Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Máster Profesional en Ingeniería Informática

Curso 2020/2021

TRANSFERENCIA DE ESTILO EN IMÁGENES

Sistemas Inteligentes para la Gestión de la Empresa

Breve descripción

Memoria para la evaluación de la parte teórica

Autor

Álvaro de la Flor Bonilla (alvdebon@correo.ugr.es) 15408846-L

RESUMEN

La transferencia de estilo neuronal es una técnica de optimización que se utiliza siguiendo un esquema en el que, a través de del uso de dos imágenes, se toma una referencia para el contenido y otra para el estilo (por ejemplo, una obra de arte).

El objetivo que se pretende alcanzar es combinar ambas de tal forma que la imagen resultante cuente con el contenido de la imagen contenido, pero con el efecto visual de contar con el estilo de la imagen de referencia de estilo.

ÍNDICE DEL PROYECTO

Resumen **1**

1 Introducción **4**

2 Deep Learning – Toma de contacto **5**

2.1 Aprendizaje no supervisado 5

2.2 Aprendizaje supervisado 5

2.2.1 CNN – Redes Neuronales Convolutivas 6

3 Tranferencia de estilo en imágenes **8**

3.1 Función de pérdida 9

3.1.1 Función pérdida de contenido 10

3.1.2 Función pérdida de estilo 10

3.2 Alternativas 11

4 Ejemplo Práctico **13**

4.1 Neural Style Tranfer 13

4.2 Fast Neural Style Tranfer 14

5 Conclusiones **15**

ÍNDICE DE ILUSTRACIONES

Ilustración 1 – Capas de una red neuronal convolucional 6

Ilustración 2 – Detección del cáncer con imágenes Fuente: https://www.nature.com/articles/s41598-019-48995-4 7

Ilustración 3 – Ejemplo de transferencia de estilo 8

Ilustración 4 – Funcionamiento de la Red 9

Ilustración 5 – Ecuación de pérdida 9

Ilustración 6 - NST Diagram 10

Ilustración 7 – Función pérdida de contenido 10

Ilustración 8 – Función pérdida de estilo 10

Ilustración 9 - Cálulo de la matriz CNN con cinco mapas 11

Ilustración 10 – Obra *“La noche estrellada”* 13

Ilustración 11 - Noche estrellada con algoritmo alternativo 14

1 INTRODUCCIÓN

¿Por qué es tan importante la transferencia de estilo en imágenes? Hoy en día una de las industrias más fuertes que existen a nivel internacional, aunque suene a broma son las redes sociales.

A modo resumen, la transferencia de estilo de imágenes la podemos entender como una nueva forma de arte, en la que combinar tanto lo realista como lo subjetivo. El mercado actual está repleto de aplicaciones como *"Snapchat"*, *"Instagram"*, *"Facebook"* … cuyo objetivo es ofrecerle algo, distinto a publicar una simple foto. En definitiva, como gran moda que estamos observando estos años es tratar de *"gamificar"* todo lo que podamos hacer con nuestros dispositivos, y este campo no se iba a quedar atrás.

El caso más ejemplarizante que podemos señalar sin duda puede ser *"Instagram"*. Nada más seleccionar una foto que se desea publicar nos ofrece una infinitud de filtros a usar, sin olvidar tampoco que también es capaz de aplicarlos (otro tipo de filtros) en tiempo real e incluso en vídeo en sus llamadas "*stories"*.

En definitiva, como hemos indicado anteriormente el objetivo de esta técnica es utilizar dos imágenes y de ellas ser capaces de unir el contenido de una con el estilo (normalmente artístico) de la otra. Está técnica es conocida en inglés como *"NST – Neural Style Transfer"* la cual ha tenido un avance significativo en la última década ya que su núcleo central se fundamenta en el uso redes neuronales que actualmente están viviendo una etapa dorada.

Centrándonos aún más en el párrafo anterior, nuestras redes neuronales hacen uso de algoritmos como el *"Deep Learning"* o aprendizaje profundo como veremos en las próximas secciones de nuestro estudio.

