Algoritmos Evolutivos - Proyecto 2020: Exploración de espacio latente para generación de imágenes de caras reales

Benjamín Machín
Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
benjamin.machin@fing.edu.uy

I. Introducción

Este documento es el informe del proyecto de fin de curso presentado en la edición 2020 de Algoritmo Evolutivos. La idea del proyecto es combinar redes neuronales pre-entrenadas con algoritmos genéticos para lograr generar artificialmente imágenes de caras de personas reales.

En la siguiente sección se introducen algunos conceptos relevantes a este trabajo y se presenta el problema propuesto. Luego, en la sección III, se presenta el formato de representación de individuos y la función de fitness. En la sección IV se describen las primeras pruebas exploratorias de validación del proyecto, dónde además se detectan algunos puntos que deberán ser solucionados para poder continuar. En V se describen la estrategia y los operadores evolutivos. En VI se presentan detalles de implementación y reproducción de resultados. En la sección séptima se presentan algunas optimizaciones realizadas para acelerar el proceso. Las secciones VIII, IX y X, cubren las etapas de búsqueda de parámetros y evaluación del algoritmo genético. Finalmente, en las secciones XI y XII se analizan algunos resultados más y se presentan conclusiones e ideas de trabajo futuro, respectivamente.

II. Descripción del problema

A. Redes neuronales generativas antagónicas

Las redes neuronales generativas antagónicas (o GANs por su abreviatura en inglés) son un tipo de red neuronal profunda diseñada originalmente para generar imágenes realistas de forma sintética. Fueron introducidas por primera vez en [1], en 2014. Ya en ese primer trabajo, se realizaron pruebas con generación de imágenes de caras. Algunos de estos ejemplos se muestran en la figura 1. Desde entonces, las GANs se han convertido en un área de especialización en si misma, dando origen a una considerable cantidad de artículos de investigación y aplicaciones prácticas. En particular, el entrenamiento de GANs para generación de imágenes de caras ha avanzado mucho, alcanzando resultados altamente realistas, como se muestra en la figura 2.

La arquitectura de las GANs se basa en la idea de poner a dos subredes a competir durante el entrenamiento. Una subred denominada "generador" se encarga de generar las imágenes, y la otra, denominada "discriminador" intenta distinguir imágenes reales de las generadas artificialmente. Durante el proceso de entrenamiento, la subred generadora se vuelve tan buena en su tarea, que la subred discriminadora deja de poder detectar las imágenes artificiales. Una vez concluido este proceso, se puede utilizar la subred generadora para obtener nuevas imágenes, partiendo en general, de un vector de números reales aleatorios.

Fig. 1. Ejemplos de caras generadas artificialmente presentadas en [1]

B. Redes neuronales para reconocimiento facial

Por otro lado, también en los últimos años ha habido un gran avance en el desarrollo de redes neuronales profundas para reconocimiento facial. Un trabajo notable en esta área es el de [2], donde se propone obtener una representación vectorial de las imágenes de las caras, de forma de poder clusterizarlas según una medida de distancia. Estos vectores, comúnmente llamados "embeddings", codifican la información necesaria para distinguir personas entre sí, y son invariantes a escala, expresión facial, oclusiones leves, y hasta incluso edad, entre otras características similares.

C. Propuesta de trabajo

Dada una o varias imágenes de una cara de una persona real, se propone generar nuevas imágenes de su cara de forma artificial. Partiendo de señales de ruido aleatorio, la idea es utilizar una GAN pre-entrenada como la presentada recientemente en [3] para generar las caras artificiales correspondientes al ruido, y luego, utilizando otra red pre-entrenada para reconocimiento facial generar los vectores característicos de cada cara y medir su distancia al vector generado de la misma forma de la cara original. Utilizando algoritmos genéticos, se espera lograr evolucionar el ruido de entrada hasta lograr generar una (o varias) nuevas imágnes artificiales de la cara de la persona real.

