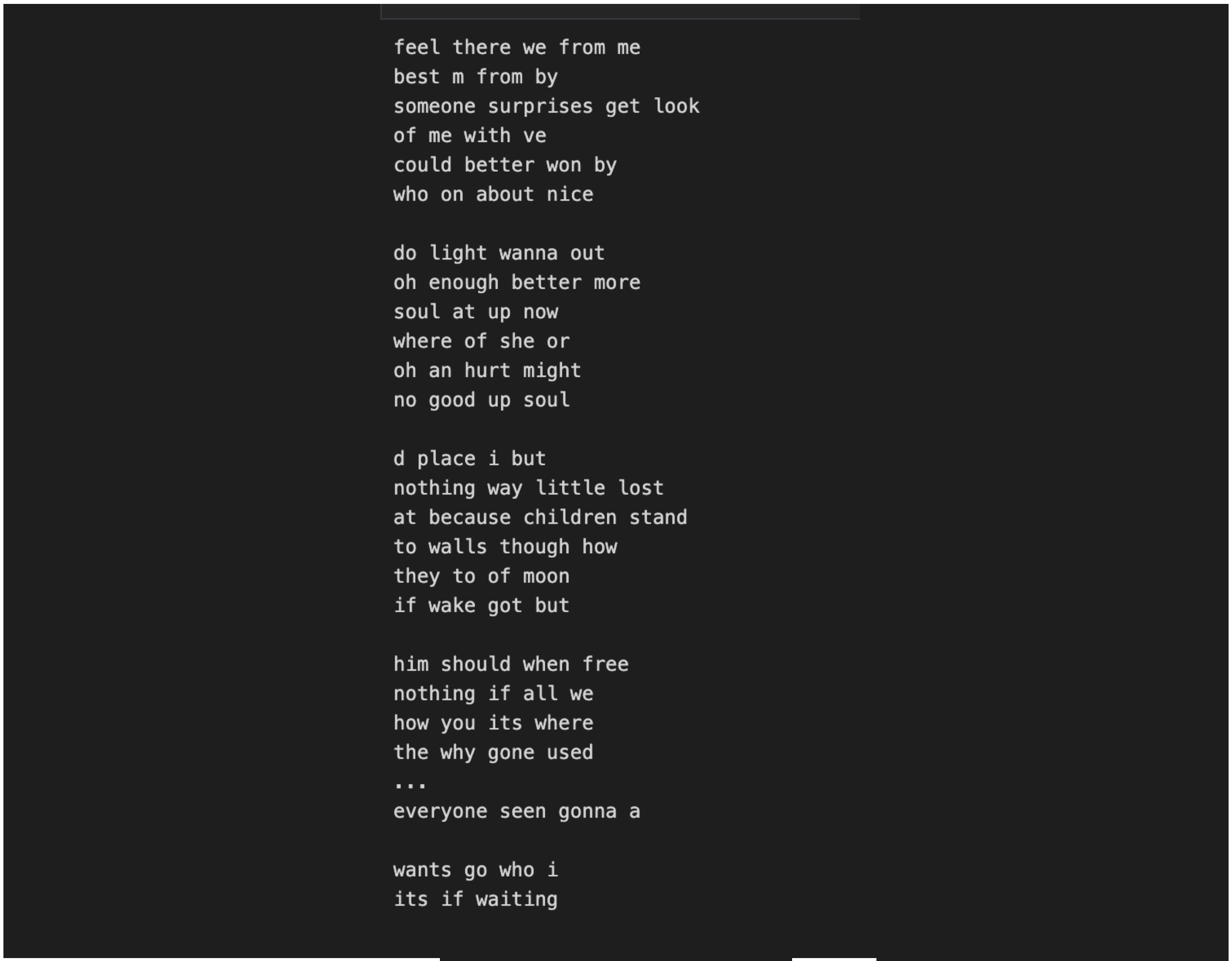
Final Project - Yenkai Huang

1. DESCRIPTION OF YOUR PROJECT PROCESS: MATERIALS, TECHNIQUES (MODELS, SOFTWARE), ITERATIONS

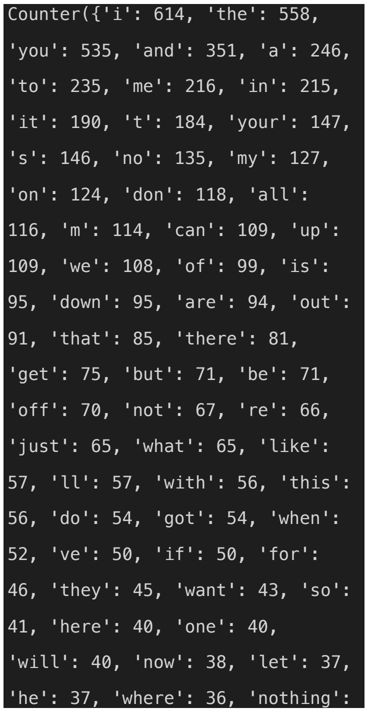
Building upon the foundational materials and methodologies from Assignment 5 for my final project, I dedicated substantial time towards understanding the process of generating long-format songs that preserve an intact music structure. Specifically, I delved deeply into the Jukebox codebase. My efforts to attain this objective were primarily bifurcated into two broad categories: lyrics conditioning and song prompting.

When working with lyrics conditioning, I experimented with generating both long and short pieces. This encompassed lyrics produced by ChatGPT, original song lyrics, repetitive single-word lyrics, and lyrics randomly generated from the most frequently used words across all the artist's albums. After several trials, it became clear that Jukebox anticipates the output to adhere to a rudimentary musical structure of intro->verse->outro. In an attempt to distribute the complete lyrics across the designated length, the model appears to have an optimal generation length in the range of 1.5 to 1.8 minutes, based on my observations. When the desired length is too brief, such as 30 seconds, the model attempts to cram all the lyrics in, resulting in poor audio quality. Conversely, if the song exceeds 2 minutes, the model finds it challenging to incorporate complex structures like the chorus, as the structure of intro->verse->chorus->verse->outro necessitates a more advanced model and greater computational resources. To put it simply, Jukebox essentially tries to extend the samples it's given, with the model striving to populate the notes for the specified duration before concluding with an outro.