2 DEEP LEARNING – TOMA DE CONTACTO

Básicamente, el proceso de transferencia de estilo en imágenes se basa en un método construido a partir de aprendizaje automático en el que de forma completamente autónoma el sistema desarrollado es capaz de transferir el estilo de una imagen a otra.

Este aprendizaje automático es un tipo de inteligencia artificial que permite *"obtener"* la capacidad de aprender, pero sin ser programadas explícitamente para ello, es decir, el sistema desarrollado es capaz de cambiar por si solo cuando se enfrentan a nuevos datos.

Aunque existen grandes diferencias, podemos llegar a comparar el proceso que se lleva a cabo en la minería de datos como el de aprendizaje automático. Los dos sistemas buscan encontrar un patrón entre la maraña de datos que se dan, pero mientras en la minería de datos lo que se busca es extraer datos para la comprensión humana, el aprendizaje automático se centra en utilizar los datos anteriores para detectar patrones y ajustar las acciones que llevara a cabo el programa en consecuencia.

Existen muchísimos tipos de algoritmos de aprendizaje automático, pero en este estudio hemos preferido clasificarlos en tres grandes bloques (según la metodología de aprendizaje):

1. Aprendizaje supervisado.
2. Aprendizaje sin supervisión.

2.1 Aprendizaje no supervisado

Esta variante el algoritmo trabajará con grandes volúmenes de datos para analizar patrones, pero a diferencia de la metodología que veremos a continuación (aprendizaje supervisado), en esta ocasión el algoritmo no contará con las soluciones esperadas sobre las que realizar correcciones, no hay un conocimiento a priori.

En el aprendizaje no supervisado, los problemas que se presentan suelen ser más complejos que los anteriores ya que se espera que el modelo aprenda sin decirle el qué. Como solo se conoce los datos de entrada (pero no tenemos información respecto a la salida de ellos), sólo se puede describir la estructura de los datos y a partir de ello intentar encontrar algún tipo de organización que simplifique el análisis, por lo que tiene un carácter exploratorio.

2.2 Aprendizaje supervisado

En este tipo de aprendizaje el sistema es capaz de *"aprender"* mediante un ejemplo. Es decir, se proporciona al algoritmo de aprendizaje automático diseñado un conjunto de datos que contienen tanto entradas como las salidas que esperamos y la función del algoritmo es la de encontrar los patrones y métodos que permiten determinar cómo conseguir estas entradas y salidas esperadas.

Durante las distintas iteraciones de entrenamiento, el algoritmo detecta patrones en la maraña de datos que le hemos dado, aprende de ellas y realiza predicciones. A lo largo de este entrenamiento el algoritmo es capaz de mejorar sus parámetros hasta alcanzar un nivel suficiente tanto en precisión como en rendimiento.

2.2.1 CNN – Redes Neuronales Convolutivas

Las redes neuronales convolucionales se crearon originalmente para clasificar imágenes, pero últimamente se han utilizado en una gran variedad de tareas como la segmentación de imágenes, estilo neuronal y otras tareas de visión por ordenador y procesamiento del lenguaje natural.

Este tipo de redes llevan a cabo un tipo de aprendizaje supervisado, que procesa sus capas imitando al córtex visual del ojo humano para identificar las características de entrada que en definitiva permiten identificar objetos.

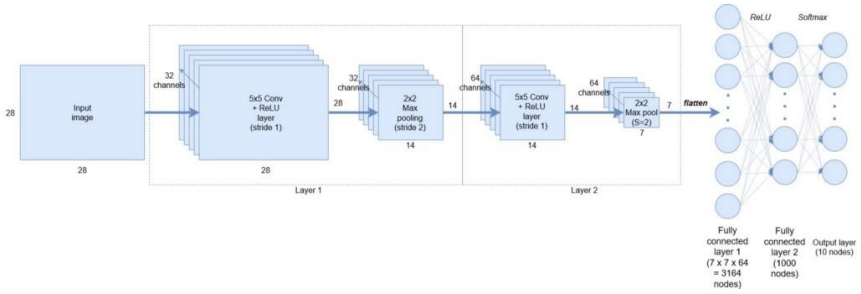


Ilustración 1 – Capas de una red neuronal convolucional

Cabe destacar que las CNN no aprenden a codificar qué imágenes son, en realidad lo que aprenden es a codificar qué representa o qué contenidos son visibles en la imagen y, gracias a la naturaleza no lineal de las redes neuronales, hemos pasado de capas superficiales a capas más profundas.

