Informe de la Actividad 4: Implementación de RNN y LSTM para Series Temporales

Arquitectura del Modelo y Justificación de las Decisiones de Diseño

Asignatura: Redes Neuronales y Aprendizaje Profundo

Nombre del autor/a: Pablo Sánchez Arias

Introducción

En esta actividad, el objetivo principal fue implementar una Red Neuronal Recurrente (RNN) y una Long Short-Term Memory (LSTM) para procesar datos de series temporales. Además, se utilizó una RNN para la generación de texto, imitando el estilo de Shakespeare.

Arquitectura del Modelo

MLP para Predicción de Series Temporales

Se comenzó con la implementación de un Perceptrón Multicapa (MLP) para la predicción de series temporales. La arquitectura del MLP consistía en una capa de entrada `Flatten` seguida de varias capas `Dense`, finalizando con una capa de salida con una única neurona. La capa `Flatten` se utilizó para convertir los datos de entrada 2D en un formato 1D, adecuado para las capas `Dense`.

RNN para Predicción de Series Temporales

Posteriormente, se implementó una RNN simple utilizando la capa `SimpleRNN` de Keras. Esta red contenía una sola capa recurrente, lo que permitió procesar secuencias de cualquier longitud, aprovechando la capacidad de la RNN para mantener la información a lo largo del tiempo.

Deep RNN y LSTM

Para mejorar la capacidad de predicción, se implementaron arquitecturas más profundas:

- **Deep RNN:** Utilizando múltiples capas `SimpleRNN` con `return_sequences=True` en las capas intermedias para mantener la secuencia completa a través de las capas.
- **Deep LSTM:** Similar a la Deep RNN pero utilizando capas `LSTM`, las cuales son más efectivas en el manejo de dependencias a largo plazo debido a su estructura interna más compleja.

Desafíos Encontrados y Soluciones

Desafío 1: Implementación del Dataset

Inicialmente, se encontró un problema con el cálculo del tamaño del dataset (`dataset_size`), ya que se estaba utilizando `tokenizer.document_count`, lo que resultaba en un tamaño incorrecto. La solución fue cambiar a `len(shakespeare_text)`, asegurando así que el tamaño reflejaba el número total de caracteres en el texto de Shakespeare.

Desafío 2: Cálculo de `steps_per_epoch`

Otro desafío crítico fue asegurar que `steps_per_epoch` no fuera cero. Este fue el error que no pude solventar.

Precisión Final y Observaciones sobre el Rendimiento del Modelo

Debido a los errores encontrados en la Tarea 7 y la Tarea 8, no se logró completar la implementación de la RNN ni la generación de texto. Estos problemas se relacionaron principalmente con errores de iteración sobre objetos `NoneType` en el método `fit` del modelo, causados por un cálculo incorrecto de `steps_per_epoch`.

Observaciones Finales

A pesar de no haber obtenido los resultados finales esperados, se realizó un progreso significativo en la implementación de varias arquitecturas de redes neuronales para la predicción de series temporales. La experiencia obtenida al resolver problemas de dataset y entrenamiento será invaluable para futuros proyectos.

Conclusión

El proyecto permitió una comprensión más profunda de las redes neuronales recurrentes y su aplicación en series temporales y generación de texto. Aunque no se alcanzó una implementación completamente funcional, las soluciones propuestas y las lecciones aprendidas proporcionan una base sólida para futuras investigaciones y desarrollos en este campo.