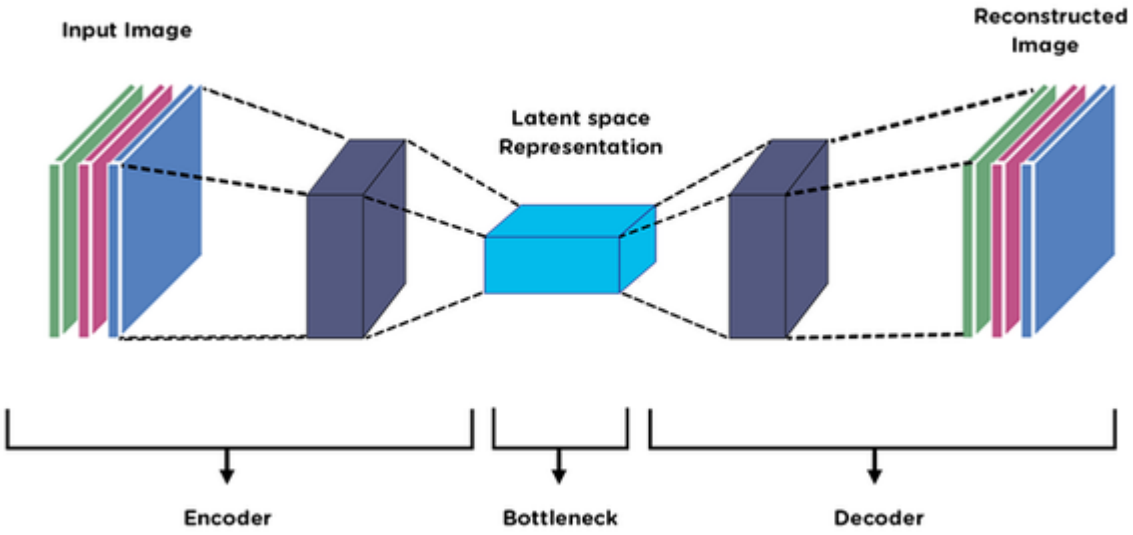
Práctica 2: Autoencoders



Luis Perera Pérez
Nicolás Trujillo Estévez

1. Introducción

En este proyecto se ha trabajado con redes neuronales de tipo autoencoder para abordar dos tareas fundamentales en el procesamiento de imágenes: eliminación de ruido y superresolución. Un autoencoder es un tipo de red neuronal diseñada para aprender una representación comprimida de los datos, y es ampliamente utilizada para tareas como la reducción de dimensionalidad, la reconstrucción de datos y la mejora de la calidad de las imágenes.

El objetivo principal del proyecto es:

- **Eliminación de ruido**: Usar un autoencoder para reducir el ruido en imágenes corruptas por ruido gaussiano o sal y pimienta.
- **Superresolución**: Modificar un autoencoder para aumentar la resolución de las imágenes y generar versiones de mayor calidad.

2. Objetivos

Los objetivos específicos del proyecto fueron los siguientes:

1. **Desarrollar un autoencoder para eliminación de ruido**:
    a. Añadir ruido gaussiano a las imágenes de entrada.
    b. Entrenar un autoencoder para que aprenda a reconstruir las imágenes originales y eliminar el ruido.
    c. Visualizar los resultados mostrando imágenes originales, ruidosas y reconstruidas.

2. **Desarrollar un autoencoder para superresolución de imágenes**:
    a. Modificar la arquitectura del autoencoder para que pueda trabajar con imágenes de 7x7 y 14x14 píxeles y generar imágenes de 28x28.
    b. Aumentar artificialmente la resolución de las imágenes sin perder calidad en la reconstrucción.
    c. Mostrar el impacto de la superresolución comparando las imágenes originales y las generadas.

