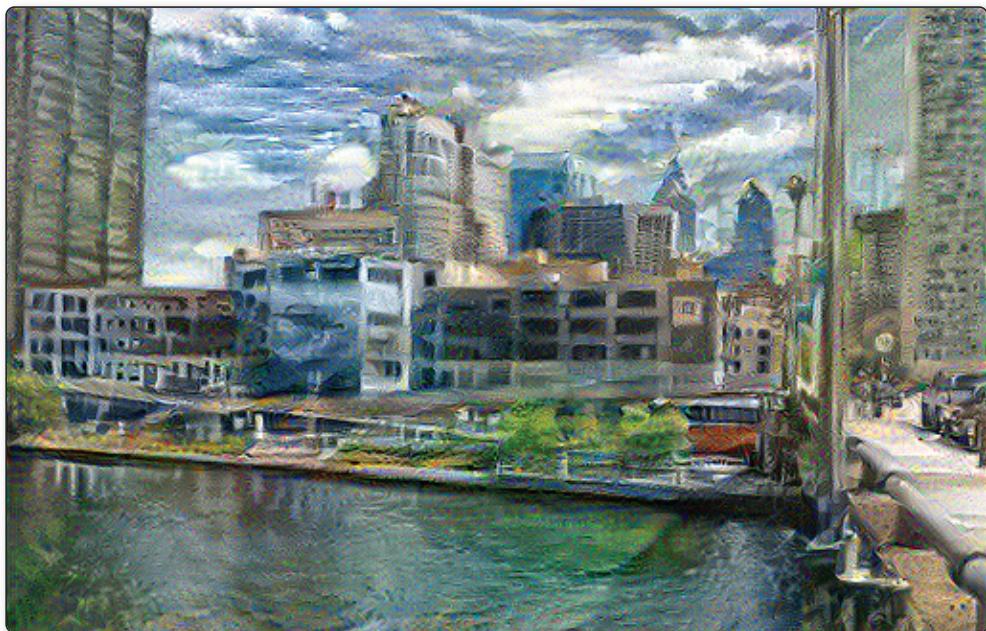
Veamos para finalizar este capítulo, algunas imágenes creadas por la Red Neuronal, en base a fotografías mías, pero a un tamaño que podamos fijarnos más en los detalles y podamos comprobar el buen resultado del trabajo. Primero paisajes, al final retratos:

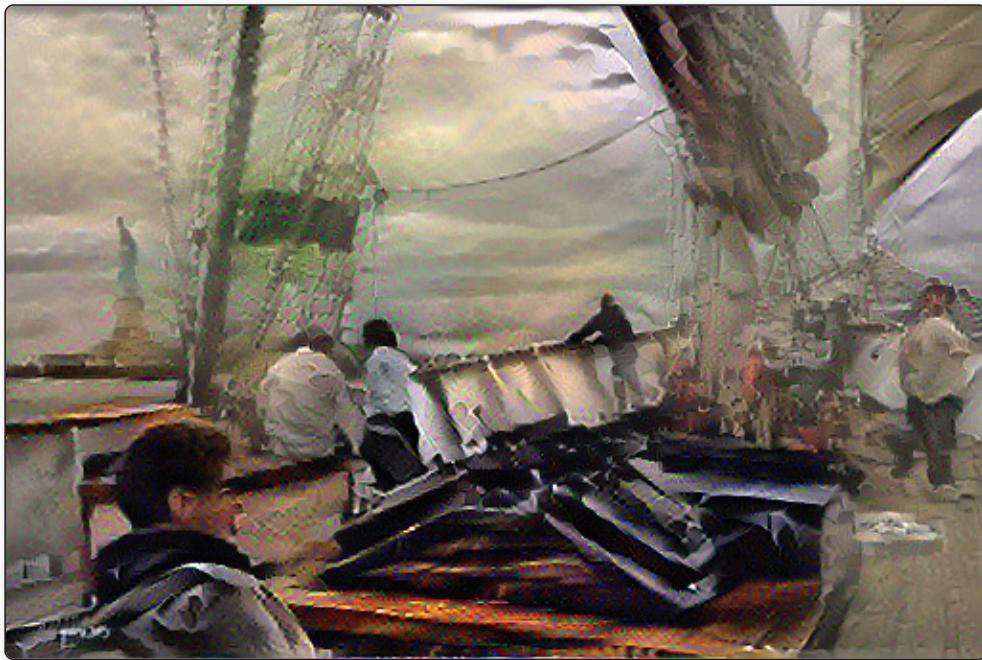


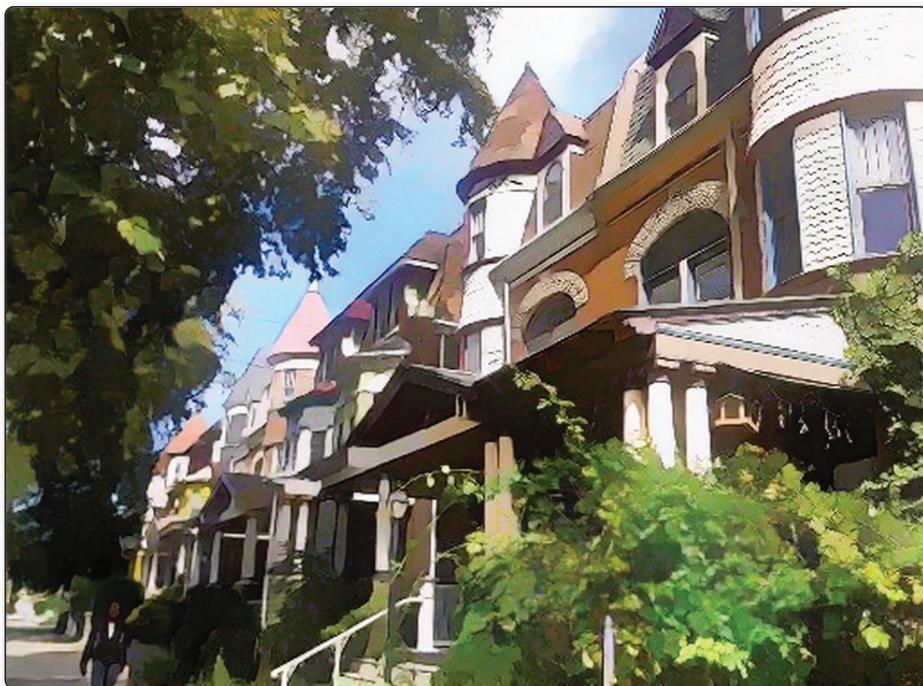
6.1 PAISAJES CREADOS CON INTELIGENCIA ARTIFICIAL





