

A classic impressionist painting of a harbor at sunset. The sky is filled with soft, warm colors of orange, yellow, and red, transitioning into cooler blues and greys. In the foreground, a small boat with two figures is silhouetted against the bright water. Several larger boats are moored along the dark, indistinct shore in the background. The brushwork is visible and expressive, creating a sense of light and atmosphere.

# Generación de cuadros impresionistas mediante Redes Neuronales

Por Victor Manuel Domínguez Rivas



# Introducción

- Avances recientes en IA: GPT3, AlphaGo...
- ¿Pueden las máquinas pensar?
- ¿Pueden las máquinas crear obras de arte?



# Objetivo de este TFG

Implementación de un modelo de Inteligencia Artificial que emule a

- Monet
- Van Gogh
- Cézanne



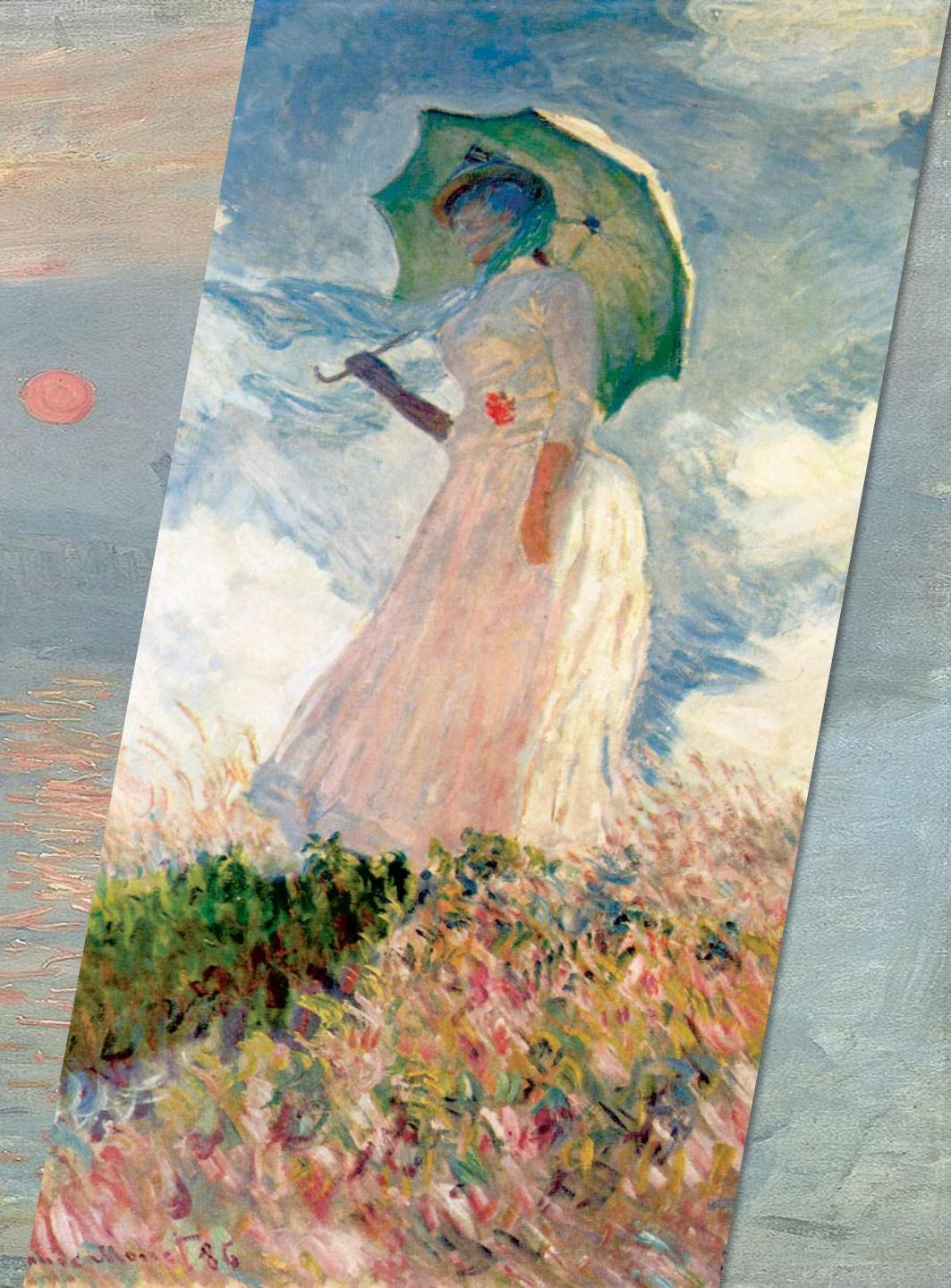
# Impresionismo

- Movimiento artístico del siglo XIX
- Muestra lo continuo, lo indefinido, el cambio perpetuo
- Formas imprecisas, pinceladas gruesas
- Experimentación con la luz y el color (paletas de varios colores)

# Claude Monet

Claude Monet (1840-1926), pintor francés, fue la figura clave del movimiento impresionista

- Innovó en el tratamiento de la luz y del color
- Tonos azulados
- Pinceladas y texturas suaves
- Captaba el instante utilizando *manchas*
- Pintura al aire libre



# Paul Cézanne

Paul Cézanne (1839-1906), pintor francés.  
Pintor ignorado en vida, tomó un camino personal.

- Intentó aunar el orden, la expresión personal y el naturalismo
- Formas simples y planos de color
- Representación simultánea de lo que el ojo observa y una abstracción de la observación
- Influyó a los cubistas (Picasso)



# Vincent Van Gogh

Vincent Van Gogh (1853-1890), pintor holandés, pintó más de 900 cuadros y 1600 dibujos.

