Universidad del Valle de Guatemala
Deep Learning y Sistemas Inteligentes
Sección 10

Laboratorio #6

GANs

21016 Javier Chavez
21085 Andres Quezada
21631 Mario Cristalitros

Práctica

Cambios realizados en el generador

- Cambio del tamaño del kernel: El kernel en las capas de transposición convolucional del generador fue cambiado de 3x3 a 5x5. Esto permite que el generador capture más características en cada paso de convolución, a pesar de usar menos capas, lo que reduce la profundidad total de la red.

- Reducción de la complejidad: utilizamos menos capas mientras se mantiene la calidad de las imágenes generadas. El generador ahora usa capas de transposición convolucionales (Conv2DTranspose) para incrementar el tamaño de las imágenes, pero con un número de parámetros más reducido.

- Uso de BatchNormalization y LeakyReLU: Estas capas se han utilizado de manera más estratégica para estabilizar el entrenamiento. BatchNormalization ayuda a acelerar el entrenamiento al normalizar las activaciones de las capas, mientras que LeakyReLU evita que el gradiente se apague, permitiendo que la red aprenda mejor sin agregar mucha complejidad computacional.

Cambios realizados en el discriminador

- Cambio del tamaño del kernel: Al igual que en el generador, cambiamos el tamaño del kernel en las capas convolucionales del discriminador de 3x3 a 5x5, lo que le permite detectar características más complejas de las imágenes generadas sin aumentar considerablemente el número de parámetros.

- Uso de Dropout: añadimos Dropout en el discriminador para evitar el sobreajuste. Dropout elimina aleatoriamente un porcentaje de las conexiones en cada paso de entrenamiento, lo que reduce la necesidad de agregar más capas o más unidades, haciendo que la red sea más ligera.

Justificación de los cambios

Estos cambios hacen que la red sea más eficiente computacionalmente debido a:

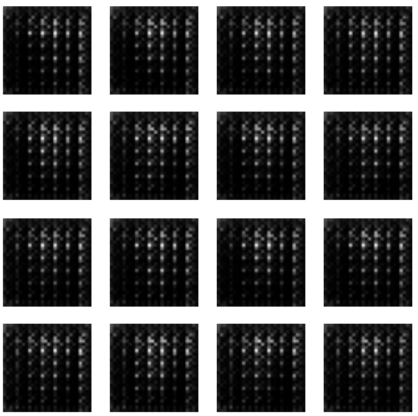
- Reducción de parámetros: El uso de kernels más grandes (5x5 en lugar de 3x3) en conjunto con BatchNormalization y Dropout permite reducir la profundidad de la red sin sacrificar la capacidad de aprendizaje. Esto disminuye el número total de parámetros entrenables y, por lo tanto, el tiempo de entrenamiento.

- Eficiencia en el entrenamiento: Al reducir el número de capas y parámetros, la red se entrena más rápido y consume menos memoria, lo que es crucial para aplicaciones donde los recursos de hardware son limitados.

- Estabilidad: El uso de BatchNormalization y LeakyReLU contribuye a un entrenamiento más estable, lo que reduce el número de épocas necesarias para obtener resultados aceptables.