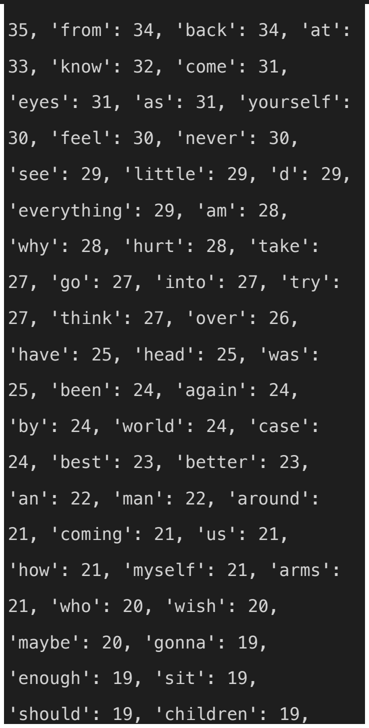
Consequently, addressing the structure problem emerged as my primary objective. I formulated a method that sliced the original song into ten segments and utilized these as prompts for Jukebox to generate numerous brief outputs. I then manually stitched the outputs of each part together using a Digital Audio Workstation (DAW), aiming to acquire the optimal combination while introducing some AI-induced variations. This approach maintains the "intentional structure" of the original song. Although this was successful, I was motivated to advance the experiment even further: to generate long-length song without editing.



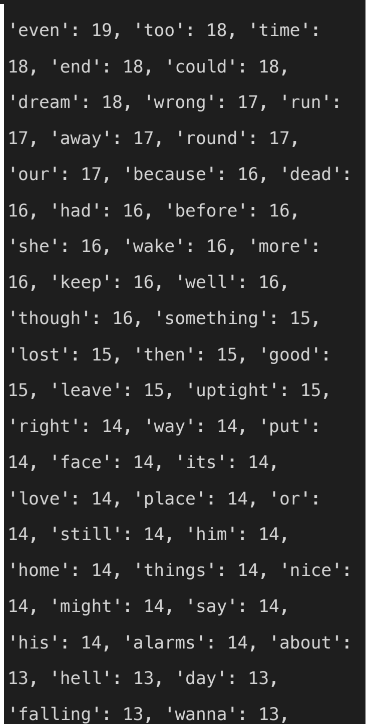
```
feel there we from me
best m from by
someone surprises get look
of me with ve
could better won by
who on about nice

do light wanna out
oh enough better more
soul at up now
where of she or
oh an hurt might
no good up soul

d place i but
nothing way little lost
at because children stand
to walls though how
they to of moon
if wake got but

him should when free
nothing if all we
how you its where
the why gone used
...
everyone seen gonna a

wants go who i
its if waiting
```



```
Counter({'i': 614, 'the': 558,
'you': 535, 'and': 351, 'a': 246,
'to': 235, 'me': 216, 'in': 215,
'it': 190, 't': 184, 'your': 147,
's': 146, 'no': 135, 'my': 127,
'on': 124, 'don': 118, 'all':
116, 'm': 114, 'can': 109, 'up':
109, 'we': 108, 'of': 99, 'is':
95, 'down': 95, 'are': 94, 'out':
91, 'that': 85, 'there': 81,
'get': 75, 'but': 71, 'be': 71,
'off': 70, 'not': 67, 're': 66,
'just': 65, 'what': 65, 'like':
57, 'll': 57, 'with': 56, 'this':
56, 'do': 54, 'got': 54, 'when':
52, 've': 50, 'if': 50, 'for':
46, 'they': 45, 'want': 43, 'so':
41, 'here': 40, 'one': 40,
'will': 40, 'now': 38, 'let': 37,
'he': 37, 'where': 36, 'nothing':
```