- Pintura atemporal, pinceladas bruscas
- Colores chillones (especialmente amarillos) y vivos
- Abandono del naturalismo
- Formas que parecen moverse o caerse

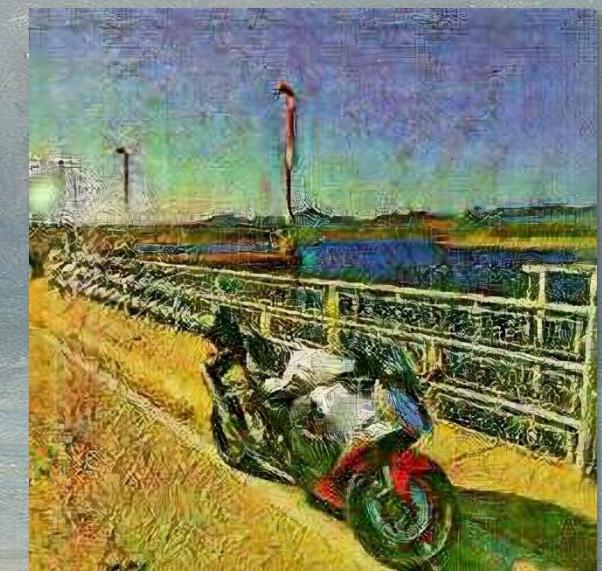


# Style Transfer

Conversión de un dominio a otro.

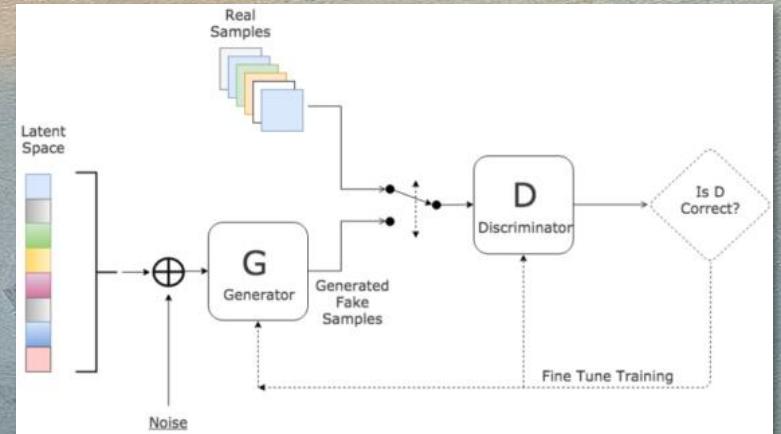
Principalmente se usan dos técnicas:

- Neural Style Transfer:  
Aprende el estilo de una sola imagen
- Cycle GAN: Aprende el estilo de un conjunto de imágenes



# Generative Adversarial Networks

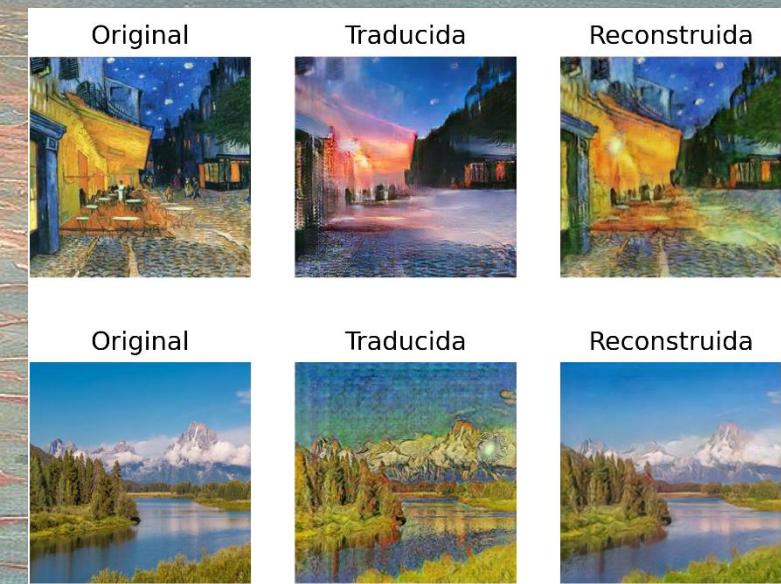
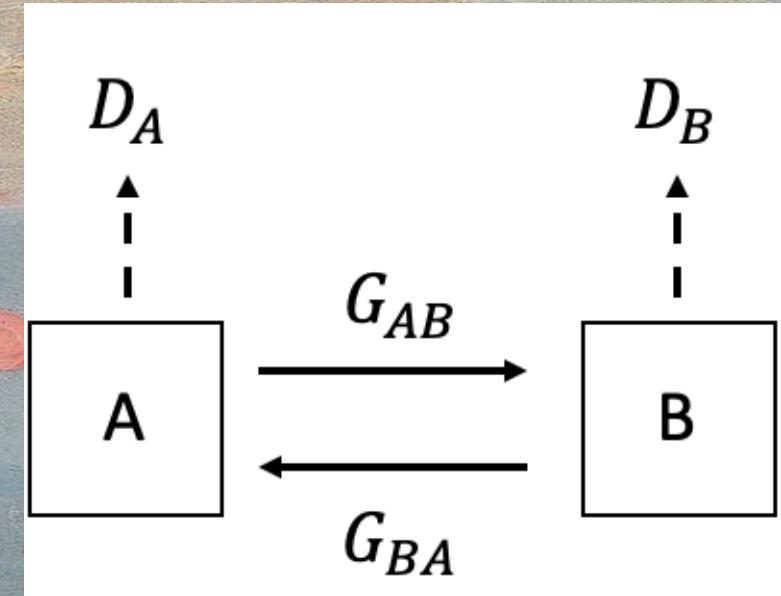
- Modelo del año 2014
- Dos redes neuronales enfrentadas: generador y discriminador
- Usadas para crear (p.ej StyleGAN)
- Complicadas de entrenar: equilibrio de Nash entre otros



# Cycle GAN

Modelo del año 2017

- Permite el cambio de dominios en ambos sentidos
- No necesitamos pares de ejemplos
- Compuesto de 2 GAN
- Tres métricas para el aprendizaje:
  - Error validez
  - Error reconstrucción
  - Error identidad
- Poca resolución