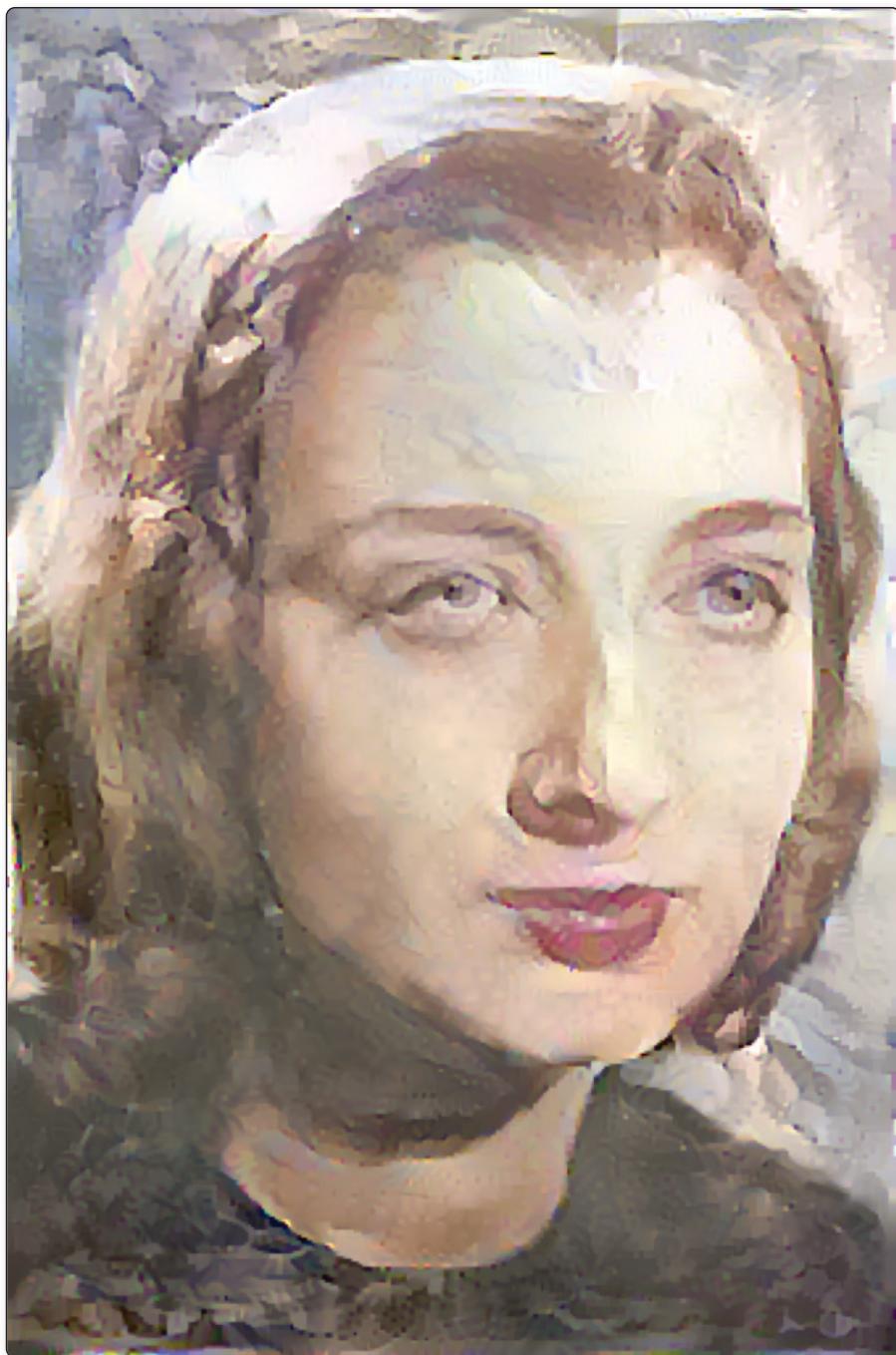


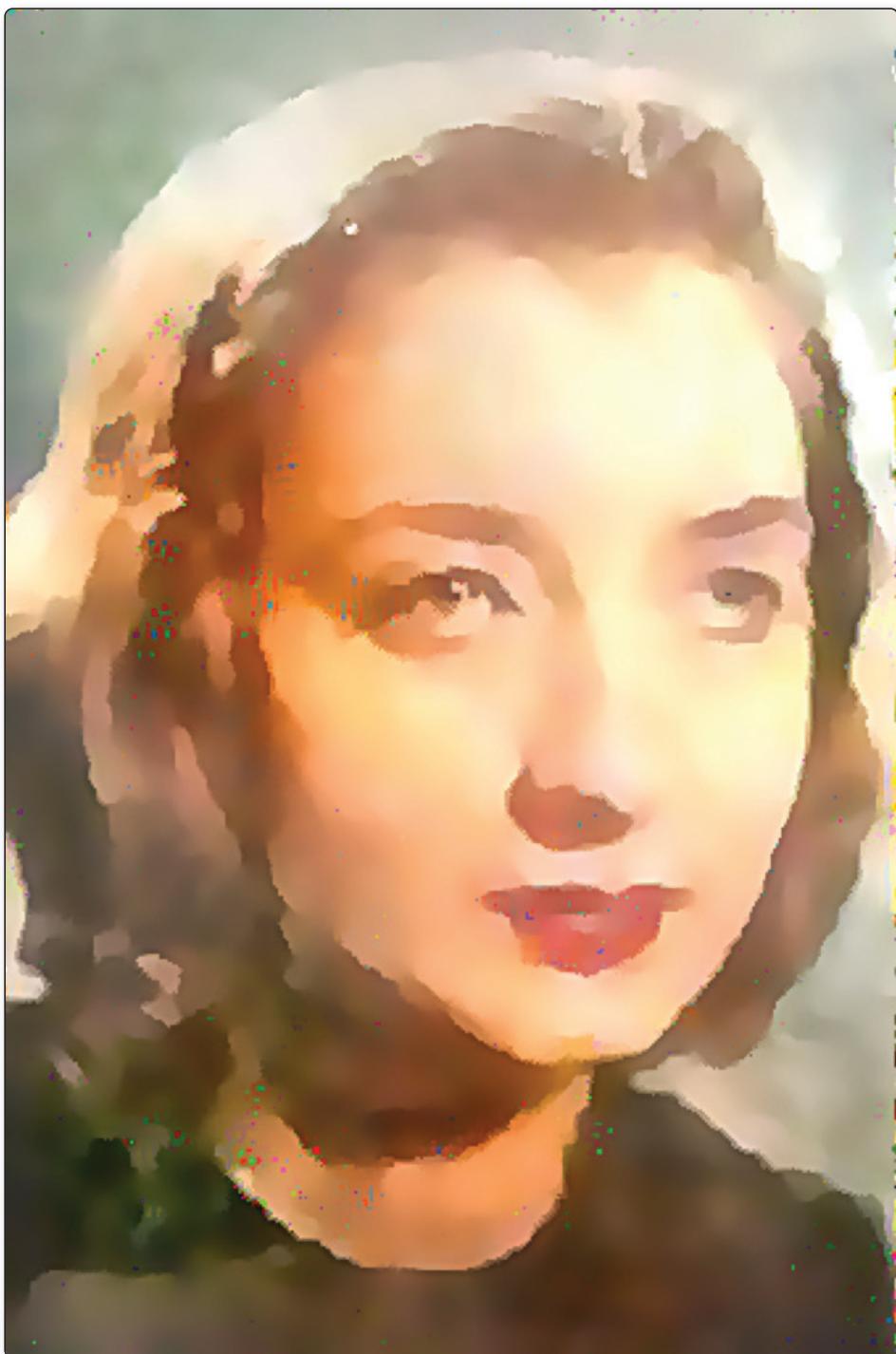


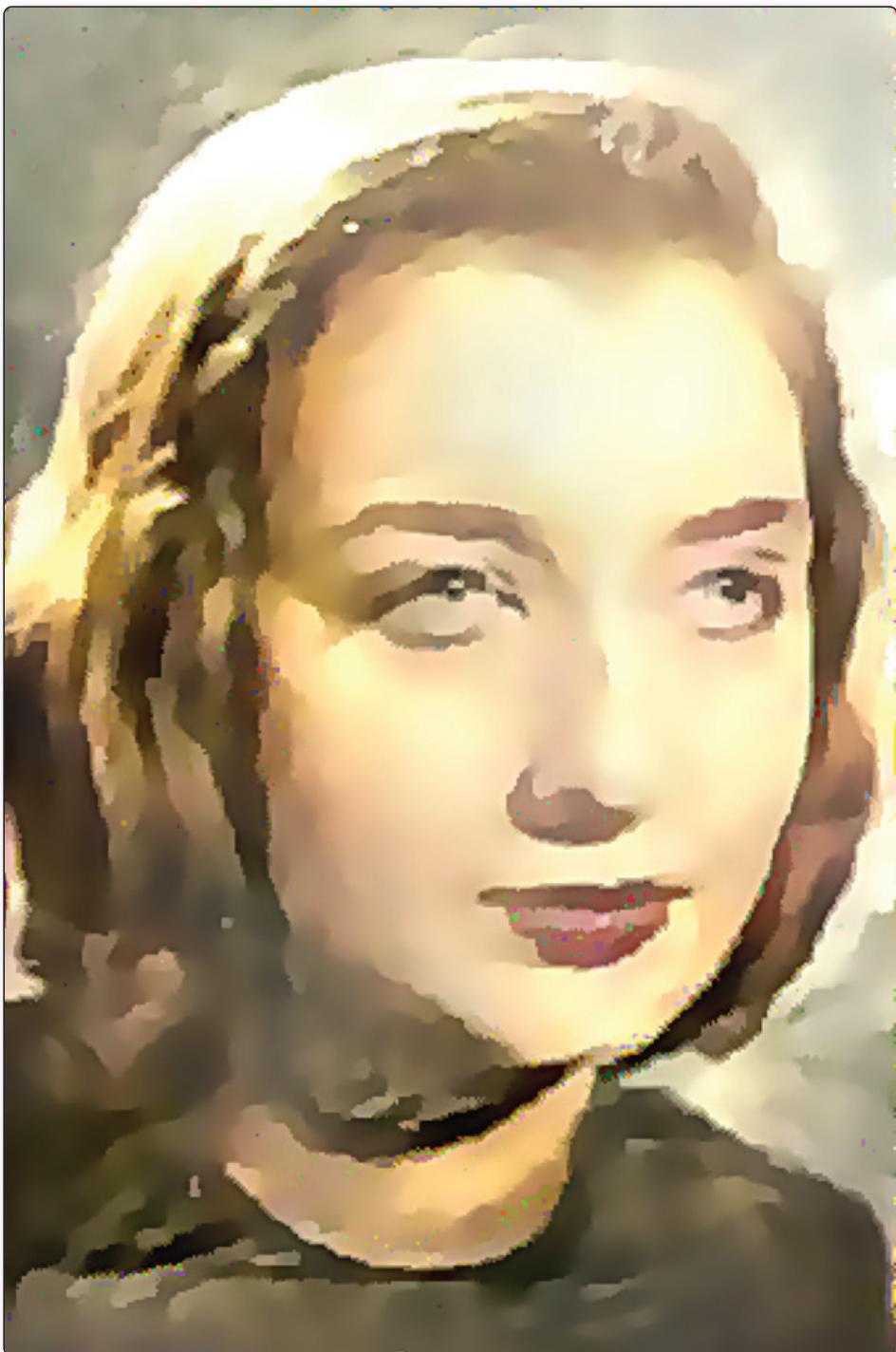




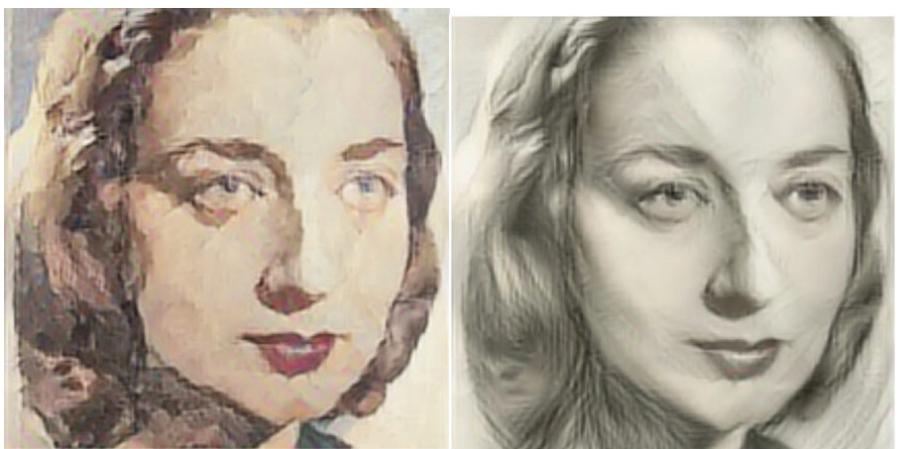