De esta forma, el experimento consta de tres componentes principales:

1) red pre-entrenada generadora de caras, a partir de vectores de números aleatorios
2) red pre-entrenada que extrae embeddings de imágenes de caras
3) algoritmo genético que explora el espacio latente de la red generadora, tratando de minimizar la distancia de los embeddings de las imágenes generadas a los de la imagen de una cara objetivo

De esta idea surgen algunos interrogantes que interesa responder:

1) Dada una GAN entrenada para generar imágenes de caras y una imagen de una cara real, ¿existe un vector en el espacio latente que al ser procesado por la GAN sea capaz de generar una imagen de la persona que aparece en la imagen real?
2) dada una red entrenada para reconocimiento facial, ¿qué tan buena es su performance con imágenes de caras generadas artificialmente?

Fig. 2. Ejemplos de caras generadas artificialmente presentadas en [3]

III. Representación de individuos y función de fitness

En esta sección se presenta la codificación de individuos y la función de fitness, aspectos básicos necesarios para las primeras pruebas.

A. Representación de individuos

En este caso, la representación de los individuos es naturalmente el vector de entrada a la red generadora de imágenes de caras, que es simplemente un vector de G números reales, donde G estará dado por la arquitectura de la red generadora a utilizar (en particular, las dimensiones de su capa de entrada). En general, los componentes de estos vectores son centrados en cero y escalados. Esto deberá ser considerado al momento de elegir las estrategias de cruzamiento y mutación.

B. Función de costo

El fitness de cada individuo queda dado por la distancia de los embeddings de la imagen de la cara generada a partir del fenotipo, a los embeddings de la imagen de la cara objetivo. Al comienzo de la ejecución del algoritmo, se hará la extracción de los embeddings de la cara objetivo y se almacenarán para su posterior uso al evaluar el fitness de los individuos creados durante el proceso de evolución. Para calcular el fitness de un individuo, se realizan los siguientes pasos:

1) Se genera la imagen a partir de su fenotipo, utilizando la GAN.
2) Se extraen los embeddings de la imagen generada en el paso anterior utilizando la red de reconocimiento facial.
3) Se calcula la distancia euclideana entre los embeddings del paso anterior y los de la imagen de la cara objetivo, almacenados al comienzo de la ejecución.

IV. Pruebas exploratorias

Para validar la idea y mitigar posibles riesgos, se realizaron algunas pruebas exploratorias detalladas a continuación.

En primer lugar, se eligieron las redes pre-entrenadas necesarias. Para la generacón de caras se utilizó un modelo disponible en el repositorio asociado a [3] (https://github.com/NVlabs/stylegan2). Nos referiremos a este modelo como StyleGAN2. Para la extracción de embeddings de imágenes de caras se utilizó la implementación de [2] de David Sandberg, disponible en https://github.com/davidsandberg/facenet, al que nos referiremos como Facenet. Adicionalmente, se intentó utilizar un modelo de detección de caras para aplicar a todas las imágenes el mismo criterio de recorte, ya que Facenet asume un recorte ajustado de las imágenes de cara lo cual favorece su performance.

En las primeras pruebas, se eligieron imágenes objetivo generadas por la misma StyleGAN2, de forma de asegurarnos la existencia de una solución óptima. Luego se realizaron también pruebas con imágenes de caras reales.

En cuanto a operadores y parámetros con respecto al algoritmo evolutivo, se tomaron las siguientes decisiones:

- Cruzamiento de dos puntos.
- Mutación gaussiana.

- Selección por torneos.
- Estrategia de evolución $\mu + \lambda$ con $\mu = \lambda = 50\%$.
- Poblaciones pequeñas de no más de 40 individuos.
- Cantidades de generaciones acotadas a no más de 30 generaciones.