# Enhance Net

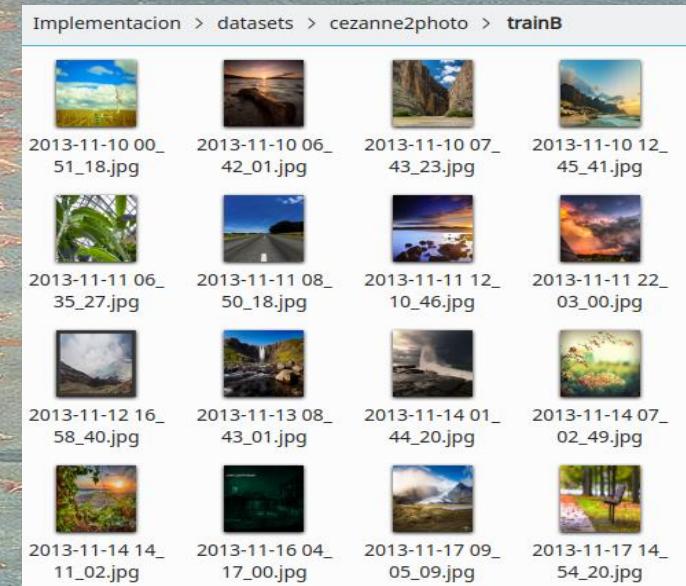
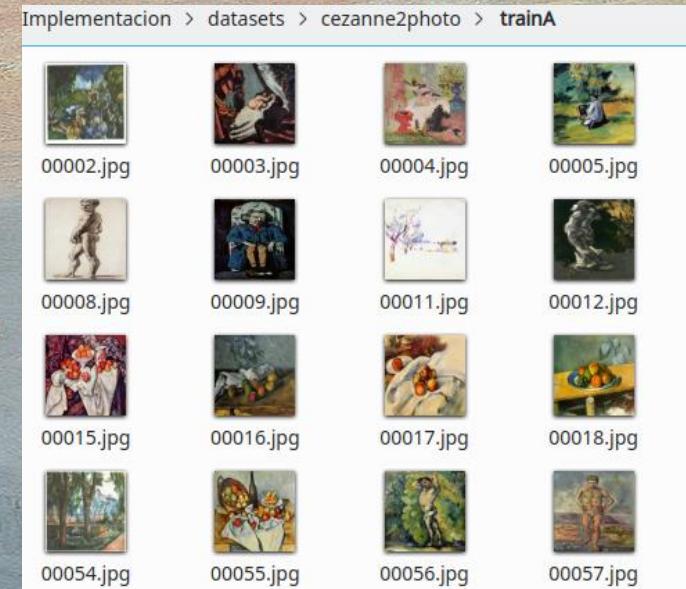
- Red neuronal convolucional para súper resolución
- Gran rendimiento
- Aumento en un factor 4x
- Implementación y pesos publicados en Github



Output size	Layer
$w \times h \times c$	Input $I_{\text{LR}}$
	Conv, ReLU
$w \times h \times 64$	Residual: Conv, ReLU, Conv
	...
$2w \times 2h \times 64$	2x nearest neighbor upsampling Conv, ReLU
$4w \times 4h \times 64$	2x nearest neighbor upsampling Conv, ReLU
	Conv, ReLU
$4w \times 4h \times c$	Conv Residual image $I_{\text{res}}$ Output $I_{\text{est}} = I_{\text{bicubic}} + I_{\text{res}}$

# Datasets utilizados

- Datasets elaborados por los creadores del modelo  
CycleGAN:
  - monet2photo
  - cezanne2photo
  - Vangogh2photo
- Estructura en cuatro carpetas

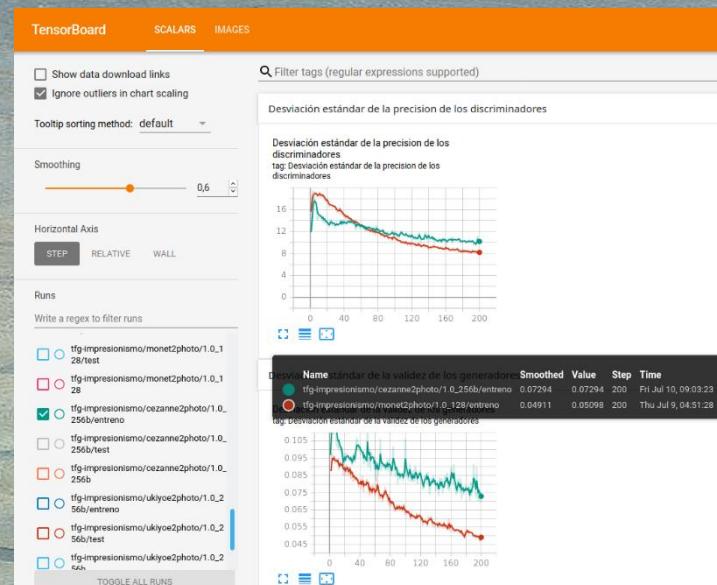


# Implementación: tecnologías

- TensorFlow y TensorBoard
- Keras
- Python
- CUDA y cuDNN
- Google Cloud Platform
- Flask
- Git

The screenshot shows a file listing interface. At the top, there's a header with tabs for 'OBJETOS', 'CONFIGURACIÓN', 'PERMISOS', 'RETENCIÓN', and 'CICLO DE VIDA'. Below the header, it says 'Depósitos > tfg-impressionismo > cezanne2photo > 1.0.128'. There are buttons for 'UPLOAD FILES', 'UPLOAD FOLDER', 'CREAR CARPETA', 'ADMINISTRAR CONSERVACIONES', and 'BORRAR'. A 'Filtrar' (Filter) button is also present. The main area displays a table with columns: Nombre (Name), Tamaño (Size), Tipo (Type), Created time (Created time), Clase de almacenamiento (Storage class), and Última modificación (Last modified). The table lists numerous files, mostly named 'events.out.tfevents.1594167...' with sizes ranging from 421.7 KB to 429.7 KB, all created on July 8, 2020, at various times between 02:00 and 08:00.

