Reporte:
Generación Musical con IA

Jesús Alberto Aguirre Caro

25 de abril de 2025

Resumen

El objetivo del proyecto fue generar música a partir de un modelo basado en transformadores, agregarle una componente de prompt por texto y lograr correr estos modelos en un GPU AMD. De estos objetivos se lograron el primero y el tercero, es decir, se logró entrenar un modelo para generar música a partir de prompts musicales en formato MIDI y se logró correr todas sus variaciones en un AMD GPU. Se utilizó el dataset MAESTRO 2.0.0 para entrenar el modelo, el cual consiste de interpretaciones de piano de música clásica de compositores como Bach, Beethoven, Chopin, et al. Se utilizó PyTorch para correr el modelo, pues no fue posible hacer correr el modelo en Keras/TF. El parámetro más útil para generar buenas secuencias fue la temperatura, ya que al subirla ligeramente, daba más naturalidad. Uno de los objetivos futuros que se desea que se originen de este trabajo es la generación de un dataset de música latina; el objetivo sigue en pie y al haber trabajado en la tokenización de las secuencias MIDI originales se esclareció lo laborioso y meticuloso que tendrá que ser este proceso.

1. Retos Importantes

1.1. Hardware

Durante el curso trabajé completamente en Google Colab para trabajar en la parte de deep learning, sin embargo cuento con el equipo para poder correrlo localmente: una laptop con un GPU NVIDIA 1070M de 8GB y 2048 procesadores y una desktop con GPU AMD 7900 XTX de 24GB y 12288 procesadores. Fue un deseo y una necesidad propia de realizar este proyecto de manera local completamente.

1.2. Instalación de Keras/TF y PyTorch

Pasé múltiples días leyendo la documentación de AMD, para utilizar su biblioteca equivalente a CUDA (ROCm), y tratando de instalar las cosas necesarias. Fueron días porque no fue trivial hacer este proceso. Para Keras nunca logré que TensorFlow corriera en ROCm; siempre había un nuevo error al arreglar el anterior y me eventualmente me rendí cuando TensorFlow por fin reconoció mi GPU, pero sus operaciones seguían arrojando errores en cuanto empezaba a entrenar la primera época del modelo. Con PyTorch también fue doloroso, pero eventualmente logró correr correctamente con advertencias de ineficiencia por librerías incompatibles en el backend. Al no tener referencia de Keras en

mi GPU AMD, omití por completo su instalación en mi GPU NVIDIA, la cual no tuvo ningún problema para instalar CUDA y correr PyTorch.

1.3. Tiempo

Además de la instalación y funcionamiento de los frameworks en un GPU AMD, también la investigación y entendimiento de los modelos pertinentes consumión una gran cantidad de días, por lo que me dejo con poco tiempo para correr los modelos e implementar las mejoras y las características deseadas. Un objetivo original, generación text-to-music, se tuvo que quitar complemtamente del alcance del proyecto a causa del poco tiempo disponible.

2. Modelo

Se utilizó una arquitecura de solo-decoder con un gran énfasis en tokenización y embedización de los datos MIDI. Lo más interesante de este Modelo es la utilización un función de atención causal, es decir que solo toma en cuenta datos que hayan sucedido antes. Este tipo de modelos se utiliza seguido en modelos tipo GPT para generación de texto, aunque en esos sí se utiliza el encoder también. La referencia principal para este modelo fue el libro de Mark Liu [1].

Para mejorar este modelo centré mi atención en la parte que calcula la atención, implementé la función built-in *scaled dot product attention* en vez de calcularlo a mano. Además de algunos cambios menores adicionales. Este cambio redujo el tiempo de ejecución por aproximadamente 15 %, por lo que lo considero un gran éxito.

El modelo contiene 20.16 millones de parámetros y se utilizaron 100 épocas para su entrenamiento.

3. Tiempos de ejecución

Cuadro 1: Comparación de tiempo de ejecución

Modelo	Minutos
PyTorch en NVIDIA	421
PyTorch en AMD	91
Modelo modificado (AMD)	75

Es clara la diferencia entre los GPUs, dado que el GPU de AMD es considerablemente más nuevo y potente. Me da mucho gusto poder confiar en que este GPU será adecuado para entrenar modelos medianos y chicos a una buena velocidad sin tener que recurrir a la nube. Me preocupa que en múltiples foros de discusión mencionan que los GPUs de AMD son menos eficientes que los de NVIDIA, pero con el estado actual de las tarifas, será muy dificil conseguir un GPU moderno de NVIDIA que valga la pena.

4. Conclusión

Habiendo completado este proyecto me ha abierto los ojos, de nuevo, a que el preprocesamiento de los datos es trabajo más importante incluso antes de querer decidir o diseñar una arquitectura. Adjuntaré múltiples MIDIs generados por el modelo (donde los primeros segundos son el prompt), y se notará para cualquier escucha, que al subir la temperatura, las piezas generadas adquieren más naturalidad.

El enfoque del modelo que trabajé se preocupa mucho por secuencias de nota y su contexto a partir del positional encoding y el embedding, pero a fin de cuentas lo ve como una secuencia. Esto límita gigantescamente al modelo sobre todo a generar melodías, pero no armonías. Para generar armonías se tendría que rehacer todo el procesamiento y la tokenización de los datos y sin duda aumentaría el número de tokens posibles. El proyecto también me enseñó a no tomar el hardaware por hecho, pues no me imaginé que sería casi imposible utilizar Keras y TF con el GPU de una de las compañías más grandes de hardware coomputacional del mundo. Sin duda quedo motivado para seguir aprendiendo y experimentando.

*Adjunto el código y los MIDIs generados al pull request

Referencias

[1] Liu,Mark. *Learn Generative AI with PyTorch.* Manning, 2024.