Algunos resultados de las primeras pruebas exploratorias se pueden apreciar en la figura 3. Si bien los resultados parecen prometedores, se detectaron varios puntos sobre los cuales fue necesario trabajar. Durante la ejecución de las pruebas, se observaron tiempos de inferencia altos para la red generadora, en el orden de los tres segundos por imagen. Este es un factor importante que debe ser mejorado, para poder ejecutar el algoritmo evolutivo durante más generaciones, y con una población mayor. Además, el modelo de detección de caras utilizado falló en detectar caras en todos los ejemplos provenientes de StyleGAN2, por lo cual se decidió continuar con la extracción omitiendo este paso, posiblemente afectando negativamente la performance de Facenet y en consecuencia, de todo el proceso. Finalmente, se observó un comportamiento demasiado estable de los valores de fitness a lo largo de las ejecuciones, donde las mejoras sustanciales parecían ser producto casi exclusivamente de las mutaciones, lo cual evidencia la necesidad de mejorar varios aspectos del algoritmo genético. Todas las pruebas exploratorias fueron realizadas en un equipo portátil con un procesador Intel Core i7-6700HQ con una frecuencia de 2.60GHz, y memoria RAM de 16.0GB.

V. Estrategia y operadores evolutivos

Una vez concluidas las primeras pruebas, se procedió a trabajar sobre la inicialización, operadores de cruzamiento y evolución, estrategia de selección y el algoritmo evolutivo a utilizar.

A. Inicialización

La inicialización se realizó de forma aleatoria, muestreando 512 valores de una distribución normal estándar.

B. Mutación

Se exploraron diferentes estrategias de mutación de forma manual, observando los cambios en las imágenes de caras generadas. Se optó por un operador de mutación que introdujera cambios sin ser totalmente destructivo. Se eligió el operador de mutación gaussiana media 0 y desviación 1 con 0.1 de probabilidad de mutación de cada posición. En la figura 4 se muestran algunos ejemplos del efecto del operador.

C. Cruzamiento

De forma similar, también se ejecutaron pruebas exploratorias con diferentes operadores de cruzamiento, con el objetivo de encontrar un operador que preserve en los descendientes, rasgos faciales de ambos individuos progenitores. Un operador que parece cumplir con este requisito es el operador de BLX-α introducido en [4], diseñado para trabajar con individuos con valores reales. Se realizaron pruebas adicionales para encontrar un valor adecuado del parámetro α del operador. En la figura 5 se muestra un ejemplo de los cruzamientos obtenidos.

Fig. 3. Algunos ejemplos de resultados obtenidos durante las primeras pruebas exploratorias. En la columna izquierda se muestran las imágenes objetivo, mientras que en la columna derecha se despliegan las imágenes resultado de la exploración del espacio latente. Las primeras dos filas corresponden a imágenes de caras generadas por StyleGAN2, mientras que la tercera es una imagen de una cara real.

D. Selección

Se utilizó el método de selección por torneos de a tres individuos participantes por torneo.

E. Estrategia de evolución

Se utilizó un algoritmo genético simple, con diferentes probabilidades de cruzamiento y mutación, cuyos valores se analizaron durante el proceso de evaluación paramétrica.

VI. Implementación y reproducción de resultados

La solución descripta en el informe fue implementada utilizando la biblioteca DEAP en su versión 1.3.1 para Python 3.7, ejecutando sobre la plataforma Colab de Google (colab.research.google.com/), en su versión paga. Para generar las imágenes de caras se utilizó StyleGAN2 (presentado este año en [3], código disponible en https://github.com/NVlabs/stylegan2), y para obtener los embeddings de las caras generadas se utilizó el paquete keras-facenet (https://pypi.org/project/keras-facenet/), un wrapper de la implementación de David Sandberg (https://github.com/davidsandberg/facenet) del artículo [2]. Los detalles de como reproducir los resultados que se reportan a continuación están documentados junto con el código en el notebook adjunto, así como el conjunto de instancias utilizadas para las pruebas.