Resultados cada 10 epochs

Época #1

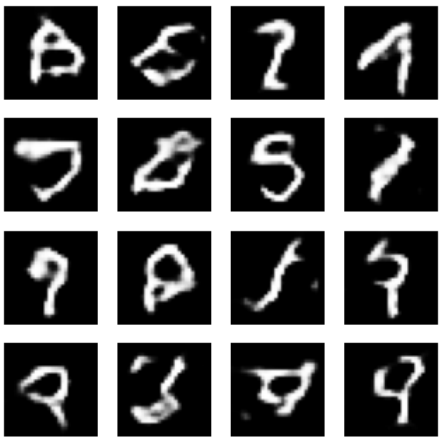


Época #10

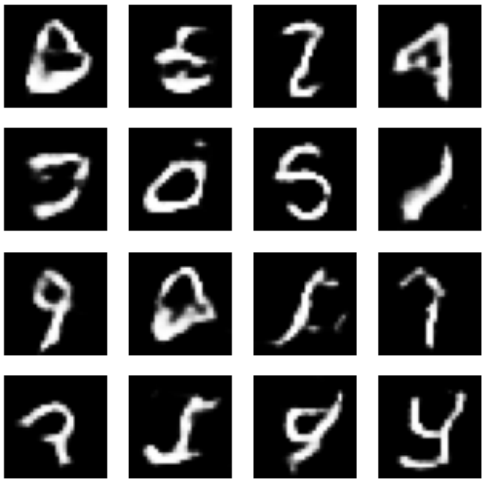
Época #20

Época #30

Época #40



Época #50



Descripción del aprendizaje

Época 1:

- En esta etapa inicial, las imágenes generadas tienen formas vagas que no se asemejan del todo a números definidos. Las formas parecen estar en proceso de construcción, pero no tienen mucha claridad. Esto es esperado al inicio, ya que el generador recién está comenzando a aprender.

Época 10:

- Para la época 10, ya se pueden distinguir algunos números en las imágenes generadas, pero todavía hay distorsiones. Aunque algunos dígitos comienzan a ser más claros, muchos números aún están deformados o incompletos.

Época 20:

- -En la época 20, las imágenes mejoran en calidad. Se pueden identificar varios dígitos, aunque algunos todavía tienen problemas en su forma. El generador parece estar aprendiendo mejor a generar formas numéricas, pero sigue siendo inconsistente.

Época 30:

- Las imágenes en la época 30 muestran un progreso considerable. Los números generados son más definidos y claros. Sin embargo, aún hay algunos que tienen ligeras distorsiones, pero en general, la calidad ha mejorado en comparación con las épocas anteriores.

Época 40:

- En esta etapa, la mayoría de los números son reconocibles. El generador ha aprendido a generar dígitos de manera bastante convincente. Aunque algunas formas todavía pueden verse borrosas, el rendimiento general es bastante bueno. Pero empieza también a perder la forma de algunos números, al volver más delgados los trazos.

Época 50:

- En la última época, los números parecen ya tener una forma definida, con trazos y detalles finos. La red parece haber alcanzado un nivel de estabilidad, donde ya solo está refinando las mismas formas. Sin embargo, no todas parecen números.

Teoría

1. Modifique las funciones de pérdida del generador y del discriminador para monitorear su avance durante el entrenamiento. (Grafíquelos)

- ¿Qué comportamiento se espera de las funciones de pérdida durante el entrenamiento?

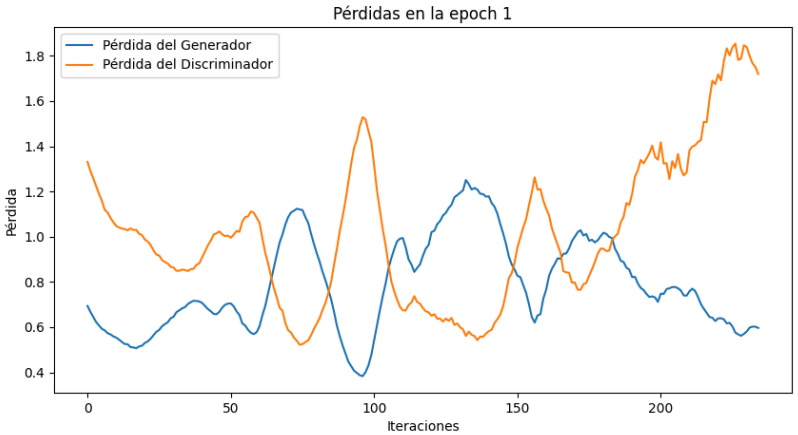
    - Al principio del entrenamiento, se espera que la pérdida del generador sea alta porque sus imágenes generadas no son capaces de engañar al discriminador. A medida que avanza debería disminuir, lo que indicaría que está mejorando en la generación de imágenes realistas que confunden al discriminador.