Las CNN se caracterizan por tener varias capas ocultas especializadas en detectar características aisladas de las distintas fotografías y/o imágenes. Por ejemplo, pueden existir capas cuyo objetivo es detectar líneas y curvas, mientras que el resto se pueden ir especializando hasta llegar al nivel de tener la capacidad de detectar desde la silueta de un animal hasta un rostro humano.

Las *"CNN"* tienen dos componentes:

1. **Extracción de características**

   Esta parte puede verla en la ilustración 1 y corresponde a las partes rectangulares. Se realizarán una serie de convoluciones y operaciones de agrupación en las que se detectarán las características.

2. **Clasificación**

   Esta parte con respecto a la ilustración 1 corresponde con los elementos que tienen forma redondeada. Las capas conectadas serán utilizadas como clasificador a partir de las características extraídas sobre la parte anterior.

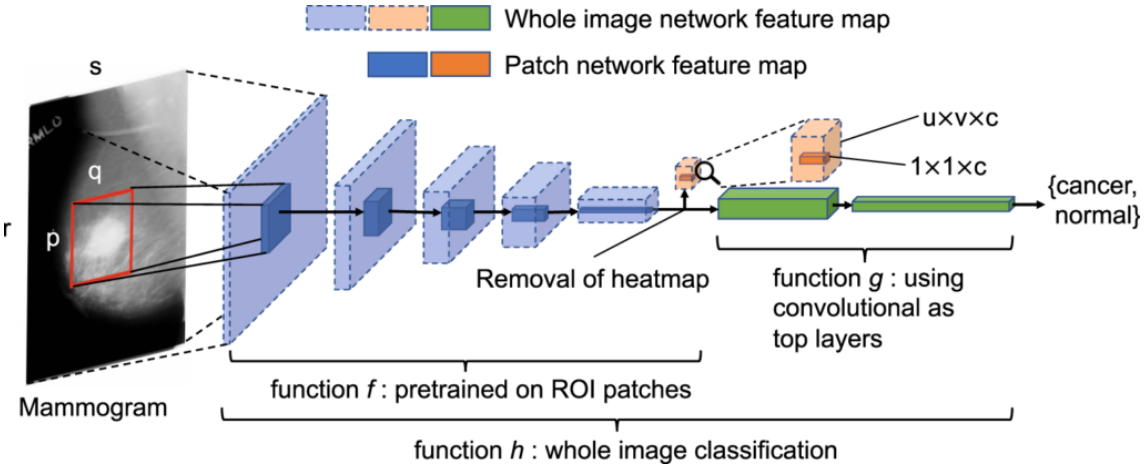


Ilustración 2 – Detección del cáncer con imágenes Fuente: https://www.nature.com/articles/s41598-019-48995-4

Este tipo de redes han tomado tal nivel de importancia, que tal y como puede verse en la ilustración anterior, en la actualidad es llegado a ser utilizado para ser capaces de detectar de una forma rápida, efectiva y minuciosa casos de cáncer de mama a partir del análisis (de forma completamente autónoma) de una mamografía. Son muy útiles para análisis de todo tipo de imágenes, muy utilizado también en la detección de células maliciosas o por ejemplo en el análisis de melanomas.

Lo más importante de las *"CNN"* es que, en un principio, las redes no saben que es la imagen, pero de manera autónoma aprenden a codificar que representa una imagen en particular. Esta forma de codificar puede ayudar (y de hecho se utiliza) para la transferencia de estilo neuronal que es la base de este trabajo.