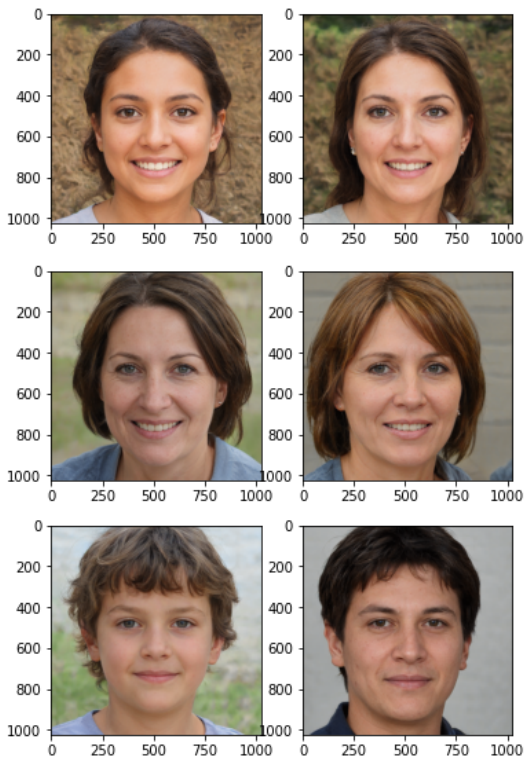

Fig. 4. Algunos ejemplos del efecto del operador de mutación sobre las imágenes generadas. A la izquierda la imagen original, a la derecha el resultado luego de la mutación.

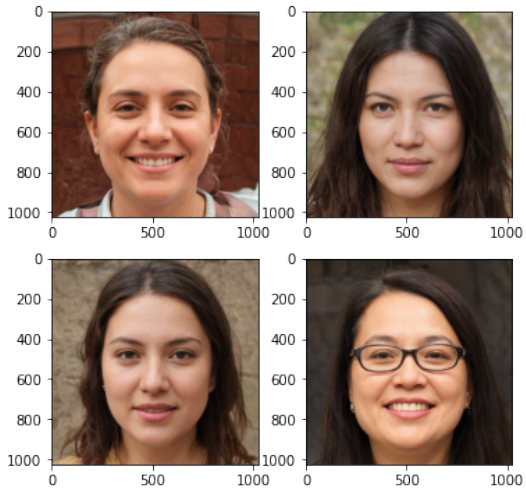

Fig. 5. Ejemplo de cruzamiento. Las dos imágenes superiores son las representaciones de los individuos progenitores, y las de abajo las de sus descendientes.

VII. Optimizaciones para reducción de tiempo de ejecución

Como fuera observado durante las pruebas iniciales, fue necesario trabajar sobre varios componentes de la solución para poder ejecutar el algoritmo evolutivo en tiempos razonables. Las optimizaciones más importantes realizadas fueron:

- cálculo del fitness de individuos en lotes.
- reducción de tamaño de imágenes generadas previa a los cálculos de fitness.
- reemplazo de algoritmo de detección de caras por una estimación de coordenadas fijas

En las siguientes subsecciones se describen estas optimizaciones.

A. *Cálculo de fitness en lotes*

Por defecto, los algoritmos genéticos provistos por DEAP requieren que se defina una función de fitness que toma de a un individuo por vez. Esto resulta muy ineficiente considerando que la arquitectura de las redes neuronales utilizadas pueden tomar entradas en lotes con tiempos de inferencia aproximadamente constantes. Por esta razón, se implementó una función de fitness por lotes, que toma un lote de individuos y calcula su fitness en forma paralela. Fue necesario modificar el algoritmo genético provisto por DEAP para soportar esta nueva función de fitness. El código de ambas modificaciones está documentado en el notebook entregado.

Si bien se mejoraron significativamente los tiempos de cálculo de fitness, la capacidad de paralelizar el cálculo está limitada por la cantidad de memoria disponible, acotada en particular por la la red generadora, que tiene altos requerimientos de memoria durante su inferencia. Por esta razón, los lotes recibidos por la función de fitness son subdivididos en sub-lotes de tamaño 20 para la red generadora.