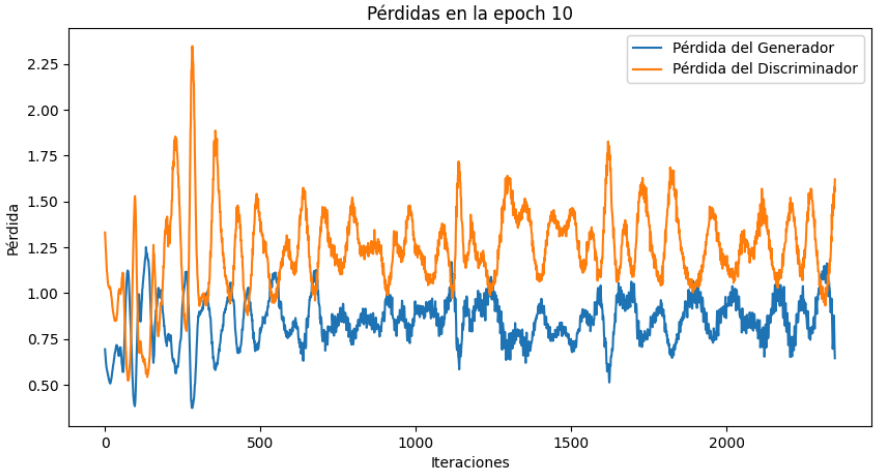
    - Inicialmente, la pérdida del discriminador debería ser baja porque es capaz de distinguir fácilmente las imágenes reales de las generadas. Conforme mejora, la pérdida del discriminador debería aumentar, ya que se vuelve más difícil diferenciar las imágenes generadas de las reales.

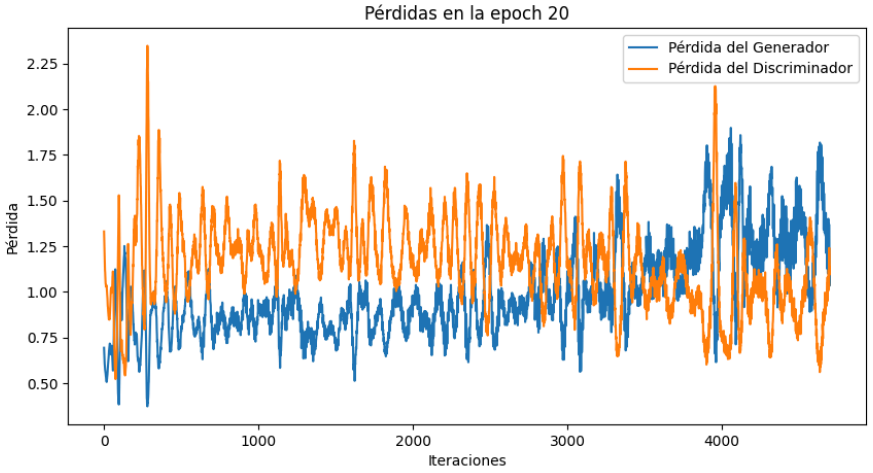
    - Idealmente, las pérdidas del generador y del discriminador alcanzarán un equilibrio en algún punto, lo que indica que ambos están aprendiendo de manera adecuada.

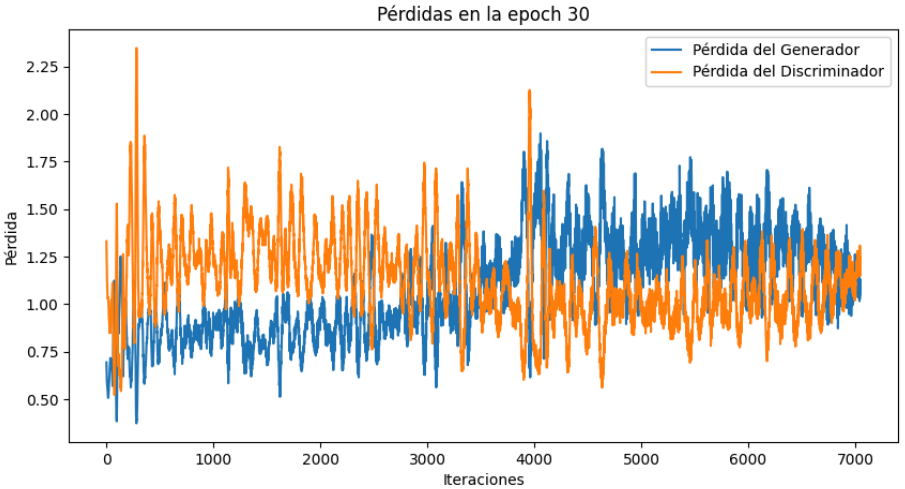
- ¿Qué comportamiento observa en la práctica?