3 TRANFERENCIA DE ESTILO EN IMÁGENES

Hasta ahora hemos hablado a nivel teórico de que la transferencia de estilo, pero realmente no hemos mostrado ningún ejemplo.

Ilustración 3 – Ejemplo de transferencia de estilo

En la ilustración anterior se muestra exactamente lo persigue esta metodología. Tenemos dos imágenes principales, la primera corresponde a una fotografía de la ciudad de Persépolis, capital del imperio persa, mientras que la segunda imagen corresponde al óleo *"La noche estrellada"* realiza por el autor neerlandés *"Vicent Van Gogh"*.

Como resultado se muestra en la zona de la derecha de la ilustración tres una combinación artística en la que se hace uso del contenido de la ilustración que captura la ciudad de Persépolis, pero haciendo uso del estilo mostrado en esta ocasión en el cuadro de *"Van Gogh"*. Llegados a este punto, ¿cómo funciona el mecanismo de la transferencia de estilo en imágenes (*"NST"*).

En primer lugar, y como ya adelantamos anteriormente, en *"NST"* se hace uso de redes convolucionales, en concreto la variante más utilizada en la actualidad es la red *"VGG-19"*.

"VGG-19" es una red neuronal convolucional preentrenada propiedad de Google que cuenta con una profundidad de 19 capas, construida y entrenada por K. Simonyan y A. Zisserman en la Universidad de Oxford en 2014 y es una de las implementaciones que utilizaremos en el ejemplo práctico. Esta red es capaz de utilizar más de 1 millón de imágenes procedentes de la base de datos de *"ImageNet"*. Como dato, esta red se entrenó con imágenes en color de 224x224 pixeles.

Comprendamos ahora, a través de un diagrama de flujo, como funciona nuestra red. En la ilustración que aparece a continuación. En primer lugar, para simplificar la ilustración no aparecen las 19 capas, téngalo en cuenta.

A nuestra red neuronal le introduciremos dos imágenes, una de contenido y otra de estilo. Se generará una imagen mixta que tendrá los contornos y textura de la imagen contenido y el patrón de color de la imagen de estilo. Durante sucesivas iteraciones repetiremos el diagrama, optimizando las funciones de pérdida.

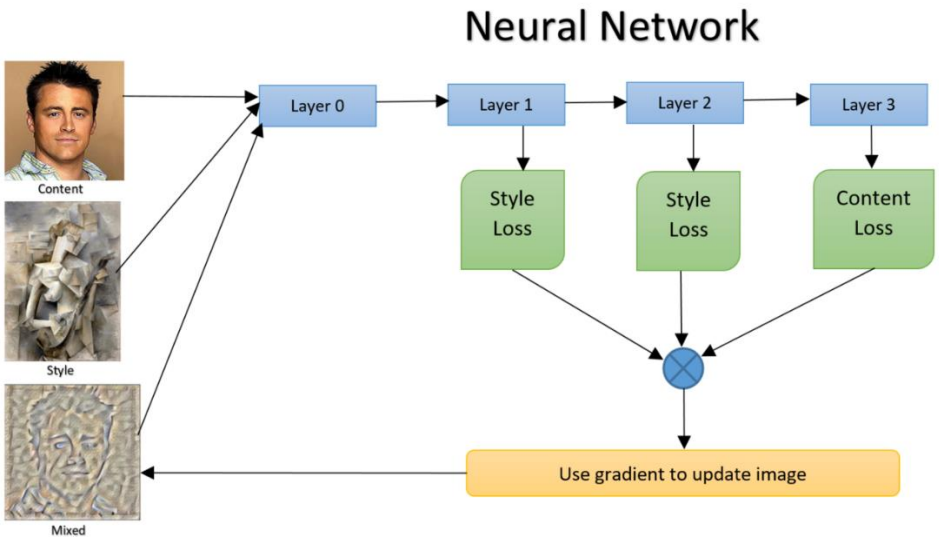


Ilustración 4 – Funcionamiento de la Red

Anteriormente mencionamos que las redes convolucionales están diseñadas de tal forma que cada una de sus capas de especializan en extraer cierto tipo de información o patrones ya sean líneas, contornos, etc.

