Red Generativa Adversaria para la creación de rostros de anime

Bergna Aguilar, Diego
Universidad Nacional de Ingeniería
Lima, Perú
diego.bergna.a@uni.pe

Mechan Osorio, Rodrigo
Universidad Nacional de Ingeniería
Lima, Perú
rodrigo.mechan.o@uni.pe

Paima Mijahuanca, Kevin
Universidad Nacional de Ingeniería
Lima, Perú
kpaimam@uni.pe

***Resumen*—Los animes han llegado a ser tantos que actualmente uno puede confundir el rostro de algún personaje de anime con alguno de otro anime y esto posiblemente debido a la falta de originalidad y creatividad o esfuerzo que le dedican los ilustradores. Todo esto por motivos como la tendencia de la forma de los rostros, o simplemente la gran cantidad de animes existentes, con lo que aumenta la probabilidad de que muchos de estos rostros se parezcan.**

El presente trabajo implementa una red generativa adversaria (GAN - por sus siglas en inglés), la cual es capaz de generar nuevos rostros de personajes al implementar dos redes que compiten entre sí, la primera red es una generadora de rostros de anime mientras que la segunda es una red que discrimina la salida que da la red generadora, logrando así resolver el problema de no tener idea de qué rostro dar a un nuevo personaje para algún anime.

Índice de Términos— GAN, red generadora, red discriminadora, rostros de anime.

I. Introducción

El anime es una parte de la industria del entretenimiento japonesa el cual genera una gran cantidad de ingresos, logrando alcanzar una cantidad de 19,9 mil millones de dólares en el año 2019, sin embargo el ámbito comercial del anime se ha visto envuelta en problemas por derechos debido a la similitud que tienen los personajes de distintas empresas pudiendo generar un pérdidas en demandas entre las mismas.

Con ello como problemática, entre las posibles soluciones se encuentra la de idear un modelo de inteligencia artificial que pueda generar nuevos personajes de anime no existentes o en caso de que el personaje generado sea semejante, sea posible modificar parámetros que varíen las características del personaje generado.

Este proyecto se centra en la realización de un modelo de redes generativas adversarias con el fin de generar rostros de personajes de anime.

I-A. Objetivos

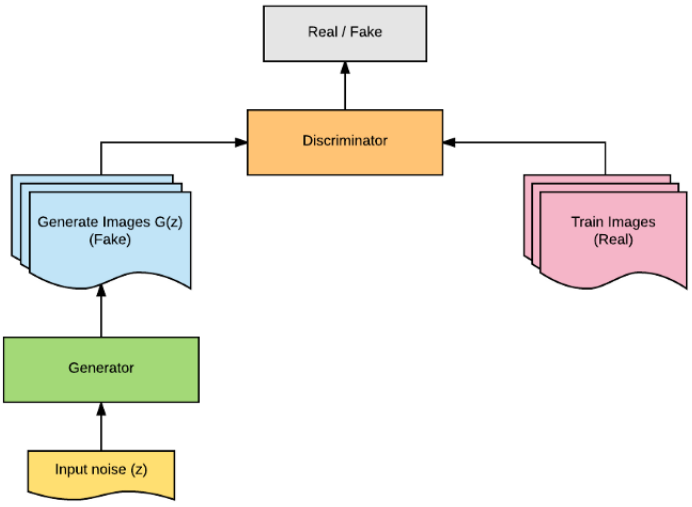
- Construir una red generativa adversaria para la generación de rostros de anime.

I-A1. Objetivos específicos:

- Implementación de la red generativa usando como referencia StyleGAN y DCGAN.
- Implementación de la red discriminadora.

II. GAN's

Las Redes Generativas Adversarias [2], mejor conocidas como GAN's por sus siglas en ingles, fueron introducidas para resolver la problemática de la complejidad de cálculo que tenían los modelos generativos profundos [1]. En el cual el modelo generativo(G) se enfrenta a un adversario, un modelo discriminatorio(D) el cual aprende a determinar si la muestra pertenece o no a los datos. Una analogía es que nuestro modelo generativo sea un falsificador de monedas mientras que nuestro modelo discriminatorio es la policía que intenta detectar esta moneda falsa.