B. *Reducción de tamaño de imágenes*

Considerando que las imágenes generadas por StyleGAN2 son de alta resolución (1024 píxeles de ancho y alto), y que la versión del modelo de Facenet utilizado fue entrenado con imágenes de caras de 160 píxeles de ancho y alto, se decidió hacer una reducción del tamaño de las imágenes generadas al momento de calcular el fitness. Esto mejoró considerablemente los tiempos de manipulación, detección de caras y de generación de embeddings con Facenet.

Notar que la reducción de tamaño no afecta la resolución de salida del algoritmo evolutivo, ya que sólo se aplica al momento de calcular el fitness de los individuos.

C. *Reemplazo de algoritmo de detección de caras*

Es necesario que la entrada a Facenet esté ajustada mediante un proceso de detección de caras; se debe recortar la imagen generada por StyleGAN2 de forma de que en la imagen sólo quede la cara lo más ajustada posible. Con este fin, se utilizó una biblioteca de detección de caras que sí funcionó con las imágenes generadas por StyleGAN2 (diferente de la utilizada en en los experimentos en IV), que debía ser aplicada a todas las caras generadas, una por una. Luego de las optimizaciones

anteriores, el tiempo de detección pasó a representar una parte importante del cálculo del fitness de cada generación, ya que cada imagen debía ser pasada por una red de detección. Se planteó la hipótesis de que en las imágenes de caras generadas por StyleGAN2, todas las caras aparecen aproximadamente en la misma posición, por lo cual debía ser posible utilizar coordenadas fijas para el recorte de las imágenes, sustituyendo al modelo de detección.

Para corroborar esto, se generaron 10.000 caras con StyleGAN2, para cada una de ellas se obtuvo el bounding box correspondiente con el modelo de detección, y finalmente se analizaron los resultados. Se obtuvieron coordenadas casi fijas, con un rango intercuartílico de ocho píxeles en promedio, lo cual no debería representar una dificultad para Facenet. Este experimento aparece en la sección "Detection analysis" del notebook entregado.

Con esta evidencia, se decidió reemplazar el modelo de detección por un recorte fijo según las coordenadas dadas por las medias del resultado de este experimento, mejorando aún un poco más los tiempos de cálculo de fitness, sin notorias consecuencias en la performance del algoritmo evolutivo.

Ahora ya con tiempos de ejecución más razonables que los encontrados durante las primeras pruebas, se procedió a realizar análisis de parámetros y evaluación final.

VIII. Análisis preliminar: cruzamiento, número de generaciones y tamaño de población

Antes de proceder al análisis de parámetros formal, y con el objetivo de reducir el espacio de búsqueda, se realizó un análisis informal preliminar para elegir parámetros del operador de cruzamiento, analizar el efecto de la cantidad de generaciones ejecutadas y el tamaño de población.

A. Cruzamiento

Se tomaron dos valores posibles para el parámetro α del operador de cruzamiento, se realizaron diez ejecuciones para cada valor para una única instancia, durante 100 generaciones, con una población de 100 individuos, y dejando fijos los valores de probabilidad de cruzamiento (0.6) y probabilidad de mutación (0.1). Los resultados se observan en la tabla I, donde también se calcula la media como medida de resumen y el rango intercuartílico como medida de dispersión. A partir de estos valores, se fija el valor de α en 0.2.

B. Tamaño de población

De forma similar, se realizaron pruebas para tres tamaños de población, fijando ahora el α del cruzamiento en 0.2, y dejando el resto de la configuración como en el caso anterior. En este caso, a medida que aumenta el tamaño de población, mejora la calidad de los resultados obtenidos (ver tabla II). Sin embargo, los tiempos se duplican en cada caso: 260 segundos en media para población de tamaño 100, 520 segundos para población de tamaño 200 y 1040 segundos para población de tamaño 400. Por esta razón, se decidió tomar el valor intermedio de 200 para el tamaño de la población.