El proceso descrito anteriormente (y conocido como aprendizaje de representación de características) es fundamental para llegar a poder realizar la transferencia de estilo. Como bien vimos en el diagrama anterior, se basa en el uso de funciones de pérdida, sin embargo, ¿cómo se logra que estas funciones actúen correctamente? Es decir, que se realice correctamente la extracción de contenido respecto a la primera imagen y de estilo respecto a la segunda, sin una mezcla o desviación entre ambas.

3.1 Función de pérdida

Según la documentación de *"TensorFlow"* que ha sido nuestra principal fuente para el desarrollo de este proyecto una de las mejores soluciones disponibles para evitar el problema de desviación entre contenidos de los que hablamos antes es dividir las funciones de pérdida entre contenido y estilo. Tal y como se muestra en la siguiente ecuación.

$$L = \alpha L_{content} + \beta L_{style},$$

Ilustración 5 – Ecuación de pérdida

En la ecuación anterior, *"Alpha"* y *"Beta"* corresponden a hiperparámetros que son definidos por el usuario. De esta forma, al tener el control de estas dos variables se pueden manejar la cantidad de contenido y estilo que se inyecta en la imagen generada.

En resumen, el proceso de optimización que hace uso de la ecuación anterior sigue el siguiente diagrama:

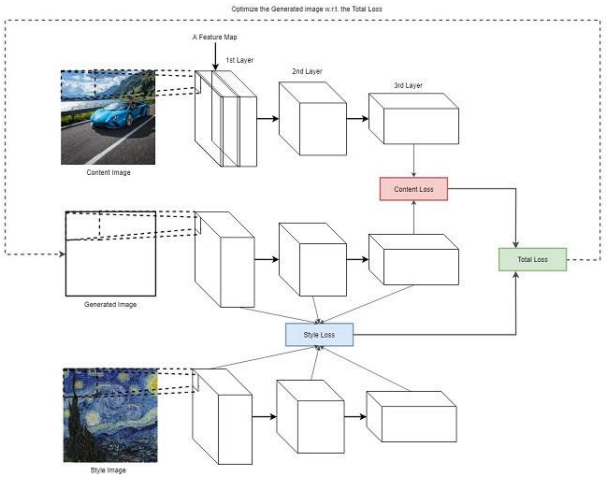


Ilustración 6 - NST Diagram

Centrémonos ahora un poco más en las funciones de pérdida por separado que hemos comentado.

3.1.1 Función pérdida de contenido

Esta función será la encargada de asegurarse de que el contenido que finalmente se presente en la imagen generada sea fiel al de la imagen modelo de contenido. Las *"CNN"* en la mayoría de las ocasiones realizan la captura de la información del contenido en los niveles más superiores, mientas los niveles más inferiores se especializan más en los valores de píxeles individuales.

Entendido lo anterior es lógico que las funciones de pérdida de contenido se centralicen en las capas más altas.

$$L_{content} = \frac{1}{2} \sum_{i,j} (A_{ij}^{l}(g) - A_{ij}^{l}(c))^2$$

Ilustración 7 – Función pérdida de contenido

La fórmula anterior indica que $L_{contenido}$ es iugal a la raíz del error cuadrático medio entre las activaciones producidas por la imagen generada y la imagen del contenido. ¿Por qué se decide usar esta ecuación? Pues porque tal y como indicamos antes, si la activación de los distintos mapas de características se activa en las capas superiores en presencia de diferentes objetos, si dos imágenes poseen el mismo contenido, deberían tener activaciones prácticamente iguales o similares en las capas superiores.