Este proceso se repetirá hasta que nuestro modelo discriminatorio sea incapaz de detectar que la muestra creada por nuestro modelo generativo es falso.

III. Estado del Arte

Los GAN's están mejorando de tal manera que suplan ciertas limitaciones que se tenían en un principio. En esta sección veremos las distintas variantes que fueron creadas.

III-A. Conditional Generative Adversarial Network (cGAN)

CGAN propuesto en el articulo del mismo nombre [5], en donde se presento un modelo en el cual los GAN's

puede extenderse a un modelo en donde tanto el modelo generador como el modelo discriminatorio son condicionados por alguna información $\boldsymbol{y}$. Pudiendo realizar un condicionamiento introduciendo $\boldsymbol{y}$ a ambos modelos como capa de entrada adicional. Esto se comprobó en el articulo usando imagenes MNIST condicionadas a sus etiqueta de clases, codificadas como vector one-hot

III-B. Deep Convolutional Generative Adversarial Network (DCGAN)

DCGAN es un modelo que fue publicado en el articulo **Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks** [7], en donde se utilizan las redes neuronales convolucionales para el aprendizaje no supervisado, planteando ciertas restricciones de tal manera que se hagan estable al momento de ser entrenados para la mayoría de los escenarios.

Pasos para una DCGAN estable:

- Reemplazar las capas de agrupación por convoluciones estriadas(modelo discriminatorio) y convoluciones parcialmente estriadas(modelo generativo).
- Usar normalización batch [3] en el modelo generativo y discriminatorio.
- Eliminar completamente todas las capas convolucionales conectadas.
- Usar la función de activación ReLU [6] en el modelo generador para todas las capas excepto para la salida, el cual usa la función hiperbólica.
- Usar la función de activación LeakyReLU [8] en el modelo descriminatorio para todas las capas.

III-C. Style-Based Generative Adversarial Network (StyleGAN)

StyleGAN es un modelo que trae una arquitectura alternativa para las redes generativas. Este modelo se presento en **A Style-Based Generator Architecture for Generative Adversarial Networks** [4]. El objetivo de esta clase de redes es la identificación de atributos o estilos muy específicos en las imágenes de manera automática y no supervisada.

Este modelo consta de la separación de la imagen en 3 capas:

- Coarse Styles: Encargada de la pose, el cabello, la forma del rostro
- Middle Styles: Encargada de las características faciales
- Fine Styles: Encargada de la coloración de la imagen

Por otro lado, este modelo separa variaciones o *ruido* de atributos mas específicos.

- Coarse Noise: Encargada de manejar el ruido de el cabello, el rizado.
- Fine Noise: Encargada de los detalles más finos, las texturas(arrugas, etc).
- No Noise: Encargada de verlas características sin ruido.

Una de las funcionalidades que influyen en los resultados es la selección del impacto con el cuál cada estilo es aplicado

- High Strength: Da una variación alta, genera imágenes *rotas*
- Low Strength: Da una variación reducida
- Negative Strength: Da de resultado un *antirostro*

IV. Metodología

Procederemos a detallar el planteamiento del desarrollo del proyecto en base a los objetivos establecidos.

Previo a la creación de un modelo, se debe analizar el conjunto de datos con los que se trabajará.

Para propósito de nuestro proyecto optamos por un dataset de rostros de anime. El dataset elegido es: Generating Anime Faces. El cual consta de un conjunto de 63500 archivos aproximadamente con rostros de anime genéricos y conocidos, muchos de ellos cuentan con artículos en el rostro e incluso algunos con rostros de colores fuera de lo común. Un defecto que encontramos en el dataset es que en su mayoría los rostros pertenecen a personajes femeninos, pero para fines prácticos no le daremos importancia por ahora.

IV-A. Implementación de la red generativa y la red discriminadora

IV-A1. Red generativa: Primero se deberá analizar la arquitectura que tendrá la red, como referencia se ha tenido algunas redes generativas de los modelos StyleGAN, DCGAN y otras.

Posterior a ello se planteará una arquitectura en función de los puntos buenos encontrados en las referencias intentando lograr una red estable que vaya de la mano con el discriminador.