Nombre	Tamaño	Tipo	Created time	Clase de almacenamiento	Última modificación
cyclegan.log	29.3 KB	application/octet-stream	8 jul. 2020 08:00	Standard	8 jul. 2020 08:16...
entreno/	—	—	—	—	—
events.out.tfevents.1594167109.mv	431.6 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:11...
events.out.tfevents.1594167219.mv	421.7 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:13...
events.out.tfevents.1594167327.mv	427.5 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:15...
events.out.tfevents.1594167432.mv	431 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:17...
events.out.tfevents.1594167538.mv	428.6 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:19...
events.out.tfevents.1594167646.mv	427 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:20...
events.out.tfevents.1594167753.mv	429.7 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:22...
events.out.tfevents.1594167861.mv	424.7 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:24...
events.out.tfevents.1594167968.mv	421.7 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:26...
events.out.tfevents.1594168075.mv	423 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:27...
events.out.tfevents.1594168182.mv	425.3 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:29...
events.out.tfevents.1594168292.mv	417.9 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:31...
events.out.tfevents.1594168406.mv	415.5 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:33...
events.out.tfevents.1594168520.mv	415.8 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:35...
events.out.tfevents.1594168624.mv	419.5 KB	application/octet-stream	8 jul. 2020 02:00	Standard	8 jul. 2020 02:37...



# Implementación: Sistema principal

## Implementación CycleGAN:

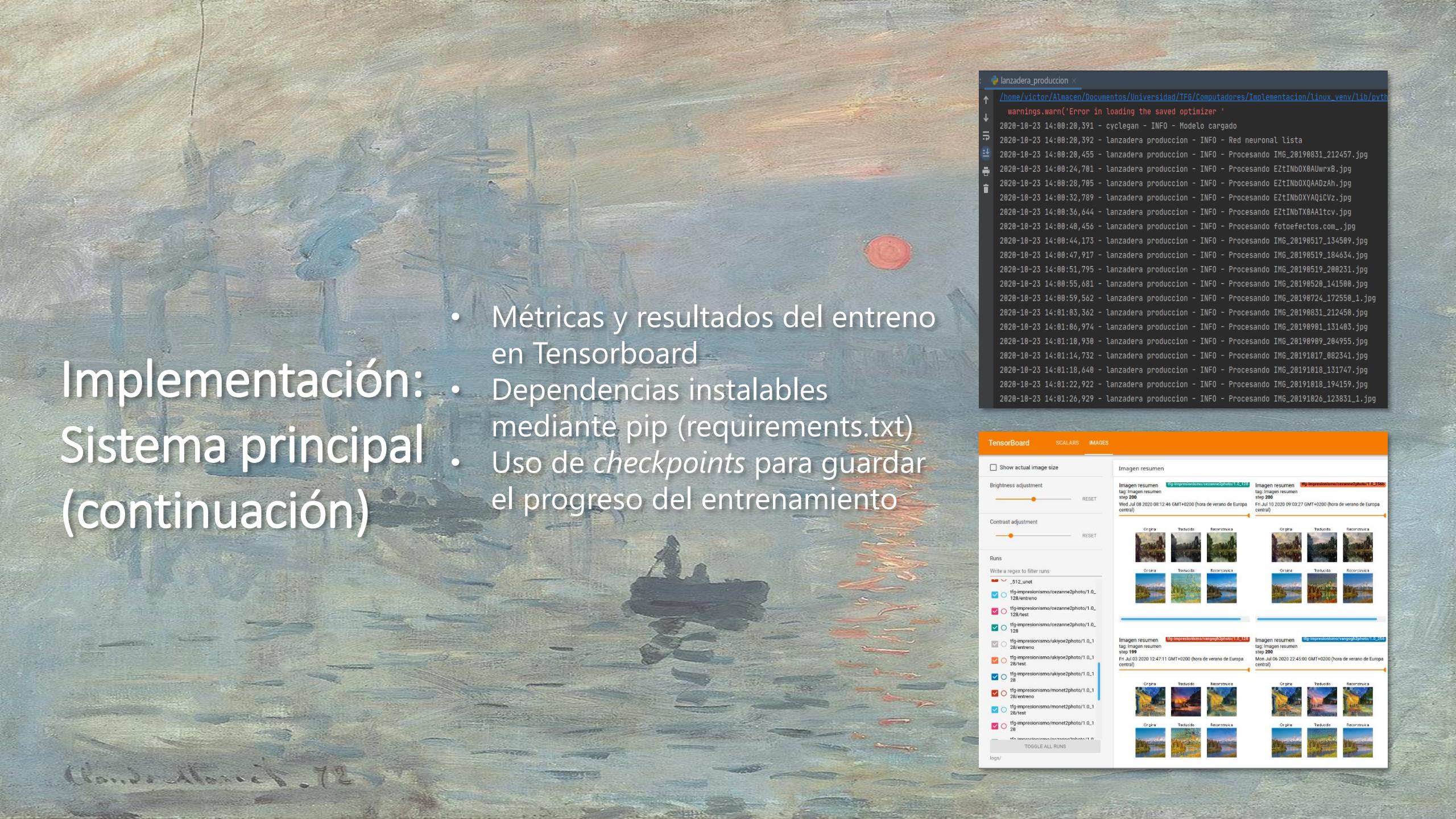
- Utilizable para entrenamiento e inferencia
- Parámetros del modelo configurables mediante archivos json
- Descarga automática de datasets
- Código eficiente: TensorFlow datasets, Python slots...

```
{  
    "configuracion_modelo":  
    {  
        "tasa_aprendizaje": "0.0002",  
        "lambda_reconstruccion": "10.0",  
        "lambda_validacion": "1",  
        "lambda_identidad": "2",  
        "ancho": "256",  
        "alto": "256",  
        "canales": "3",  
        "epochs": "200",  
        "tamanio_buffer": "1000",  
        "tamanio_batch": "1",  
        "filtros_generador": "64",  
        "filtros_discriminador": "64"  
    },  
    "url":  
    {  
        "api_aumento": "http://localhost:5000/TFG-Computadores/api-aumento/aumento",  
        "datasets": "https://people.eecs.berkeley.edu/~taesung_park/CycleGAN/datasets/"  
    },  
    "gcp":  
    {  
        "bucket": "gs://tfwg-impressionismo/"  
    },  
    "dataset":  
    {  
        "monet2photo":  
        {  
            "imagen_pintor_muestra": "00360.jpg",  
            "imagen_foto_muestra": "2014-08-24_16_41_27.jpg"  
        },  
        "cezanne2photo":  
        {}  
    }  
}
```