3.1.2 Función pérdida de estilo

Según la documentación oficial de *"VGG-19"*, la función de pérdida de estilo queda definida como:

$$L_{style} = \sum_{l} w^{l} L_{style}^{l} \text{ where,}$$

$$L_{style}^{l} = \frac{1}{M^{l}} \sum_{ij} (G_{ij}^{l}(s) - G_{ij}^{l}g)^2 \text{ where,}$$

$$G_{ij}^{l}(I) = \sum_{k} A_{ik}^{l}(I) A_{jk}^{l}(I).$$

Ilustración 8 – Función pérdida de estilo

A diferencia del caso anterior, para extraer el estilo la función es mucho más compleja, además de que en esta ocasión son usadas todas las capas de nuestra *"CNN"*.

La función de pérdida de información se define como la diferencia correlación que existe entre los mapas característicos calculados por la imagen generada y la imagen de estilo. En esta ocasión *"w^l"* representa un peso dado, a cada capa, durante el cálculo de la pérdida y *"M^l"* es un hiperparámetro que depende del tamaño de la l-ésima capa.

Lo que se pretende conseguir es lograr conformar una matriz de estilo tanto para la imagen generada como para la imagen de estilo. La pérdida de estilo se define como la raíz de la diferencia cuadrática media entre las dos matrices de estilo.

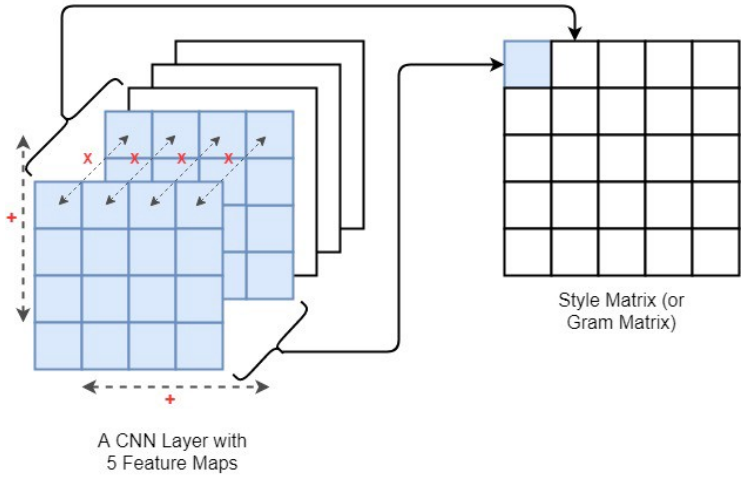


Ilustración 9 - Cálulo de la matriz CNN con cinco mapas

En la imagen anterior se muestra cómo se conforma una matriz de estilo. En este caso estamos construyendo una matriz de Gram en la que los elementos *"i"* y *"j"* se calculan a partir de la multiplicación por los elementos de los mapas de características i-ésimo y j-ésimo sumando también el ancho y el alto. En la figura de arriba el elemento rojo denota la operación y las flechas los elementos que se tienen en cuenta en dicha operación.

Una vez hemos logrado calcular la matriz anterior, seremos capaces de calcular la pérdida de estilo de exactamente la misma forma que lo hicimos con la pérdida de contenido.

3.2 Alternativas

El problema de la metodología anterior reside en que cada vez que utilizamos una nueva imagen de contenido se deben restablecer cada uno de los nuevos píxeles de la imagen generada y el proceso de búsqueda de píxeles se debe realizar nuevamente.

¿Qué ocurre entonces? Pues que el proceso es excesivamente lento (después lo veremos más en detalle, pero puede demorar hasta 20 minutos) y además no es seguro que se consigan buenos resultados.

Como solución se plantea intentar modificar el enfoque, de tal forma que se pueda construir una red neuronal capaz de aprender a aplicar un tipo específico de estilo en cualquier imagen de entrada. De esta forma, aunque este enfoque tampoco es muy bueno, es mucho mejor que el anterior.

En resumen, en este nuevo tipo de enfoque necesitaremos dos redes: una que será utilizada para extraer características de un modelo previamente entrenado y otra red que será utilizada para la transferencia.