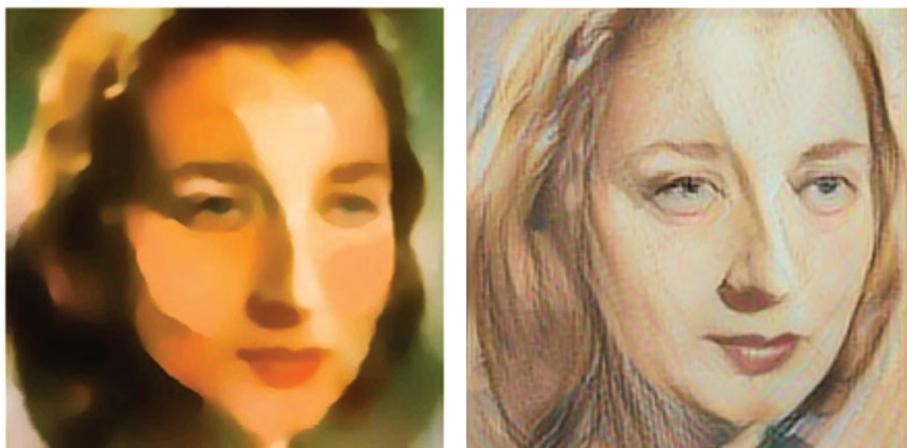
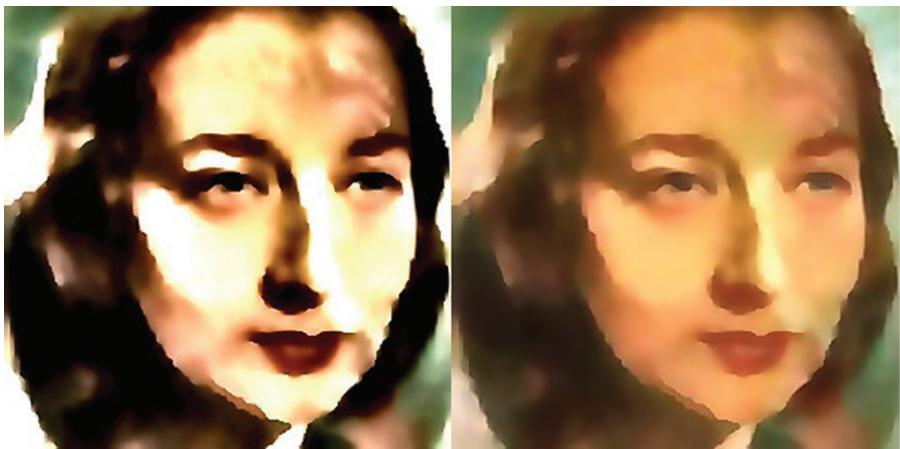
6.2 RETRATOS CREADOS CON INTELIGENCIA ARTIFICIAL

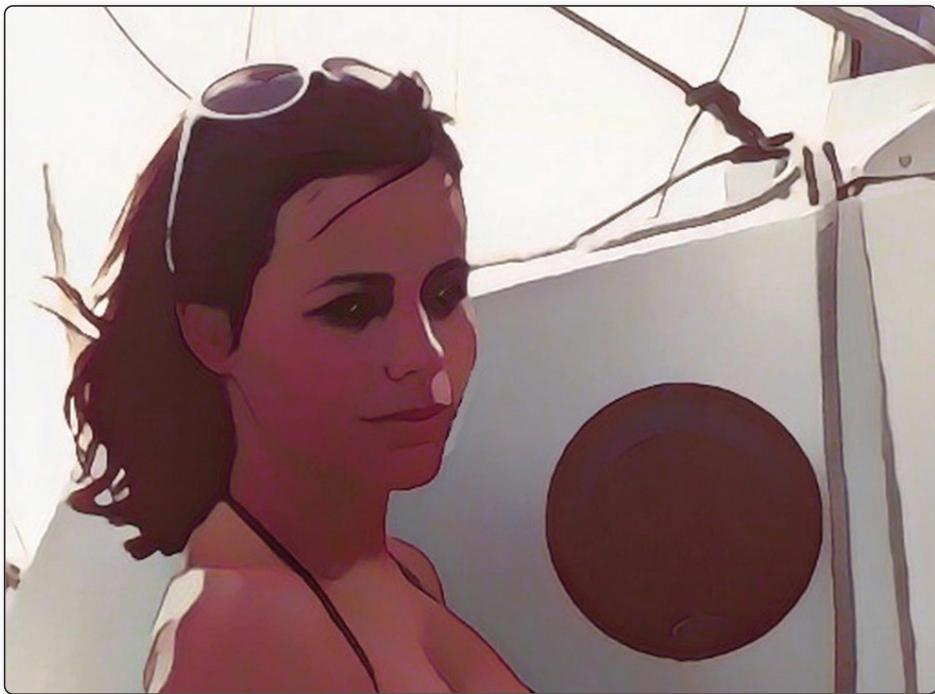












En los siguientes apartados incluyo una explicación de los códigos que he utilizado para realizar el tratamiento de las imágenes que hemos visto. Dichos códigos están a disposición del lector/a en el repositorio:

<https://github.com/rtenreiro/IAmPintor>

The screenshot shows the GitHub repository page for 'IAmPintor' owned by 'rtenreiro'. The 'Code' tab is selected. Below it, there's a list of files with their commit history:

File	Commit Message	Date
LICENSE	Initial commit	2 days ago
README.md	Create README.md	2 days ago
RT-Img2Pict.ipynb	Add files via upload	2 days ago
RT-M-PyV-style_transfer.ipynb	Add files via upload	2 days ago
RT-img2WaCo1.ipynb	Add files via upload	2 days ago
RT-img2WaCo2.ipynb	Add files via upload	2 days ago

El código está escrito en Python y se presenta en Jupyter Notebook. Se puede ejecutar, bajarlo y modificarlo.

Ahora veamos la explicación con detalle de los tres principales códigos, el primero realiza una transformación de la imagen a través de una transferencia de estilo sencilla, el segundo modifica las características de la imagen sin afectar al estilo. Y por último, una transferencia de estilo compleja que además logra muy buenos resultados.

Veámoslos:

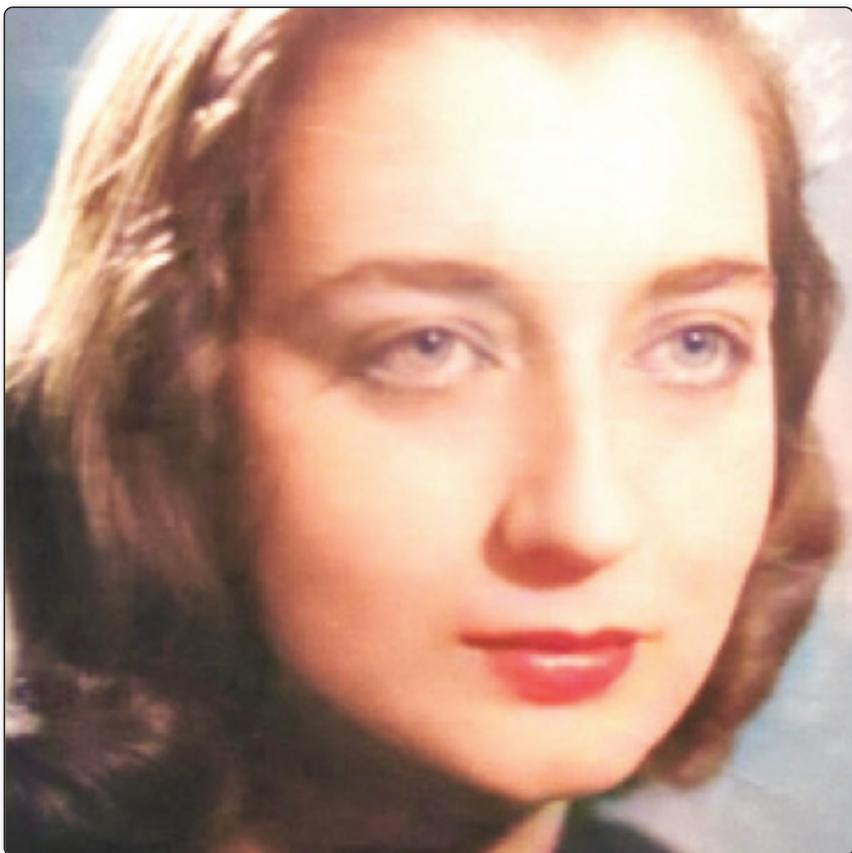
6.3 MÓDULO IMAGEN FOTOGRÁFICA INTERPRETADA O DIBUJADA (PASO 1)

El código realiza varias funciones relacionadas con la carga y preparación de imágenes, así como la visualización de estas.

Importamos las bibliotecas necesarias para el procesamiento de imágenes, incluyendo TensorFlow y TensorFlow Hub para cargar y procesar modelos preentrenados, y Matplotlib para visualización de imágenes.

Definición de funciones para cargar y visualizar imágenes:

- ▶ `crop(image)`: esta función recorta la imagen cuadrada centrada a partir de una imagen rectangular. Se utiliza para garantizar que las imágenes tengan las mismas dimensiones antes de ser procesadas.
- ▶ `load_image(source)`: esta función carga y preprocesa la imagen, la convierte en un tensor de tipo float32, la redimensiona y la normaliza.
- ▶ `image_plot(images,title,grid)`: utiliza esta función para mostrar la imagen de contenido y las imágenes de estilo con sus respectivos títulos.