3. Metodología

3.1. Eliminación de Ruido

Para abordar la tarea de eliminación de ruido, se implementó un autoencoder con una arquitectura basada en redes convolucionales. El autoencoder consta de dos partes principales: el codificador y el decodificador. El codificador comprime la imagen de entrada a una representación latente, mientras que el decodificador reconstruye la imagen a partir de esta representación comprimida.

Pasos seguidos:

1. **Adición de ruido**: Se implementaron dos tipos de ruido: ruido gaussiano y sal y pimienta. Este ruido se añadió a las imágenes originales de MNIST antes de pasarlas al autoencoder.
2. **Entrenamiento del modelo**: Se entrenó el autoencoder usando el conjunto de datos MNIST con las imágenes ruidosas como entradas y las imágenes originales como objetivo.
3. **Visualización de resultados**: Se generaron imágenes reconstruidas y se compararon con las imágenes originales y las ruidosas para evaluar la efectividad del autoencoder.

3.2. Superresolución de Imágenes

En la tarea de superresolución, el objetivo fue generar imágenes de mayor resolución a partir de imágenes de menor resolución, aumentando la calidad visual de las imágenes generadas.

Pasos seguidos:

1. **Arquitectura modificada**: Se diseñó un autoencoder con un decodificador modificado para aumentar la resolución de las imágenes. El modelo fue entrenado para tomar imágenes de tamaño 7x7 o 14x14 y generar imágenes de tamaño 28x28.
2. **Entrenamiento**: El modelo fue entrenado utilizando imágenes de MNIST, pero las imágenes de entrada fueron reescaladas a resoluciones menores antes de ser pasadas al modelo.
3. **Evaluación**: Se compararon las imágenes originales con las imágenes generadas para evaluar la capacidad del autoencoder para aumentar la resolución sin perder detalles.

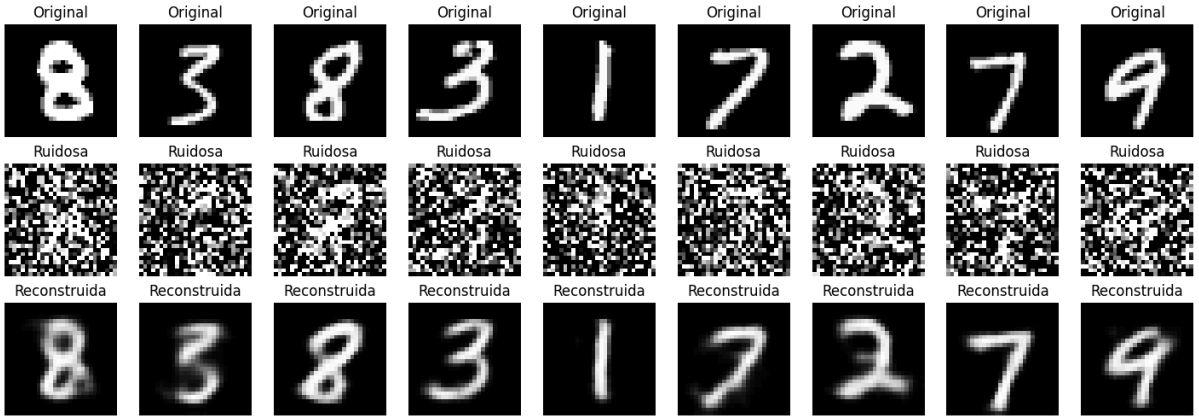
4. Resultados

4.1. Eliminación de Ruido

El autoencoder para eliminación de ruido demostró ser eficaz en la reducción del ruido gaussiano añadido a las imágenes. Al entrenar el modelo durante varias épocas, la red neuronal aprendió a reconstruir imágenes cercanas a las originales a partir de las imágenes ruidosas. La visualización de los resultados mostró claramente cómo las imágenes ruidosas se mejoraban con el tiempo.