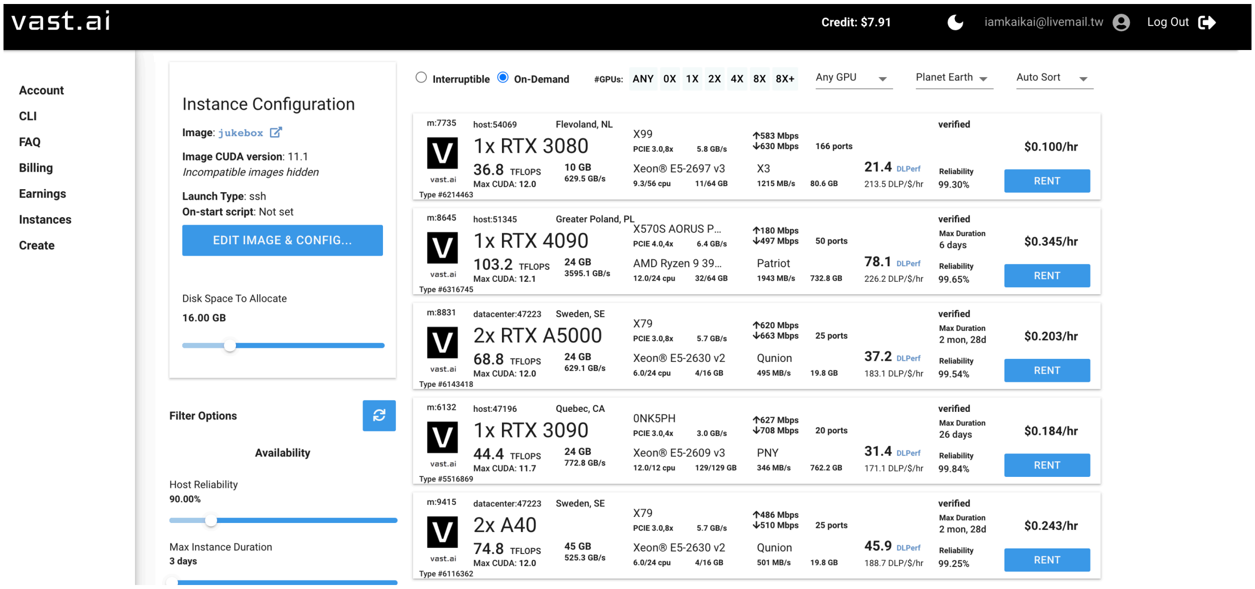


```
35, 'from': 34, 'back': 34, 'at':
33, 'know': 32, 'come': 31,
'eyes': 31, 'as': 31, 'yourself':
30, 'feel': 30, 'never': 30,
'see': 29, 'little': 29, 'd': 29,
'everything': 29, 'am': 28,
'why': 28, 'hurt': 28, 'take':
27, 'go': 27, 'into': 27, 'try':
27, 'think': 27, 'over': 26,
'have': 25, 'head': 25, 'was':
25, 'been': 24, 'again': 24,
'by': 24, 'world': 24, 'case':
24, 'best': 23, 'better': 23,
'an': 22, 'man': 22, 'around':
21, 'coming': 21, 'us': 21,
'how': 21, 'myself': 21, 'arms':
21, 'who': 20, 'wish': 20,
'maybe': 20, 'gonna': 19,
'enough': 19, 'sit': 19,
'should': 19, 'children': 19,
```



```
'even': 19, 'too': 18, 'time':
18, 'end': 18, 'could': 18,
'dream': 18, 'wrong': 17, 'run':
17, 'away': 17, 'round': 17,
'our': 17, 'because': 16, 'dead':
16, 'had': 16, 'before': 16,
'she': 16, 'wake': 16, 'more':
16, 'keep': 16, 'well': 16,
'though': 16, 'something': 15,
'lost': 15, 'then': 15, 'good':
15, 'leave': 15, 'uptight': 15,
'right': 14, 'way': 14, 'put':
14, 'face': 14, 'its': 14,
'love': 14, 'place': 14, 'or':
14, 'still': 14, 'him': 14,
'home': 14, 'things': 14, 'nice':
14, 'might': 14, 'say': 14,
'his': 14, 'alarms': 14, 'about':
13, 'hell': 13, 'day': 13,
'falling': 13, 'wanna': 13,
'standing': 13, 'walking': 13,
'surprises': 13, 'waiting': 12,
'gone': 12, 'look': 12, 'please':
12, 'wears': 12, 'always': 12,
'stand': 12, 'would': 12,
'everyone': 12, 'release': 12,
'slow': 12, 'quiet': 12, 'moon':
12, 'free': 12, 'ripcord': 12,
'wants': 11, 'light': 11,
'happen': 11, 'running': 11,
'bit': 11, 'broken': 11,
'someone': 10, 'seen': 10, 'big':
10, 'going': 10, 'reasonable':
10, 'stop': 10, 'walls': 10,
'long': 10, 'escape': 10, 'oh':
10, 'soul': 10, 'new': 10, 'won':
10, 'cut': 10, 'used': 10,
'comes': 10, 'turn': 10,
'forget': 10, 'denial': 10,
'inside': 10, 'her': 10,
'friends': 10, 'living': 10,
'bed': 10, 'rain': 10, 'prove':
10, 'doors': 10, 'need': 9,
'two': 9, 'any': 9, 'fall': 9,
'really': 9, 'words': 9, 'give':
9, 'left': 9, 'relief': 9,
'stay': 9, 'house': 9, 'tell': 9,
'high': 9, 'thing': 9, 'live': 9,
'hanging': 9, 'shadows': 9,
'open': 9, 'tried': 9, 'such': 9,
'land': 9, 'car': 8, 'hand': 8,