TABLE I
ANÁLISIS PRELIMINAR: BLEND α

ID	Mín. ($\alpha = 0.2$)	Mín. ($\alpha = 0.5$)
1	0.690	0.790
2	0.660	0.640
3	0.700	0.690
4	0.740	0.710
5	0.670	0.820
6	0.720	0.750
7	0.760	0.760
8	0.620	0.780
9	0.810	0.730
10	0.840	0.800
Media	0.710	0.755
IQR	0.080	0.073

TABLE II
ANÁLISIS PRELIMINAR: TAMAÑO DE POBLACIÓN

ID	Mín. (pob=100)	Mín. (pob=200)	Mín. (pob=400)
1	0.890	0.660	0.570
2	0.600	0.710	0.610
3	0.840	0.730	0.650
4	0.710	0.650	0.690
5	0.740	0.720	0.670
6	0.720	0.720	0.620
7	0.750	0.740	0.760
8	0.710	0.580	0.670
9	0.760	0.730	0.600
10	0.770	0.710	0.720
Media	0.745	0.715	0.660
IQR	0.055	0.055	0.072

C. Número de generaciones

Finalmente, se repiten las pruebas pero ahora para diferentes cantidades de generaciones. Los resultados se observan en la tabla III. Además, en la figura 6 se muestra la evolución de los valores mínimos de fitness durante 500 generaciones para 10 ejecuciones independientes. Dado que no se observan mejoras sustanciales luego de 100 generaciones en casi ningún caso, y considerando el costo en tiempo de ejecución, se fija el valor de la cantidad de generaciones en 100.

TABLE III
ANÁLISIS PRELIMINAR: CANTIDAD DE GENERACIONES

ID	Mín. (gen=100)	Mín. (gen=200)	Mín. (gen=400)
1	0.750	0.740	0.700
2	0.740	0.720	0.590
3	0.780	0.790	0.790
4	0.680	0.590	0.750
5	0.680	0.670	0.780
6	0.630	0.720	0.670
7	0.710	0.740	0.760
8	0.760	0.640	0.650
9	0.720	0.820	0.710
10	0.830	0.760	0.780
Media	0.720	0.720	0.710
IQR	0.070	0.070	0.090

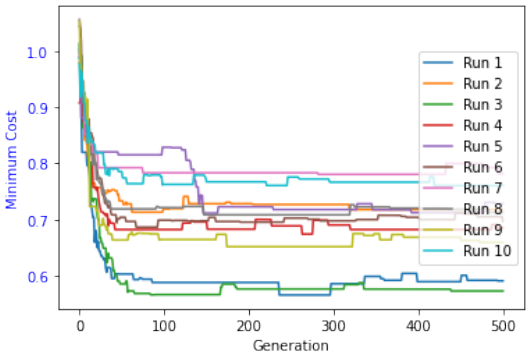

Fig. 6. Evolución de los valores mínimos de fitness en cada generación durante 500 generaciones, para 10 ejecuciones independientes.

IX. Búsqueda de parámetros

Contando con los resultados de los experimentos anteriores, se procedió a realizar una búsqueda de valores para las probabilidades de cruzamiento y mutación, para tres instancias distintas, ejecutando 30 veces cada experimento. Se tomaron tres valores de probabilidad de cruzamiento (0.6, 0.75 y 0.9) y tres valores para la probabilidad de mutación (0.1, 0.01 y 0.001), dando un total de 810 ejecuciones. Para el ajuste de parámetros, se eligieron dos caras reales y una cara generada por StyleGAN2, como un caso especial para el cual se sabe que hay una solución perfecta. La figura 7 muestra las instancias utilizadas. En las tablas IV, V y VI se detallan para cada combinación de parámetros, el p-value resultante de un test de normalidad Kolmogorov-Smirnov (todos los resultados tienen distribución normal), promedio y desviación estándar del fitness en las 30 ejecuciones y el rango de cada algoritmo para la instancia en particular.

Observando las tablas se concluye que los mejores resultados se obtienen cuando la probabilidad de cruzamiento es de 0.75. Para las tres combinaciones con los valores de probabilidad de mutación, se realizaron tests ANOVA, resultando en que las distribuciones de cualquiera de las tres combinaciones no son significativamente distintas entre si, para las tres instancias. Por lo cual, da lo mismo tomar cualquiera de las tres combinaciones. Esto sugiere que el efecto de la mutación no parece ser muy relevante para el proceso de evolución, aspecto que deberá investigarse en el trabajo futuro. Se eligió la combinación con probabilidad de mutación 0.01.