Ejemplo de resultados visuales:

- La primera fila de las imágenes mostró las imágenes originales.
- La segunda fila mostró las imágenes con ruido añadido (ruido gaussiano).
- La tercera fila mostró las imágenes reconstruidas por el autoencoder.

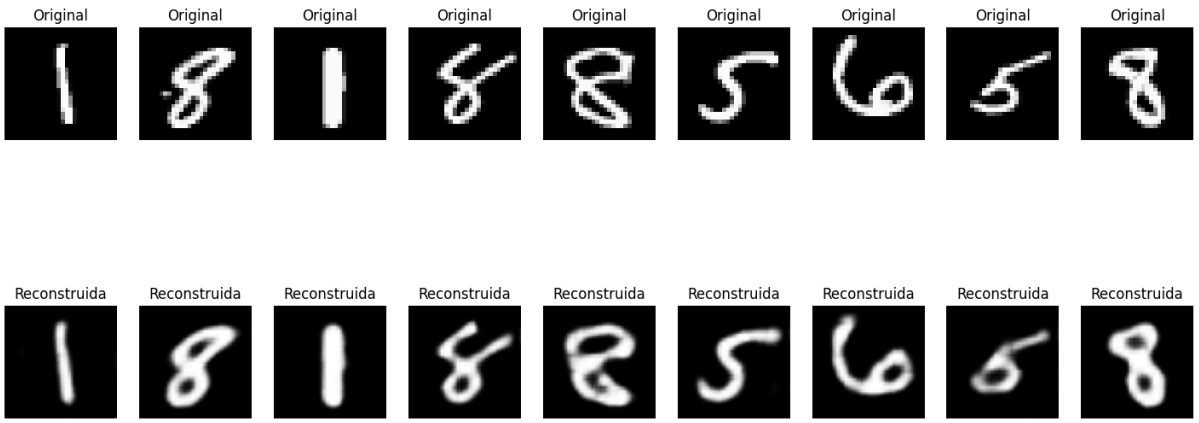


4.2. Superresolución

En la tarea de superresolución, el autoencoder fue capaz de aumentar la resolución de las imágenes de 7x7 o 14x14 a 28x28. Sin embargo, los resultados fueron menos nítidos en comparación con las imágenes originales, debido a la naturaleza del proceso de interpolación y la arquitectura del modelo.

Ejemplo de resultados visuales:

- La primera fila mostró las imágenes originales de 28x28 píxeles.
- La segunda fila mostró las imágenes de menor resolución (7x7 o 14x14).
- La tercera fila mostró las imágenes generadas por el autoencoder a mayor resolución (28x28).



5. Conclusión

En este proyecto, se implementaron y entrenaron redes neuronales de tipo autoencoder para abordar dos tareas fundamentales en el procesamiento de imágenes: la eliminación de ruido y la superresolución. Los resultados obtenidos demuestran la capacidad de los autoencoders para reconstruir imágenes de alta calidad a partir de datos ruidosos y para mejorar la resolución de imágenes de baja calidad.

En la tarea de eliminación de ruido, el autoencoder demostró ser altamente eficaz en la reducción de ruido gaussiano y ruido sal y pimienta, produciendo imágenes reconstruidas que se aproximan mucho a las originales. Esto destaca la utilidad de los autoencoders en escenarios donde las imágenes están deterioradas por factores externos.

Por otro lado, en la tarea de superresolución, aunque se logró aumentar la resolución de las imágenes de 7x7 o 14x14 a 28x28, las imágenes generadas mostraron cierta falta de nitidez en comparación con las originales. Este comportamiento puede atribuirse a las limitaciones de la arquitectura utilizada y a la complejidad del problema en sí.

6. Repositorio GitHub

El código y los archivos relacionados con este proyecto se encuentran disponibles en el repositorio de GitHub. El repositorio contiene todo el código necesario para la implementación de los autoencoders, así como los conjuntos de datos utilizados.

Enlace al repositorio: https://github.com/LuisPereraPerez/AA2_practicas