Como entrada se tendrá tensores de ruido los cuales se someterán a diversos procesos como convoluciones, normalizaciones entre otros para dar lugar a una imagen la cual será evaluada por la red discriminadora y esta devolverá una puntuación en función al criterio de la última para decirle a la red generadora si va por buen camino al crear las imágenes o no.

Pese a que se menciona que se generará una imagen, realmente se están generando un lote de imágenes y este mismo lote se evalúa en el discriminador.

IV-A2. Red discriminadora: La red discriminadora puede interpretarse como una red clasificadora que indicará la probabilidad que la imagen recibida pertenezca probablemente a la categoría objetivo, en este caso, la probabilidad de que la imagen, o lote de imágenes, recibida tenga las características de un rostro de anime.

Para el entrenamiento de nuestra red discriminadora, como entrada tendremos imágenes PNG con dimensiones 128x128px las cuales pasarán por una transformación para obtener su información en forma de tensores

IV-B. Entrenamiento

En las GAN's se deben entrenar a la vez ambos modelos, esto se debe a que si se entrenase por separado la red discriminadora de la generadora, una de ellas terminaría especializándose primero en su campo con lo que no daría oportunidad a la otra red de aprender de sus errores. Imagina que el discriminador sea muy bueno en su trabajo, siempre estaría seguro que la información que recibe por parte del generador es falsa por lo que al devolverle su puntuación a la red generadora, esta última no tendría dirección para empezar su aprendizaje por lo que estaría destinada a nunca aprender.

Sucede lo mismo con la red generadora, si esta se entrenase primero, sería tan buena que la red discriminadora nunca podría detectar si una imagen es tiene rostro de anime o no. Por lo que siempre categorizará lo que sea que envíe la red generadora como rostro de anime, y eso perjudica también a la red generadora pues podría darse el caso que sus imágenes empiecen a mutar y siga creyendo que hace un buen trabajo generando rostros de anime.

Se realizarán pruebas para distintos intervalos de entrenamiento como 10, 50, 100 y 200 épocas de entrenamiento y observar cómo mejoran ambas redes al generar y distinguir entre lo que podría ser un rostro de anime y una imagen cualquiera.

Si se detecta que en algún momento no hay mejoras en el entrenamiento, se decidirá modificar los hiperparámetros o la arquitectura misma a fin de encontrar un mejor desempeño para ambas redes y obtener así mejores resultados.

V. Experimentación y resultados

En esta sección se mostrarán resultados obtenidos para experimentaciones realizadas.

V-A. Limpieza de datos

Lo primero en realizarse fue un análisis del dataset, de las 63500 imágenes muchas de ellas tuvieron dimensiones diferentes, por lo que fueron redimensionadas a imágenes en formato PNG de 128px x 128px.

Posterior a ello se realizó una búsqueda de duplicados y se encontraron alrededor de 21000 imágenes repetidas las cuales fueron eliminadas dejando solo un ejemplar de las mismas para obtener finalmente 42mil imágenes para el entrenamiento aproximadamente.

V-B. Definición del modelo de la red generativa

Como se observa en 1 se optó por tomar como referencia base la arquitectura DCGAN en conjunto de WGAN, construyendo un modelo que tome un vector de ruido, y mediante un conjunto de convoluciones transpuestas, normalización por lote, y funciones de activación intermedias como ReLU, LeackyReLU y Tanh.

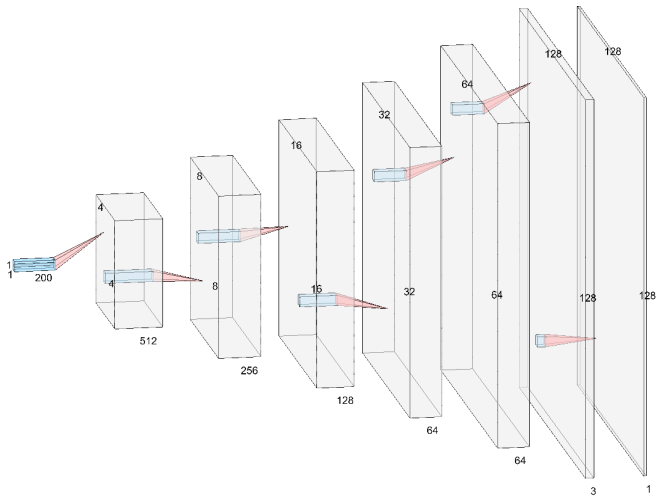


Figura 1. Arquitectura de red generativa

El modelo cuenta con 6 capas de convolución en donde las 5 capas interiores realizan un proceso de BatchNorm2D para la normalización de los datos de entrada, seguidas de una función de activación que buscará transmitir la información obtenida de cada convolución. Posterior a la convolución de la capa final, solo se realiza la función de activación Tanh para procesar la información final.