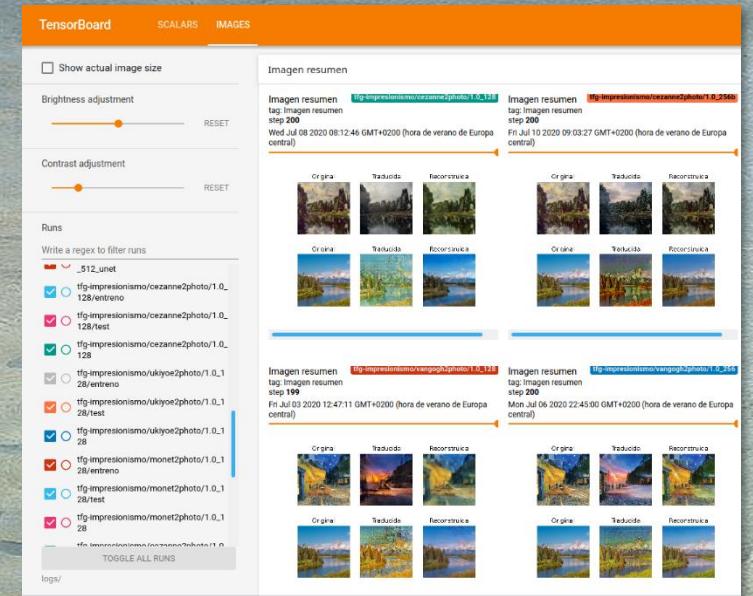
```
class CargadorImagenes(metaclass=Singleton):  
    """Clase que sirve para alimentar de imágenes a la red cuando entrena"""  
  
    __slots__ = ["tamanio_batch", "tamanio_buffer", "dimensiones",  
                "logger", "numero_imagenes_entreno_pintor",  
                "numero_imagenes_entreno_foto", "numero_imagenes_test_pintor",  
                "numero_imagenes_test_foto", "dataset_entreno_pintor",  
                "dataset_entreno_foto", "dataset_test_pintor",  
                "dataset_test_foto"]  
  
    def __init__(self):  
  
        utils = Utilidades()  
  
        self.tamanio_batch = utils.obtener_tamanio_batch()  
        self.tamanio_buffer = utils.obtener_tamanio_buffer()  
        self.dimensiones = utils.obtener_dimensiones()  
  
        rutas_imagenes_entreno_pintor = utils.obtener_rutas_imagenes_entreno_pintor()  
        rutas_imagenes_entreno_foto = utils.obtener_rutas_imagenes_entreno_foto()  
        rutas_imagenes_test_pintor = utils.obtener_rutas_imagenes_test_pintor()  
        rutas_imagenes_test_foto = utils.obtener_rutas_imagenes_test_foto()  
  
        # Atributos para calcular n_batches de forma eficiente  
        self.numero_imagenes_entreno_pintor = len(rutas_imagenes_entreno_pintor)  
        self.numero_imagenes_entreno_foto = len(rutas_imagenes_entreno_foto)  
        self.numero_imagenes_test_pintor = len(rutas_imagenes_test_pintor)  
        self.numero_imagenes_test_foto = len(rutas_imagenes_test_foto)  
  
        self.logger = utils.obtener_logger("lector_imagenes")  
  
        inicio = timestamp()  
        listado_dataset_entreno_pintor = tf.data.Dataset.list_files(rutas_imagenes_entreno_pintor)
```

# Implementación: Sistema principal (continuación)

- Métricas y resultados del entrenamiento en Tensorboard
- Dependencias instalables mediante pip (requirements.txt)
- Uso de *checkpoints* para guardar el progreso del entrenamiento