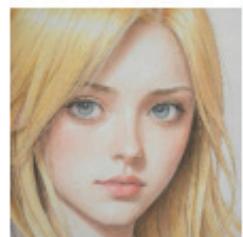
```
path_c='E://NEURONAL//EBOOK//EBOOK//IMAGENES//HITA.jpg'
```

Variable que contiene la carpeta y nombre de la imagen a modificar,

```
content=load_image() #Carga la imagen aplicando la función descrita.
```

i_style=load_image(i). A través de un bucle y la función load_image, carga las imágenes que van a servir para la transferencia de estilo. En este caso utilice una pintura mía, la primera, otra de mi tío, otra de mi primo y 3 imágenes obtenidas de img2go.

Imágenes de Estilo



TensorFlow es una biblioteca de código abierto desarrollada por Google Brain Team para realizar cálculos numéricos eficientes y crear modelos de aprendizaje automático, especialmente redes neuronales. Se utiliza ampliamente en investigación y producción para una variedad de tareas de aprendizaje automático y procesamiento de datos, como clasificación de imágenes, procesamiento de lenguaje natural, y más.

Proporciona una interfaz de programación flexible y extensible que permite a los desarrolladores crear y entrenar modelos de aprendizaje automático de manera eficiente. Ofrece características como gráficos computacionales para definir y ejecutar

operaciones matemáticas, optimizadores para entrenar modelos, y una variedad de herramientas y bibliotecas adicionales para simplificar el desarrollo de modelos de aprendizaje automático.

TensorFlow se utiliza en una amplia gama de aplicaciones, desde investigación en Inteligencia Artificial hasta desarrollo de productos comerciales. Es especialmente popular en la comunidad de aprendizaje automático debido a su flexibilidad, escalabilidad y soporte para una variedad de dispositivos y plataformas, incluyendo CPUs, GPUs y TPUs.

TensorFlow Hub es una biblioteca desarrollada por Google que proporciona una plataforma para compartir, descubrir y reutilizar módulos preentrenados de aprendizaje profundo. Estos módulos, también conocidos como “módulos de TensorFlow Hub”, son modelos de aprendizaje automático preentrenados y componentes reutilizables que pueden utilizarse para tareas específicas de procesamiento de datos, como la clasificación de imágenes, la extracción de características y la transferencia de estilo.

TensorFlow Hub simplifica el proceso de integración de modelos preentrenados en aplicaciones de aprendizaje automático al proporcionar una interfaz unificada y fácil de usar para buscar y descargar módulos, así como para utilizarlos en código. Los módulos de TensorFlow Hub son compatibles con TensorFlow y se pueden cargar y utilizar directamente en modelos personalizados.

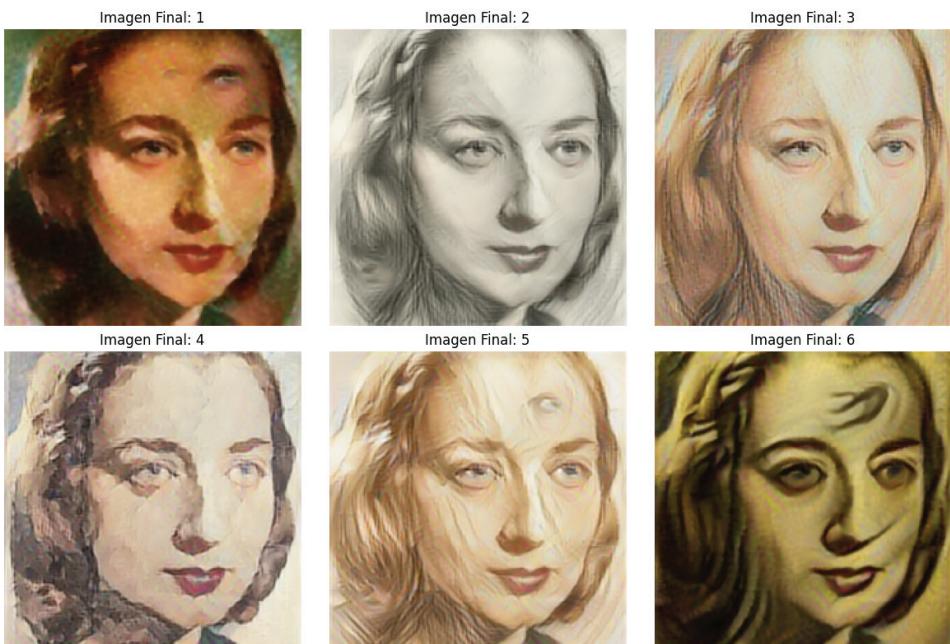
Usamos el siguiente código para utilizar el módulo de transferencia:

```
#Carga desde kaggle el modelo de GOOGLE Variación 256 versión 2
_= "https://www.kaggle.com/models/google/arbitrary-image-stylization-v1/frameworks/TensorFlow1/variations/256/versions/2"
style_transfer_model=hub.load(_)
# Transferencia de Estilos
stylized_images = []
for i, style_image in enumerate(style_url):
    style_image = load_image(style_image)
    stylized_image = style_transfer_model(content, style_image)
    stylized_images.append(stylized_image[0])
```