A diferencia de la red DCGAN se ha buscado experimentar tanto con los valores del kernel, stride y padding de cada capa de convolución sin buscar afectar el tamaño de salida de la misma.

V-C. Definición del modelo de la red discriminadora

Recordemos que una GAN se basa en el juego MinMax, en donde uno de los modelos busca reducir la diferencia entre las distribuciones de datos (generador), mientras que el otro modelo busca maximizar o resaltar las diferencias entre las distribuciones de los datos (discriminador). Con ello en mente se implementado una red discriminadora opuesta a la red generadora, es decir, que realice sus procesos de manera inversa para obtener la información como se observa en 2.

Este modelo toma como entrada un tensor bidimensional con la información de una imagen a la cual se le aplica un proceso de 6 convoluciones con el fin de obtener sus características y reducir el tamaño de la información, convolución tras convolución; aplicando tras cada una, una normalización y una función de activación.

Para nuestro modelo discriminador, se desarrolló como un modelo simple de clasificación binaria el cual indica si la información brindada por la red generadora coincide con la distribución de las imágenes que el modelo considera real.

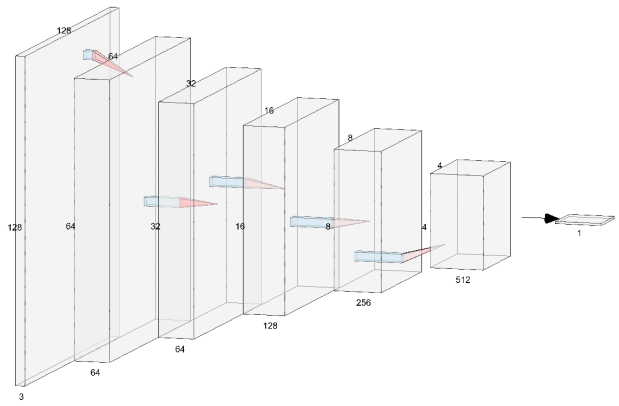


Figura 2. Arquitectura de red discriminadora

V-D. Complementos

Dataloader. Nuestro dataloader primero realiza una confirmación del tamaño de nuestra imagen y en caso sea diferente de 128 x 128px lo redimensiona a ese tamaño; luego las imagenes se pasan a tensores y las normalizamos. Estos se toman en pequeños batch para ingresar al entrenamiento.

Carga de datos. Para poder realizar la carga e iteración con los datos se usa nuestro Dataloader.

Noise Generator. Generará un vector aleatorio el cual sera del tamaño del batch que toma el dataloader en una época el cual tendrá una profundidad de 200.