Básicamente en el modelo anterior que fue diseñado por *"Gatys"* lo que hacemos es optimizar la imagen de salida respecto a la función de pérdida (pérdida de contenido + pérdida de estilo), siendo este como hemos dicho antes un proceso muy lento.

La transferencia de estilo rápido permite entrenar una vez y generar las imágenes infinitas, incluso en vídeos o imágenes en tiempo real (cosa imposible en el modelo anterior). Estas nuevas redes son conocidas como *"Fast neural style tranfer"*.

Como resumen esta red es mucho más rápida construyendo la imagen estilizada, pero como contra, deberemos contar con un modelo preentreanado para cada una de las imágenes estilo que deseemos utilizar.

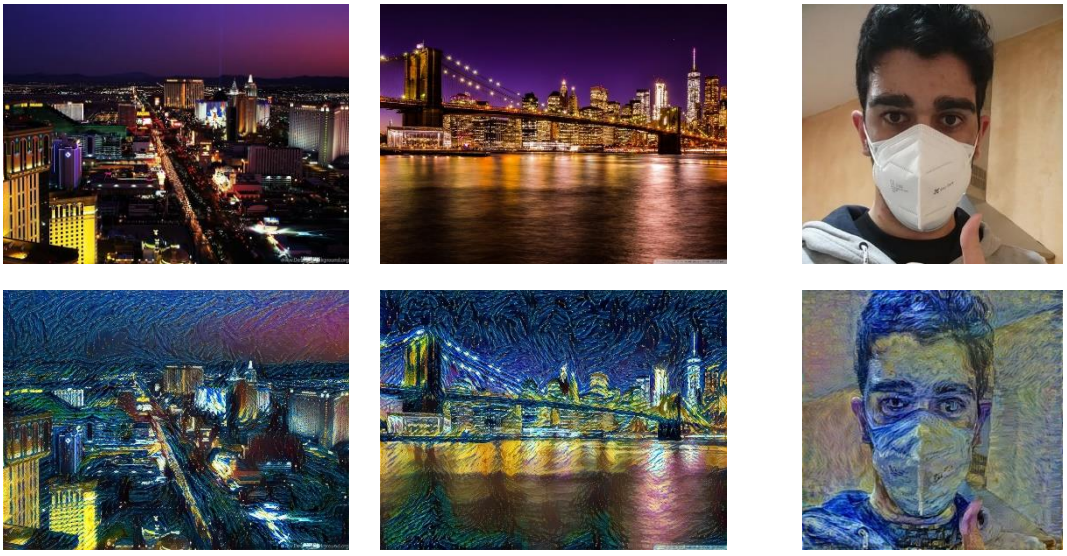
4 EJEMPLO PRÁCTICO

4.1 Neural Style Tranfer

En esta primera sección trataremos de mostrar los resultados obtenidos utilizando la técnica general.

En esta primera comparativa para todas las imágenes utilizaremos como referencia de estilo la obra *"La noche estrellada"* de *"Vincent van Gogh"*.