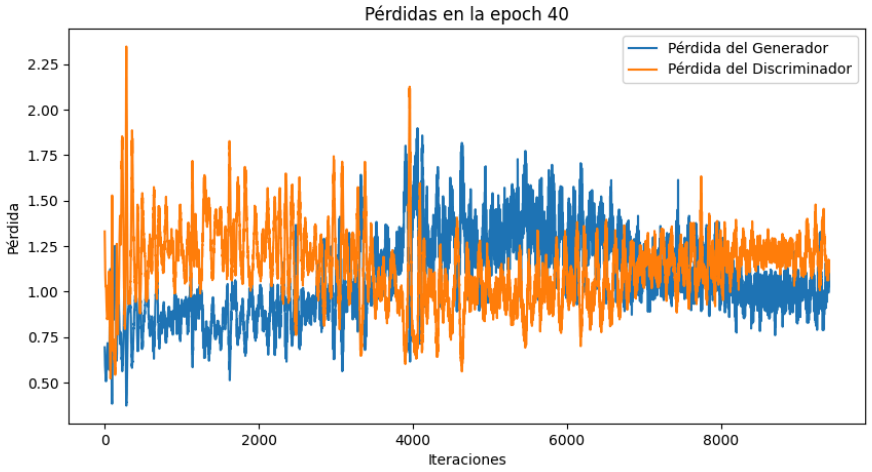
- A continuación podemos observar el comportamiento de las funciones de pérdida de ambos en cada época.

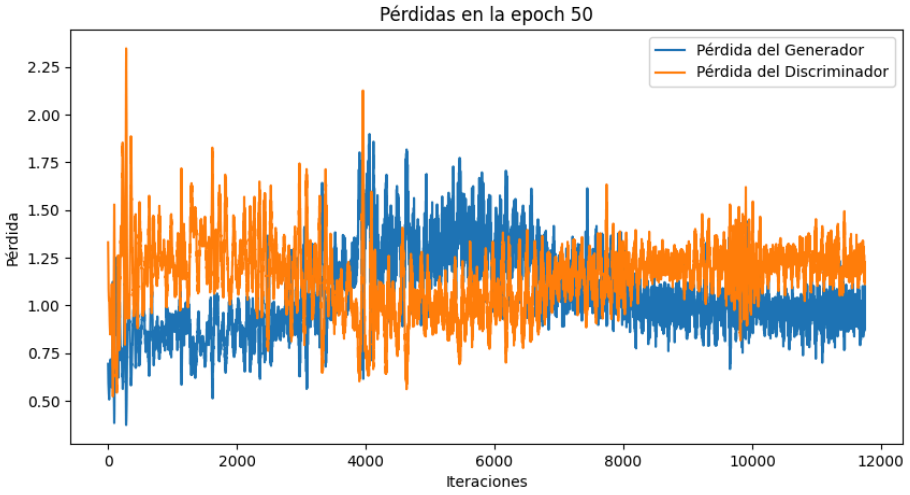



Pérdidas en la epoch 10
Pérdida del Generador
Pérdida del Discriminador
Pérdida
Iteraciones
0
500
1000
1500
2000
0.50
0.75
1.00
1.25
1.50
1.75
2.00
2.25


Pérdidas en la epoch 20
Pérdida del Generador
Pérdida del Discriminador
Pérdida
Iteraciones
0
1000
2000
3000
4000
0.50
0.75
1.00
1.25
1.50
1.75
2.00
2.25


Pérdidas en la epoch 30
Pérdida del Generador
Pérdida del Discriminador
Pérdida
Iteraciones
0
1000
2000
3000
4000
5000
6000
7000
0.50
0.75
1.00
1.25
1.50
1.75
2.00
2.25





Al analizar las gráficas podemos observar el siguiente comportamiento:

Época 1:

- Al principio, ambas pérdidas son muy variables, con oscilaciones grandes. La pérdida del discriminador es generalmente más alta, lo que significa que el generador está fallando en engañarlo, lo cual es esperado al inicio del entrenamiento. Lo interesante de esta gráfica es que podemos observar el fenómeno minimax, con los opuestos máximos y mínimos conforme se ajustan.