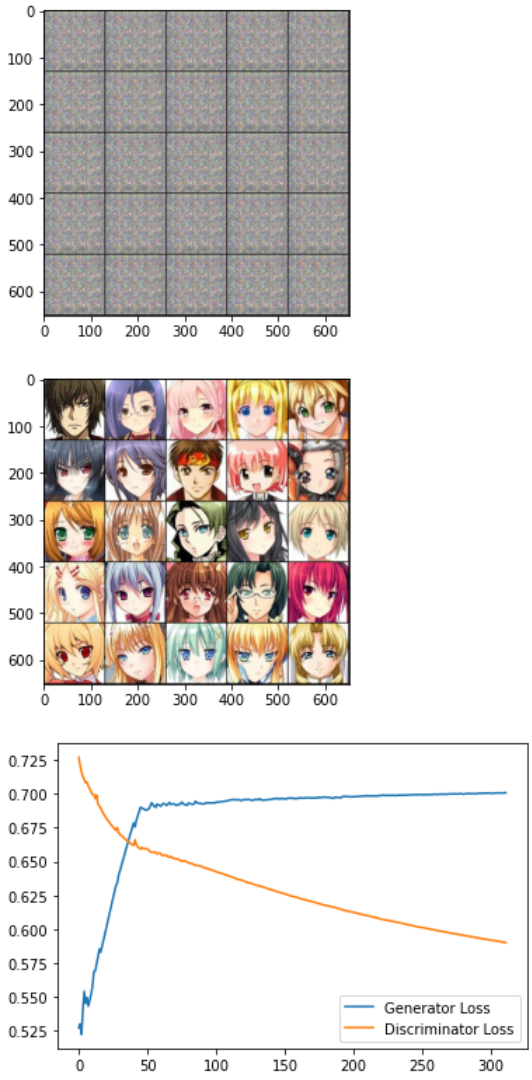


Figura 3. En nuestra novena época vemos un funcionamiento aparentemente correcto

V-E. Entrenamiento

Durante las primeras pruebas, se observó que la arquitectura preliminar que construimos no funcionaba pues no brindaba una generación de imágenes por lo que se decidió replantear la arquitectura muchas veces más. Durante el quinto intento se obtuvo una arquitectura que aparentaba funcionar por lo que decidimos seguir con su implementación. Durante el entrenamiento se observó el alto consumo de recursos del sistema, lo cual llevó al colapso del modelo. 3 4

Durante el entrenamiento se observó un límite en el valor de pérdida convergía a 0.7 y la puntuación de pérdida para el modelo discriminador se reducía llegando finalmente a colapsar cuando se acabó la memoria de trabajo.

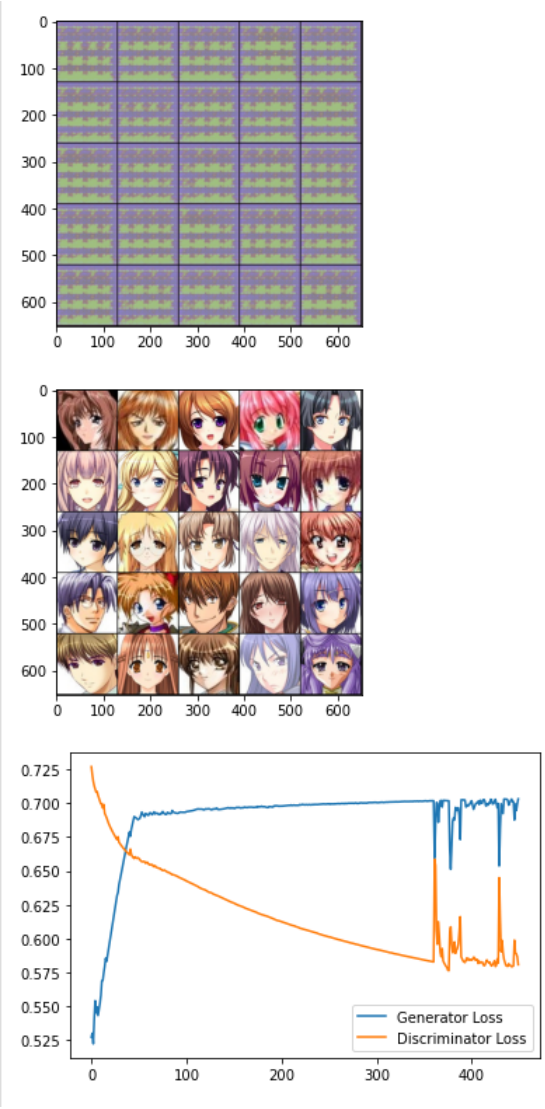


Figura 4. Llegamos al limite de capacidad de memoria y colapsó del modelo

VI. Discusión de resultados

En esta sección se hablará de los resultados y presentando argumentos que respalden los mismos.

VII. Conclusiones

Referencias

[1] Ludovic Arnold and Yann Ollivier. Layer-wise learning of deep generative models, 2013.
[2] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.
[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.
[4] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019.
[5] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets, 2014.
[6] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.
[7] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.
[8] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network, 2015.