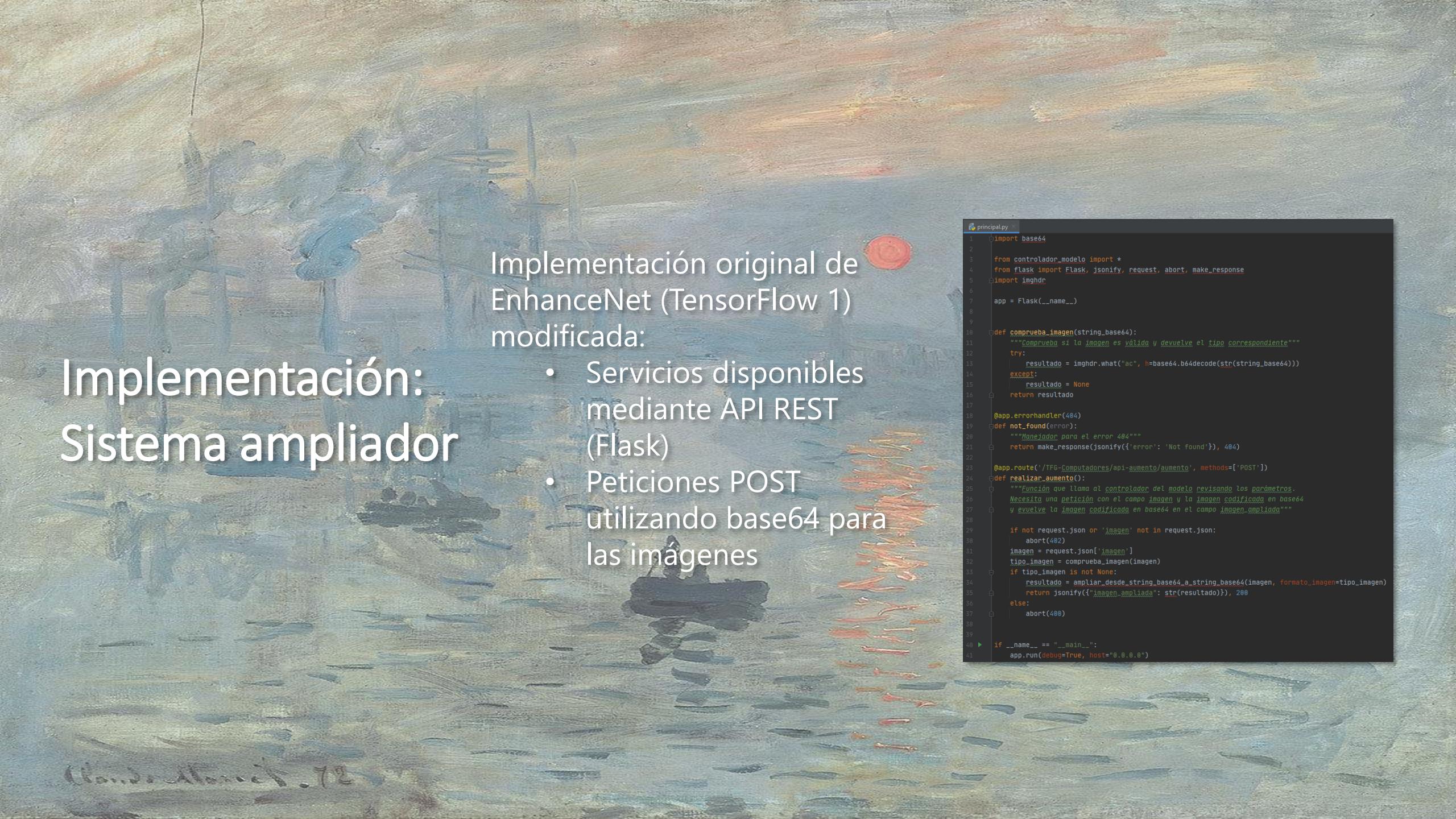
```
lanzadera_produccion x
/home/victor/Almacen/Documentos/Universidad/TFG/Computadores/Implementacion/linux_venv/lib/python3.7/site-packages/cyclegan/main.py:10: UserWarning: 'Error in loading the saved optimizer'
warnings.warn('Error in loading the saved optimizer')
2020-10-23 14:00:20,391 - cyclegan - INFO - Modelo cargado
2020-10-23 14:00:20,392 - lanzadera produccion - INFO - Red neuronal lista
2020-10-23 14:00:20,455 - lanzadera produccion - INFO - Procesando IMG_20190831_212457.jpg
2020-10-23 14:00:24,701 - lanzadera produccion - INFO - Procesando EZtINDOXAUwrxB.jpg
2020-10-23 14:00:28,705 - lanzadera produccion - INFO - Procesando EZtINDOXQAADzAh.jpg
2020-10-23 14:00:32,789 - lanzadera produccion - INFO - Procesando EZtINDXYAQICVz.jpg
2020-10-23 14:00:36,644 - lanzadera produccion - INFO - Procesando EZtINDTX0AA1tcv.jpg
2020-10-23 14:00:40,456 - lanzadera produccion - INFO - Procesando fotoefectos.com.jpg
2020-10-23 14:00:44,173 - lanzadera produccion - INFO - Procesando IMG_20190517_134509.jpg
2020-10-23 14:00:47,917 - lanzadera produccion - INFO - Procesando IMG_20190519_184634.jpg
2020-10-23 14:00:51,795 - lanzadera produccion - INFO - Procesando IMG_20190519_200231.jpg
2020-10-23 14:00:55,681 - lanzadera produccion - INFO - Procesando IMG_20190520_141500.jpg
2020-10-23 14:00:59,562 - lanzadera produccion - INFO - Procesando IMG_20190724_172550_1.jpg
2020-10-23 14:01:03,362 - lanzadera produccion - INFO - Procesando IMG_20190831_212450.jpg
2020-10-23 14:01:06,974 - lanzadera produccion - INFO - Procesando IMG_20190901_131403.jpg
2020-10-23 14:01:10,930 - lanzadera produccion - INFO - Procesando IMG_20190909_204955.jpg
2020-10-23 14:01:14,732 - lanzadera produccion - INFO - Procesando IMG_20191017_082341.jpg
2020-10-23 14:01:18,640 - lanzadera produccion - INFO - Procesando IMG_20191018_131747.jpg
2020-10-23 14:01:22,922 - lanzadera produccion - INFO - Procesando IMG_20191018_194159.jpg
2020-10-23 14:01:26,929 - lanzadera produccion - INFO - Procesando IMG_20191026_123831_1.jpg
```



# Implementación: Sistema ampliador

Implementación original de  
EnhanceNet (TensorFlow 1)  
modificada:

- Servicios disponibles mediante API REST (Flask)
- Peticiones POST utilizando base64 para las imágenes



A painting of a bridge over water by Claude Monet, showing a Impressionist style with visible brushstrokes and a focus on light and color.

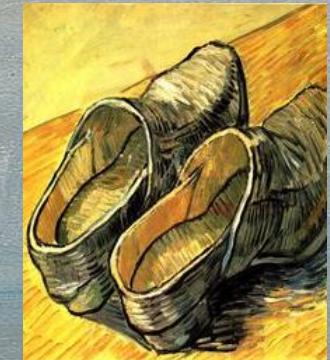
```
# principal.py
1 import base64
2
3 from controlador_modelo import *
4 from flask import Flask, jsonify, request, abort, make_response
5 import imghdr
6
7 app = Flask(__name__)
8
9
10 def comprueba_imagen(string_base64):
11     """Comprueba si la imagen es válida y devuelve el tipo correspondiente"""
12     try:
13         resultado = imghdr.what("ac", h=base64.b64decode(str(string_base64)))
14     except:
15         resultado = None
16     return resultado
17
18 @app.errorhandler(404)
19 def not_found(error):
20     """Manejador para el error 404"""
21     return make_response(jsonify({'error': 'Not found'}), 404)
22
23 @app.route('/TFG_Computadores/api-aumento/aumento', methods=[POST])
24 def realizar_aumento():
25     """Función que llama al controlador del modelo revisando los parámetros.
26     Necesita una petición con el campo imagen y la imagen codificada en base64
27     y devuelve la imagen codificada en base64 en el campo imagen_ampliada"""
28
29     if not request.json or 'imagen' not in request.json:
30         abort(402)
31     imagen = request.json['imagen']
32     tipo_imagen = comprueba_imagen(imagen)
33     if tipo_imagen is not None:
34         resultado = ampliar_desde_string_base64_a_string_base64(imagen, formato_imagen=tipo_imagen)
35         return jsonify({'imagen_ampliada': str(resultado)}), 200
36     else:
37         abort(400)
38
39
40 if __name__ == "__main__":
41     app.run(debug=True, host="0.0.0.0")
```