Épocas 10 a 20:

- En las épocas intermedias, se puede ver una disminución en las oscilaciones de las pérdidas. La pérdida del generador comienza a estabilizarse, indicando que está aprendiendo a generar imágenes más creíbles, aunque todavía no son perfectas. La pérdida del discriminador también fluctúa menos, lo que muestra que el modelo está mejorando en reconocer imágenes generadas.

Épocas 30 a 50:

- Las pérdidas del generador y del discriminador comienzan a estabilizarse más cerca de un valor constante, aunque aún se observan algunas oscilaciones. Esto indica que se está alcanzando cierto nivel de equilibrio, donde ambos modelos (generador y discriminador) están en competencia, y aunque la mejora no es lineal, ambos están aprendiendo.

Equilibrio en pérdidas:

- Se puede observar que no hay una convergencia perfecta hacia un punto específico, pero las oscilaciones se hacen más pequeñas con el tiempo. Esto es típico en el entrenamiento de una GAN, ya que los dos modelos están en constante "lucha". Las pérdidas fluctúan alrededor de valores similares, lo que puede indicar que el sistema ha alcanzado un punto en el que ambos están en equilibrio.

Conclusión:

En la práctica, la pérdida del generador disminuye progresivamente y se estabiliza a medida que el entrenamiento avanza, lo que significa que el generador está mejorando en su tarea. Por su parte, la pérdida del discriminador fluctúa, ya que intenta adaptarse a las imágenes generadas por el generador. A partir de las épocas 30 a 50, ambos parecen estar en equilibrio, lo que indica que el entrenamiento está progresando correctamente. Las fluctuaciones en las pérdidas son normales en GANs debido a la naturaleza competitiva del proceso de entrenamiento.

2. Tenga en cuenta la función de pérdida de una GAN y disecte sus elementos.

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

- Se representa como un juego de minimax entre el generador G y el discriminador D.

- D(x) es la probabilidad de que una imagen real *x* sea clasificada correctamente como falsa por el discriminador.

- D(G(z)) es la probabilidad de que una imagen generada G(z) sea clasificada correctamente como falsa por el discriminador.

- Pdata(x) es la distribución real de los datos

- Pz(z) es la distribución del ruido que se usa como entrada al generador

- La función de pérdida busca maximizar la capacidad del discriminador para clasificar correctamente imágenes reales y generadas, y al mismo tiempo minimizar la capacidad del generador para engañar al discriminador.

3. Explique con sus palabras, qué es un problema de minimax y por qué la función de pérdida de una GAN se considera uno.

- Un problema de minimax se refiere a una situación en la que dos “agentes” compiten entre sí, tratando de optimizar objetivos opuestos. El generador intenta minimizar la función de pérdida, es decir, generar imágenes que el discriminador clasifiquen incorrectamente como reales. Por otro lado, el discriminador intenta maximizar la función de pérdida, identificando correctamente las imágenes generadas como falsas.

- La función de pérdida de una GAN se considera un problema de minimax porque el Generador está intentando minimizar el valor de D(G(z)) (para que el discriminador clasifique imágenes generadas como reales). Mientras que el Discriminador intenta maximizar su capacidad para distinguir entre las imágenes reales *x* y las generadas G(z) manteniendo D(x) cercano a 1 y D(G(z)) cercano a 0. Entonces están compitiendo haciendo que ambos mejoren durante el entrenamiento.

Repositorio:

https://colab.research.google.com/drive/1q3ehiIWaXAXIxWE-F6Y6PSx6Ov9SXR9U?usp=sharing

https://github.com/JaniMariQuesiRami/DeLe-Lab6

Imágenes y gráficas:

https://drive.google.com/drive/folders/1z22EoXQAZBeCBc6dQhU28Hhyv1Dc9z4z?usp=sharing