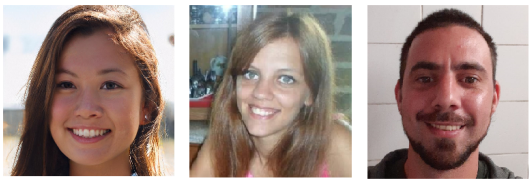

Fig. 7. Imágenes utilizadas durante la evaluación parámetrica: “gan 1”, “mujer 1” y “hombre 1”.

TABLE IV
Resultados de eval. paramétrica para la instancia “gan 1”

ID	cxpb	mutpb	*p-value KS*	*Promedio*	*STD*	*Rango*
1	0.6	0.1	0.416	0.420	0.051	5
2	0.6	0.01	0.647	0.455	0.064	9
3	0.6	0.001	0.442	0.454	0.072	8
4	0.75	0.1	0.327	0.407	0.040	3
5	0.75	0.01	0.462	0.396	0.036	1
6	0.75	0.001	0.184	0.403	0.050	2
7	0.9	0.1	0.391	0.431	0.047	7
8	0.9	0.01	0.253	0.426	0.052	6
9	0.9	0.001	0.930	0.414	0.058	4

TABLE V
Resultados de eval. paramétrica para la instancia “hombre 1”

ID	cxpb	mutpb	*p-value KS*	*Promedio*	*STD*	*Rango*
1	0.6	0.1	0.66255	0.475	0.049	4
2	0.6	0.01	0.15594	0.485	0.058	8
3	0.6	0.001	0.44567	0.519	0.056	9
4	0.75	0.1	0.76714	0.473	0.043	3
5	0.75	0.01	0.86157	0.465	0.042	1
6	0.75	0.001	0.78796	0.466	0.034	2
7	0.9	0.1	0.53227	0.477	0.032	6
8	0.9	0.01	0.67818	0.483	0.040	7
9	0.9	0.001	0.73313	0.476	0.038	5

X. Validación

Finalmente, se realizaron pruebas de validación sobre otras tres instancias: otra imagen generada por StyleGAN2, y otras dos imágenes de personas reales. Se utilizaron las configuraciones encontradas en los experimentos anteriores, y se realizaron también 30 ejecuciones para cada instancia. Se reportan las métricas de distancia y de tiempos de ejecución, cada una con sus tests de normalidad Kolmogorov-Smirnov como en el caso de la búsqueda de paramétros. Los resultados numéricos se muestran en la tabla VII, en dónde podemos observar también esta vez que las distribuciones de los resultados son normales, y que el algoritmo resulta razonablemente robusto. En la figura 8 se muestran las instancias junto con las mejores soluciones encontradas para cada una de ellas durante la validación.

XI. Otros resultados

En esta sección se discuten algunos ejemplos más, de soluciones encontradas durante las pruebas, desplegados en

TABLE VI
Resultados de eval. paramétrica para la instancia “mujer 1”

ID	cxpb	mutpb	*p-value KS*	*Promedio*	*STD*	*Rango*
1	0.6	0.1	0.89641	0.586	0.077	5
2	0.6	0.01	0.26502	0.636	0.100	9
3	0.6	0.001	0.98071	0.609	0.074	8
4	0.75	0.1	0.28833	0.573	0.066	2
5	0.75	0.01	0.69731	0.569	0.060	1
6	0.75	0.001	0.92763	0.575	0.080	3
7	0.9	0.1	0.45925	0.590	0.071	6
8	0.9	0.01	0.84681	0.576	0.063	4
9	0.9	0.001	0.93754	0.604	0.078	7