url_m: esta variable contiene la URL del módulo de TensorFlow Hub que se va a cargar. En este caso, el módulo es “Arbitrary Image Stylization” de Magenta, que permite estilizar imágenes con un estilo dado.

style_transfer_model: esta variable carga en la memoria el módulo del modelo, utilizando la función hub.load(). Este módulo contiene el modelo preentrenado necesario para realizar la transferencia de estilo.

En resumen, este paso del código carga un módulo preentrenado de TensorFlow Hub que se utiliza para estilizar una imagen de contenido con las imágenes de estilo específicas. Este proceso es bastante rápido, especialmente cuando se ejecuta en una GPU, y proporciona una manera eficiente de aplicar efectos de estilo a las imágenes.

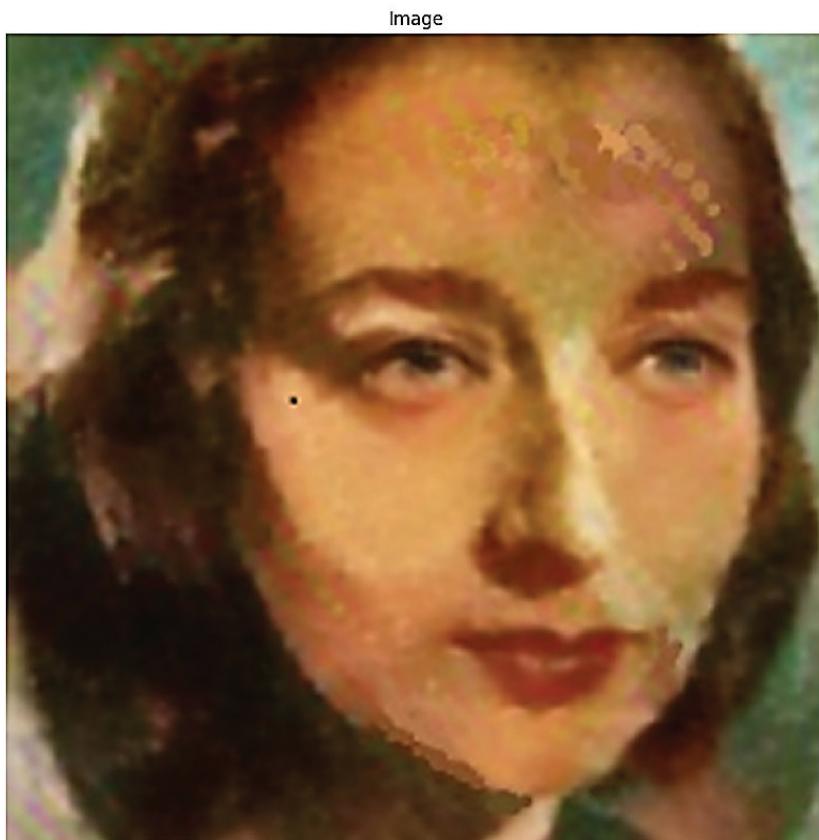


Las imágenes de arriba son el resultado de transferir a la imagen contenido los estilos de las imágenes de estilo, en este caso a través del algoritmo de TensorFlow Hub. Comprobamos como efectivamente cada una de las imágenes finales representan las caras de la retratada con el estilo adecuado.

Vemos que el resultado es bueno, pero que las imágenes contienen también ruido, cosas raras, trazos, ojos, etc. Para eliminar todo ello vamos a realizar un segundo proceso. Para ello elegimos la imagen de arriba a la izquierda.

6.4 MÓDULO IMAGEN FOTOGRÁFICA A INTERPRETADA O DIBUJADA (PASO 2)

En este segundo paso, tomamos la imagen obtenida en el paso anterior y, a través de un proceso manual, eliminamos alguno de los ojos y cosas raras de la imagen.



Vamos a utilizar las siguientes bibliotecas:

- ▶ **OpenCV (cv2):** desarrollado originalmente por Intel en 1999 y desde entonces mantenido por una comunidad activa de desarrolladores de código abierto, OpenCV es una de las bibliotecas más utilizadas en el campo de la visión por computadora y el procesamiento de imágenes. En este código, se utiliza para cargar la imagen desde el disco.
- ▶ **NumPy (np):** creado por Travis Oliphant en 2005 como parte del proyecto SciPy, NumPy es esencial en el ecosistema de Python para computación