ML Jedi Training I: Modelado de Funciones Sinusoidales con Redes Neuronales Y Neural ODEs.

Matias Nuñez

24 de agosto 2023

1 Modelado de Funciones Sinusoidales

1.1 Modelá una Función Sinusoidal con una Red Neuronal Feedforward

Contexto: Las funciones sinusoidales son un pilar en ciencias e ingeniería. Ahora, ¿qué pasa si intentamos modelar una de estas usando redes neuronales?

Objetivo: Tu misión es armar y entrenar un modelo usando una red neuronal para capturar una función sinusoidal $y(t) = \sin(t)$ con un toque de ruido.

Pasos:

1. Generá datos usando la función $y(t) = \sin(t)$ y metele un poco de ruido gaussiano para simular mediciones reales.
2. Dividí tus datos: una parte para entrenar y otra para probar.
3. Construí y entrená el modelo usando una red neuronal con, al menos, dos capas densas.
4. Evaluá en el set de prueba.

Tips:

- No te olvides de escalar tus datos antes de mandarlos a la red.
- Jugá con diferentes arquitecturas y cantidades de neuronas.
- Usá una función de pérdida como el error cuadrático medio, proba otras.
- Los datos de entrada para la red serán los valores de t y la salida será $y(t) = \sin(t)$ con ruido.

1.2 Modelá una Función Sinusoidal con una RNN

Contexto: Las funciones sinusoidales, especialmente cuando se agregan componentes temporales o ruido, pueden tener relaciones temporales complejas. Las RNNs, con su capacidad de "memoria", pueden ser herramientas poderosas para modelar estas relaciones.

Objetivo: Tu misión es construir y entrenar un modelo usando una RNN para capturar la función sinusoidal $y(t) = \sin(t)$ con ruido, y comparar su rendimiento con el modelo feedforward anterior.

Pasos:

1. Generá datos usando la función $y(t) = \sin(t)$ y añade un poco de ruido gaussiano.
2. Divide tus datos en secuencias para alimentar a la RNN.
3. Construye y entrena el modelo usando una RNN.
4. Evalúa en el conjunto de prueba y compara los resultados con el modelo feedforward.

Tips:

- Al preparar datos para una RNN, considera cómo estructuras los datos en secuencias. Por ejemplo, podrías usar los valores de t en los pasos de tiempo 1 a 10 para predecir $y(t)$ en el paso de tiempo 11.

- Experimenta con diferentes tipos de RNN, como LSTM o GRU, que pueden manejar mejor las dependencias temporales a largo plazo.
- La longitud de las secuencias y la cantidad de unidades en la RNN son hiperparámetros clave que pueden necesitar ajuste.

1.3 Modelá la Función Sinusoidal con una Neural ODE

Objetivo: Ahora, cambiamos de estrategia. Vamos a usar una Neural ODE para hacer el mismo laburo.

Pasos:

1. Usando los mismos datos que generaste antes, armá un modelo que use una Neural ODE.
2. Entrená este nuevo modelo.
3. Compará cómo rinde esta Neural ODE con respecto a la red neuronal común y corriente.

Tips:

- Pensá en cómo las derivadas con respecto al tiempo pueden ayudar a modelar la función sinusoidal.
- Las Neural ODEs pueden ser un poco más complicadas de entrenar, así que armate de paciencia y jugá con los parámetros.
- Los datos de entrada para la Neural ODE serán los valores de t y el valor inicial de la función. La salida será la función sinusoidal modelada a lo largo del tiempo.

1.4 Parte 3: Combinación de Red Neuronal y Neural ODE

Objetivo: Finalmente, ¿qué pasa si combinamos lo mejor de ambos mundos? Vamos a combinar una red neuronal tradicional con una Neural ODE.

Pasos:

1. Armá un modelo que empiece con una o dos capas densas y luego pase por una Neural ODE.
2. Entrená este modelo combinado.
3. Evaluá y compará su rendimiento contra los dos modelos anteriores.

Tips:

- Las capas densas iniciales pueden ayudar a la Neural ODE a concentrarse en las dinámicas más sutiles de la función.
- Este modelo tiene más parámetros, así que cuidado con el sobreajuste.
- Al combinar una red neuronal con una Neural ODE, los datos de entrada y salida serán similares a los de la Parte 1 y Parte 2, respectivamente. La red neuronal tradicional procesará los valores de t, y la Neural ODE modelará la función a lo largo del tiempo.

1.5 Parte 4: Modelá una Función Más Compleja

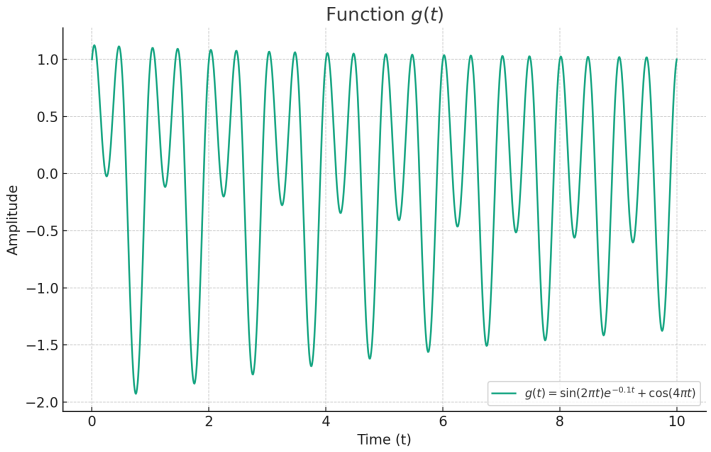

Figure 1: Gráfica de $g(t) = \sin(2\pi t)e^{-0.1t} + \cos(4\pi t)$

Contexto: Hasta ahora hemos trabajado con funciones sinusoidales básicas. Pero, ¿qué pasa si intentamos modelar algo más picante? Una función que

combine varias sinusoides y una envolvente exponencial podría ser interesante.

Objetivo: Tu misión es armar y entrenar un modelo para capturar la función $g(t) = \sin(2\pi t)e^{-0.1t} + \cos(4\pi t)$.

Pasos:

1. Idem caso anterior.

Tips:

- Dado que esta función es más compleja, es posible que necesites una arquitectura de red más profunda o más neuronas por capa.
- Experimentá con diferentes optimizadores y tasas de aprendizaje de Flux/Lux.
- Fijate que pasa si usas GPU o no, sobre todo para el caso de la neural ODE.
- Los datos de entrada para la red serán los valores de t y la salida será $g(t)$ con o sin ruido.