'wrapped': 8, 'else': 8,
'pretty': 8, 'real': 8, 'last':
8, 'much': 8, 'baby': 8, 'red':
8, 'taking': 8, 'town': 8, 'sea':
8, 'life': 8, 'suck': 8, 'dry':
8, 'these': 8, 'fear': 8,
'alive': 8, 'belong': 8,
'holding': 8, 'fire': 8, 'stars':
8, 'only': 7, 'full': 7, 'has':
7, 'dance': 7, 'ground': 7,
'minute': 7, 'people': 7, 'wall':
7, 'poison': 7, 'blue': 7,
'ever': 7, 'crazy': 7, 'weird':
7, 'mean': 7, 'bullet': 7,
'proof': 7, 'burst': 7, 'help':
7, 'sometimes': 7, 'bones': 7,
'first': 7, 'happening': 7,
'morning': 7, 'kids': 7, 'sleep':
7, 'breathing': 7, 'choke': 7,
'heart': 7, 'fell': 7, 'next': 6,
'understand': 6, 'talking': 6,
'body': 6, 'follow': 6, 'speak':
6, 'watch': 6, 'stuffed': 6,
'soon': 6, 'hear': 6, 'turns': 6,
'drop': 6, 'black': 6, 'crawled':
6, 'alright': 6, 'did': 6,
'cause': 6, 'videotape': 6,
'control': 6, 'earth': 6, 'hit':
6, 'under': 6, 'kill': 6,
'another': 6, 'wanted': 6,
'their': 6, 'fade': 6, 'breathe':
6, 'rest': 6, 'until': 6, 'eat':
6, 'some': 6, 'half': 6,
'message': 6, 'read': 6, 'yeah':
6, 'yesterday': 6, 'somewhere':
6, 'meet': 6, 'may': 6,
'remember': 6, 'height': 6,
'grow': 6, 'myxomatosis': 6,
'doesn': 6, 'heaven': 6, 'start':
6, 'set': 6, 'doubt': 6,
'special': 6, 'course': 6,
'stick': 5, 'slowly': 5, 'name':
5, 'mind': 5, 'thinking': 5,
'empty': 5, 'cat': 5, 'once': 5,
'through': 5, 'everybody': 5,
'warning': 5, 'climbing': 5,
'ladder': 5, 'fucking': 5,
'whatever': 5, 'hour': 5, 'went':
5, 'houses': 5, 'talk': 5,
'listen': 5, 'green': 5,
'matter': 5, 'bottom': 5,
'cards': 5, 'mine': 5, 'throw':
5, 'hurts': 5, 'young': 5,
'shrinkers': 5, 'uncle': 5,
'bill': 5, 'belisha': 5,
'beacon': 5, 'every': 5, 'spit':
5, 'blow': 5, 'blood': 5, 'feed':
5, 'race': 5, 'god': 5, 'change':
5, 'bunker': 5, 'laugh': 5,
'allowed': 5, 'ice': 5, 'came':
5, 'fantasy': 5, 'lies': 5,
'kid': 5, 'alarm': 5, 'idiot': 5,
'makes': 5, 'born': 5, 'peace':
5, 'believe': 5, 'anytime': 5,
'care': 5, 'crack': 5, 'wings':
4, 'days': 4, 'lying': 4,
'anyone': 4, 'blame': 4, 'human':
4, 'white': 4, 'ought': 4,
'plastic': 4, 'song': 4, 'eye':
4, 'nightmare': 4, 'hide': 4,
'call': 4, 'police': 4, 'dark':
4, 'bastard': 4, 'topsy': 4,