# Resultados: Cuadro a imagen

De izquierda a derecha: Monet, Cézanne, Van Gogh

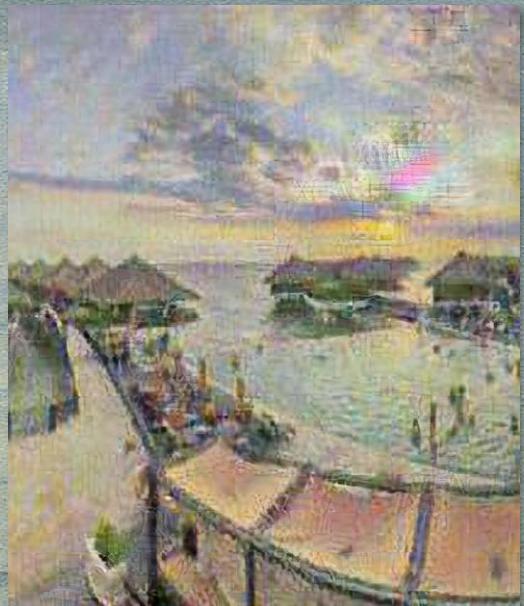
De arriba abajo: cuadro, imagen generada

- Conseguido gracias a las características de Cycle GAN
- Es capaz de *deshacer* la paleta de colores característica del pintor en la mayoría de casos (especialmente en Monet)
- Resultados mejorables en las texturas *reales* en las que se basan los cuadros
- Malos resultados con retratos de personas



# Resultados: Imagen a cuadro (Monet)

- Es capaz de adquirir la tonalidad azulada propia de Monet
- Adquiere la textura de óleo sobre lienzo
- No difumina las texturas, ausencia de *manchas*
- Fallos de iluminación



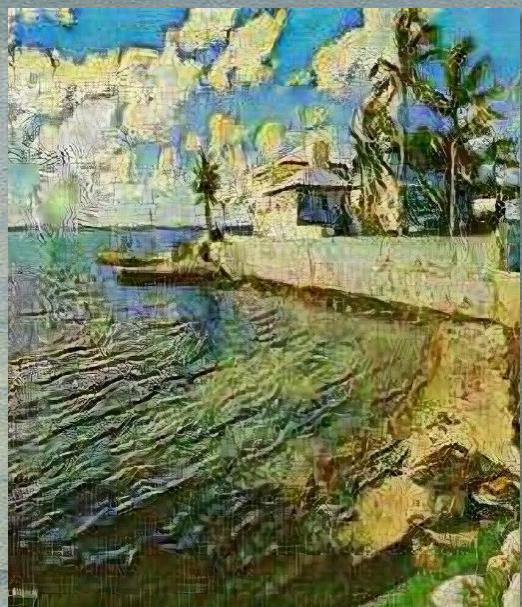
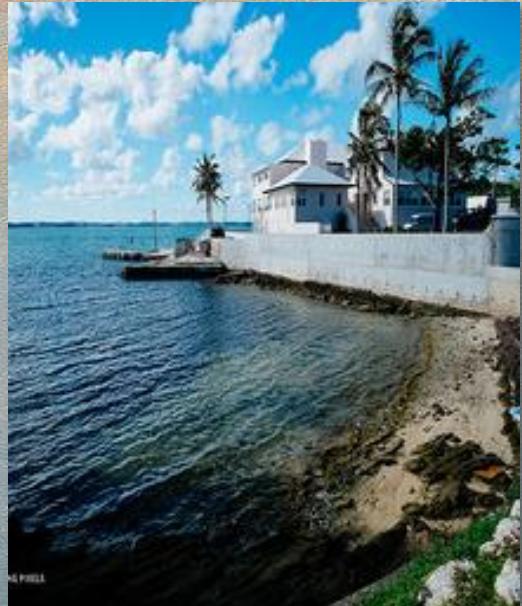
# Resultados: Imagen a cuadro (Cézanne)

- Adquiere muy bien los tonos rojizos: véase los árboles y edificios
- Texturas relativamente bien conseguidas
- Transmite sensación de desenfoque, algo que no presenta Monet
- Fallos de iluminación



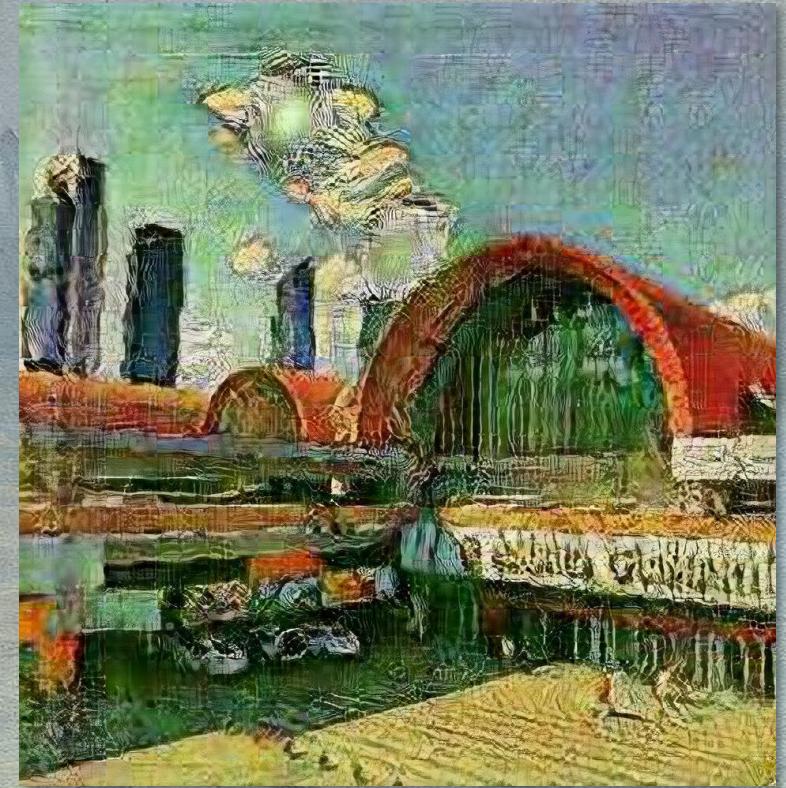
# Resultados: Imagen a cuadro (Van Gogh)