TABLE VII
RESULTADOS DE VALIDACIÓN

Instancia	*Distancia mín.*	*p-value KS*	*Distancia promedio*	*Distancia STD*	*p-value KS (tiempo)*	*Tiempo promedio*	*Tiempo STD*
Gan 2	0.350	0.547	0.453	0.041	0.505	625.202	7.728
Mujer 2	0.550	0.614	0.655	0.049	0.833	642.153	5.983
Mujer 3	0.420	0.789	0.495	0.041	0.991	904.768	5.540

Fig. 8. Arriba: imágenes utilizadas durante la validación: "gan 2", "mujer 2" y "mujer 3". Abajo: imágenes generadas a partir de las soluciones encontradas por el algoritmo genético durante la validación.

la figura 9. Por un lado se observa que Facenet es robusto a variaciones en atributos como la edad, pose, expresión facial, y oclusiones parciales, como el uso de lentes, incluso para imágenes artificiales. Es posible que estas característsitcas estén presentes desde la primer generación de individuos y que como Facenet funciona bien a pesar de todas estas posibles variantes, no sea necesario para el algoritmo evolutivo modificarlas para avanzar en la optimización. Esto también demuestra la alta calidad de las imágenes generadas por StyleGAN2. Sin embargo, se observan también algunas imágenes con artefactos extraños característicos de versiones anteriores de StyleGAN, lo cual evidencia que estos problemas no han sido totalmente resueltos en StyleGAN2, y también dejan ver formas posibles de atacar sistemas de reconocimiento facial basados en Facenet.

Más allá de lo mencionado, en cualquier caso los resultados parecen ser aceptables al ojo humano.

XII. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo se exploraron las capacidades modelos de aprendizaje profundo combinados con algoritmos genéticos para generar de forma artificial caras de personas reales. Se comprobó la existencia de vectores en el espacio latente de StyleGAN2 a partir de los cuales se pueden generar imágenes con caras cuyas distancias según Facenet son inferiores a los umbrales usuales para tareas de reconocimiento facial, y se validó la utilidad de los algoritmos genéticos para encontrar esos vectores. Esto coincide además con con la percepción humana durante la inspección visual de los resultados. Por otro lado, durante los experimentos se detectaron algunas fallas en StyleGAN2, que deberían ser reportadas y estudiadas para entender su origen. También se realizaron cambios sobre la biblioteca DEAP que son posibles contribuciones a la comunidad para acelerar la ejecución en ciertos casos en los que calcular el fitness de los individuos pueda ser paralelizable.

Hay varios puntos sobre los cuales trabajar en el futuro. Por un lado, seguir mejorando los tiempos de ejecución puede ser útil para poder trabajar con tamaños de población mayores, un aspecto que resultó provechoso durante el análisis pero que no fue explotado al máximo por limitaciones de hardware. También se debe realizar una revisión más profunda del operador de mutación, ya que durante las pruebas no pareció tener un impacto tangible en el proceso de evolución.

Interesa también introducir posibles variantes al problema, como por ejemplo optimizar la distancia a más de un vector objetivo, logrando que el resultado se asemeje a varias caras de la misma persona, o incluso de dos o más personas distintas. Este cambio no afectaría casi los tiempos de ejecución ya que el cálculo de distancia es insignificante en comparación al resto de las operaciones.

Otra dirección a explorar implica cambiar los componentes de la solución, en particular el modelo generador y el modelo guía, para poder explorar otros espacios latentes y generar artificialmente otras imágenes interesantes.

REFERENCES

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[2] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2015.7298682

[3] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," 2020.

[4] L. J. Eshelman and J. D. Schaffer, "Real-coded genetic algorithms and interval-schemata," in *Foundations of Genetic Algorithms*, ser. Foundations of Genetic Algorithms, L. D. WHITLEY, Ed. Elsevier, 1993, vol. 2, pp. 187 – 202. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780080948324500180

Fig. 9. Algunos resultados más para cuatro de las instancias utilizadas durante las pruebas. En la primer fila se encuentran las imágenes objetivo, y debajo de cada una de ellas, tres ejemplos de soluciones.