Ilustración 10 – Obra *"La noche estrellada"*

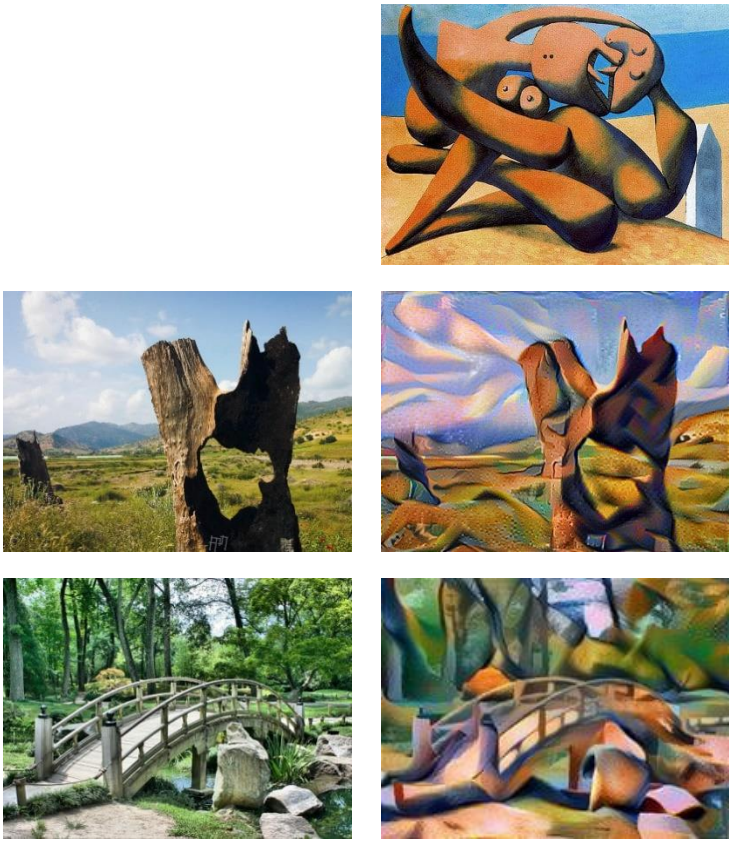


De las imágenes anteriores podemos sacar algunas conclusiones. Antes de comenzar, para todas ellas se ha utilizado el siguiente algoritmo disponible en este enlace.

Las dos primeras imágenes tienen muy alta resolución, mientras que la última es de baja calidad y esta realizada con una *"webcam"*. Como vemos, en los dos primeros casos se obtienen resultados bastante buenos, sin embargo, en la imagen creada a partir de mi foto, a pesar de que tomó 23 minutos de procesamiento (1200 iteraciones) parece bastante irreal.

4.2 Fast Neural Style Tranfer

Por otro lado, he hecho algunas otras pruebas con la variante del algoritmo que comentamos anteriormente, y estos han sido los resultados.



Como vemos en esta ocasión, a pesar de utilizar la técnica variante, los resultados son especialmente buenos, incluso podemos destacar el resultado de la última imagen. Como contra, en esta técnica, no podemos utilizar cualquier estilo, sino que para ello únicamente podemos utilizar aquel modelo preentrenado que tengamos.

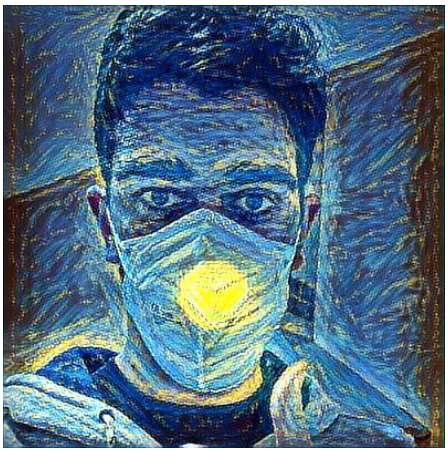


Ilustración 11 - Noche estrellada con algoritmo alternativo

Lo que me ha parecido interesante en esta ocasión (además de la rapidez con la que la genera), es que me ha gustado más el resultado que el que utiliza el algoritmo iterativo.

5 CONCLUSIONES

Una de las cosas que más me ha llamado la atención de este trabajo es que los resultados son bastante extremos, ya que o bien se obtienen composiciones que son muy logradas y files a la representación del estilo o bien podemos llevarnos la sorpresa de que el resultado final para nada es bueno y ejemplarizante de una buena composición como esperábamos.

Además, también nos gustaría señalar que la variante *"Fast NTS"* no debería pasar tan desapercibida ya que, aparte de consumir muchos menos recursos, sobre todo en tiempo (apenas unos segundos de procesamiento contra 17 minutos de media) obtiene unos resultados que se acercan, incluso superan en mi opinión en algunos casos, el rendimiento de la *"NTS"* tradicional.

Es un campo con mucho camino por avanzar por ahora, pero creo que tiene un gran margen de explotación, sobre todo en la industria de las redes sociales.