SaShiMi: Adapted for Google Colab

Leo Ren[1], Roger Jin[2]
[1]Buckingham Browne & Nichols, Cambridge, MA
[2]Massachusetts Institute of Technology, Cambridge, MA

Abstract

This document explains how to convert SaShiMi, a music generation software, into something more resource-friendly.

Key Innovations

- Adapting code to Python 3.7
- Allowing SaShiMi to run on lower resources and minimal storage

Practical Implications

This paper allows the more average person without expensive equipment to run and train AI models.

Introduction

SaShiMi (GGDR22) is a music generation AI that uses S4D (GGR21) to train the model. The model uses three second clips of a single instrument and uses that to train. This was created by GitHub users albertfgu, krandiash, and tjppires. They have three example datasets, called Beethoven, sc09, and YoutubeMix, which were all available to download on huggingface. The creators of this model used A100 GPUs to train the model.

Adaptations for Python 3.7.13

SaShiMi runs on Python 3.10. This project used Google Colab, which ran on 3.7.13, which did not include many of the features of 3.10, such as the walrus operator and certain unpacking commands. All walrus operators were replaced with code suitable for Python 3.7. On top of that, certain unpacking commands, such as ** and *** were also replaced with code that worked in Python 3.7. This was done by reworking some parentheses, which allowed Python 3.7 to unpack the variables correctly.

Our fork is made available here.

Data

The dataset used to run and edit SaShiMi was the sc09 dataset provided by krandiash on their huggingface page. To limit storage on the computer, the dataset was downloaded from the page, uploaded to a throwaway Google Drive account, and unzipped within Colab. This dataset was used to bug test, translate the code from Python 3.10 to Python 3.7, and make sure that the model was able to run with Colab's resources. The actual dataset used was a complete playlist of Mozart's Sonatas, which was found on Youtube. This was downloaded with YouTubeDL, which homebrew, a package runner for MacOS, ran. Once downloaded, ffmpeg was used to cut the Sonatas into 2 second clips, which again, was run using homebrew. This dataset was then zipped and uploaded onto Google Drive.

Running the Code

When running the code, there were many problems, most of which were due to Colab not having enough resources to run the code. We made the mistake by buying Colab Pro+, which did not help as the free version Colab has the same GPU as Colab Pro+. In the end, by changing the model.d size, the amount of layers, and the loader batch size, the code was able to be run in Colab. The code used was

```
python -m train experiment=sashimi-sc09 \
wandb=null model.n_layers=1 model.d_model=32
```

which allowed it to run within the resources provided by Colab.

Checkpointing

When running the code, issues with runtime arose. Since Google Colab only allows limited runtime, checkpointing was needed. To limit the storage space needed, Google Drive was mounted to the Notebook, which allowed it to save on the cloud for it to be picked up later. This allows us to run mostly everything, except for the data collection, on Google's resources.

Conclusion

This paper has shown how to run code while using minimal resources from your own computer. It also covered how to download data from YouTube and process it automatically.

References

[GGDR22] Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré. It's Raw! Audio Generation with State-Space Models. *arXiv*, February 2022. arXiv:2202.09729, doi:10.48550/arXiv.2202.09729.

[GGR21] Albert Gu, Karan Goel, and Christopher Ré. Efficiently Modeling Long Sequences with Structured State Spaces. *arXiv*, October 2021. arXiv:2111.00396, doi:10.48550/arXiv.2111.00396.