- Representa muy bien las tonalidades de Van Gogh: colores amarillentos y chillones
- Realiza cambios de texturas con relativa eficacia: véase las nubes, el agua y la camiseta
- Sin embargo, no capta las pinceladas bruscas de Van Gogh en los fondos
- Fallos de iluminación, si bien no tan visibles como en otros pintores



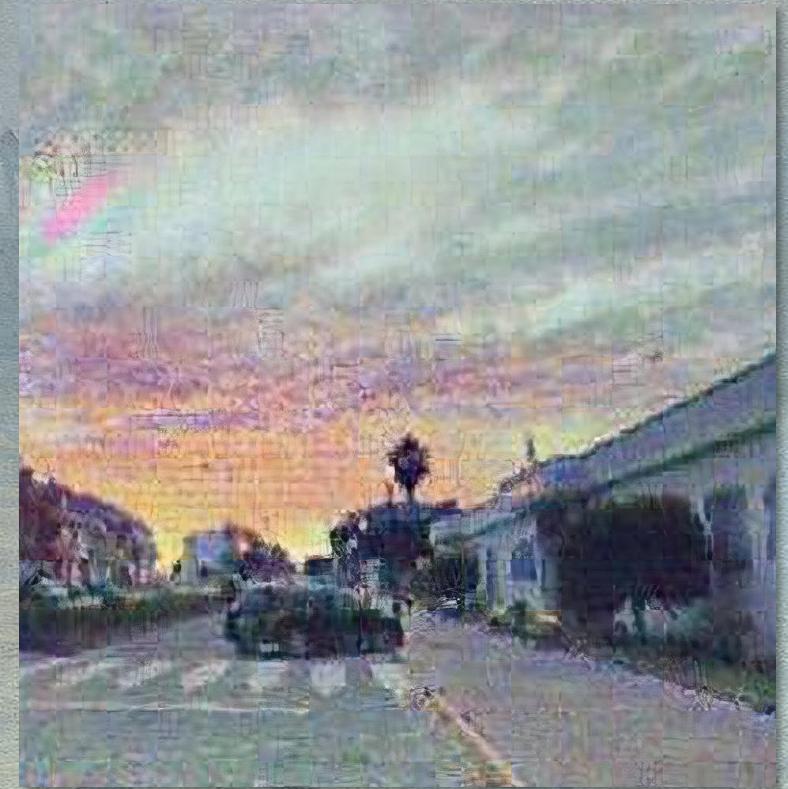
# Problemas encontrados

- Ausencia de correlación entre las métricas y la calidad artística
- Incompatibilidad de la implementación EnhanceNet con TensorFlow 2
- Entrenamiento de las redes CycleGAN muy costoso, lo que limita la experimentación y el diseño
- Dificultad de implementar y depurar la red. Problemas de iluminación y resultados irregulares
- Documentación del modelo y de algunas las librerías escasa y/o confusa



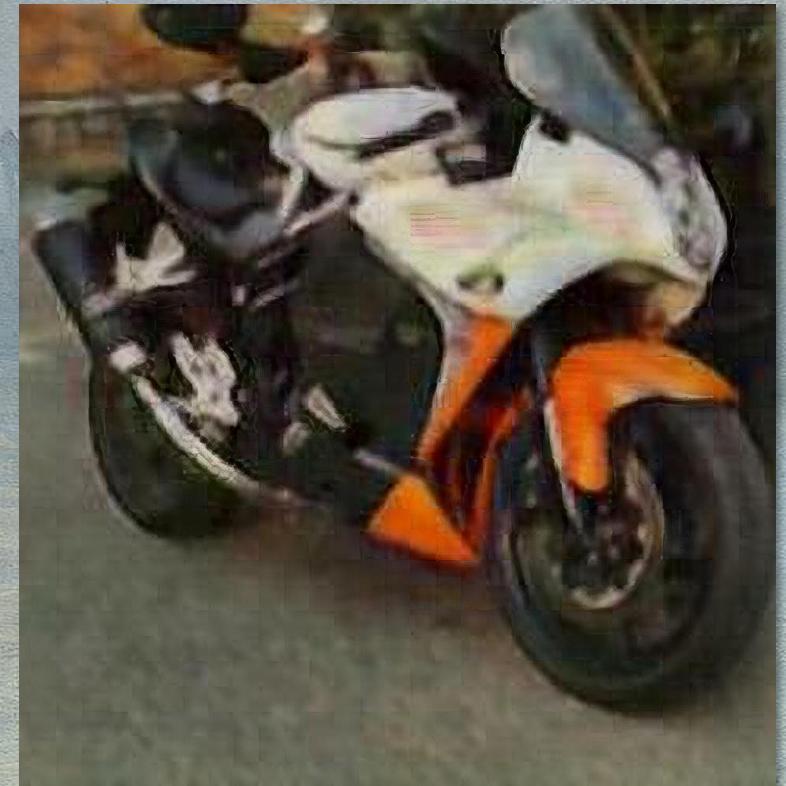
# Impacto social y medioambiental

- Inteligencia Artificial como complemento al artista
- No dispone de creatividad innata, solo imita
- Entrenamiento costoso en recursos, lo que limita su acceso
- Centros de cómputo con energías renovables



## Líneas futuras

- Mejorar la integración del sistema ampliador
- Aplicación de escritorio y/o móvil más accesible
- Refinar la implementación para solucionar errores de iluminación
- Añadir nuevos pintores y/o combinaciones de pintores
- Entrenar al sistema con mayor resolución de salida



# Para terminar...

Yo no tengo ídolos. Tengo admiración por el trabajo, la dedicación y la competencia.

Ayrton Senna da Silva, triple campeón de Fórmula 1



No es mucho, pero es trabajo honesto





¡Gracias!