'turvy': 4, 'started': 4,
'tongue': 4, 'instrumental': 4,
'powerful': 4, 'today': 4,
'edge': 4, 'chance': 4, 'fishes':
4, 'friend': 4, 'table': 4,
'frightened': 4, 'pay': 4,
'money': 4, 'hole': 4, 'ones': 4,
'sky': 4, 'alone': 4, 'window':
4, 'wash': 4, 'happy': 4, 'walk':
4, 'break': 4, 'sulk': 4, 'burn':
4, 'already': 4, 'door': 4,
'strong': 4, 'age': 4,
'dinosaurs': 4, 'bell': 4,
'furniture': 4, 'park': 4,
'drill': 4, 'lights': 4,
'nobody': 4, 'beautiful': 4,
'while': 4, 'woke': 4, 'sucking':
4, 'lemon': 4, 'lock': 4, 'far':
4, 'forwards': 4, 'backwards': 4,
'hey': 4, 'mess': 4, 'paranoid':
4, 'android': 4, 'does': 4,
'great': 4, 'water': 4, 'tied':
4, 'frozen': 4, 'pull': 4,
'interstellar': 4, 'save': 4,
'universe': 4, 'hope': 4, 'sick':
4, 'show': 4, 'hysterical': 4,
'were': 4, 'flan': 4, 'calls': 4,
'reach': 4, 'ah': 4,
'scatterbrain': 4, 'babies': 4,
'bend': 4, 'otherside': 4,
'bells': 4, 'weary': 4,
'carrying': 4, 'stole': 4,
'hook': 4, 'crook': 4, 'hard': 4,
'everytime': 4, 'waters': 4,
'inevitable': 4, 'touch': 4,
'caught': 4, 'whispering': 4,
'shouting': 4, 'confusion': 4,
'guitar': 4, 'chat': 4, 'after':
4, 'years': 4, 'honesty': 4,
'wander': 4, 'promised': 4,
'animal': 3, 'trapped': 3}
```

I devised a resource-intensive solution: to generate as many long, low-level (fast but noisy) samples as possible, with the hope of securing a "perfect" output that would eliminate the need for editing. To accomplish this, I discovered a Docker image that can mirror the development environment of Jukebox: https://hub.docker.com/r/btrude/jukebox-docker. This resource opened up the opportunity for me to manipulate even more hyperparameters and potentially fine-tune the model. Additionally, I found a cost-effective GPU-sharing platform Vast.ai , which granted me access to high-performance hardware.



Instance Configuration

Image: jukebox

Image CUDA version: 11.1
Incompatible images hidden

Launch Type: ssh
On-start script: Not set

EDIT IMAGE & CONFIG...

Disk Space To Allocate
16.00 GB

Filter Options

Availability

Host Reliability
90.00%

Max Instance Duration
3 days

○ Interruptible ● On-Demand #GPUs: ANY 0X 1X 2X 4X 8X 8X+ Any GPU Planet Earth Auto Sort

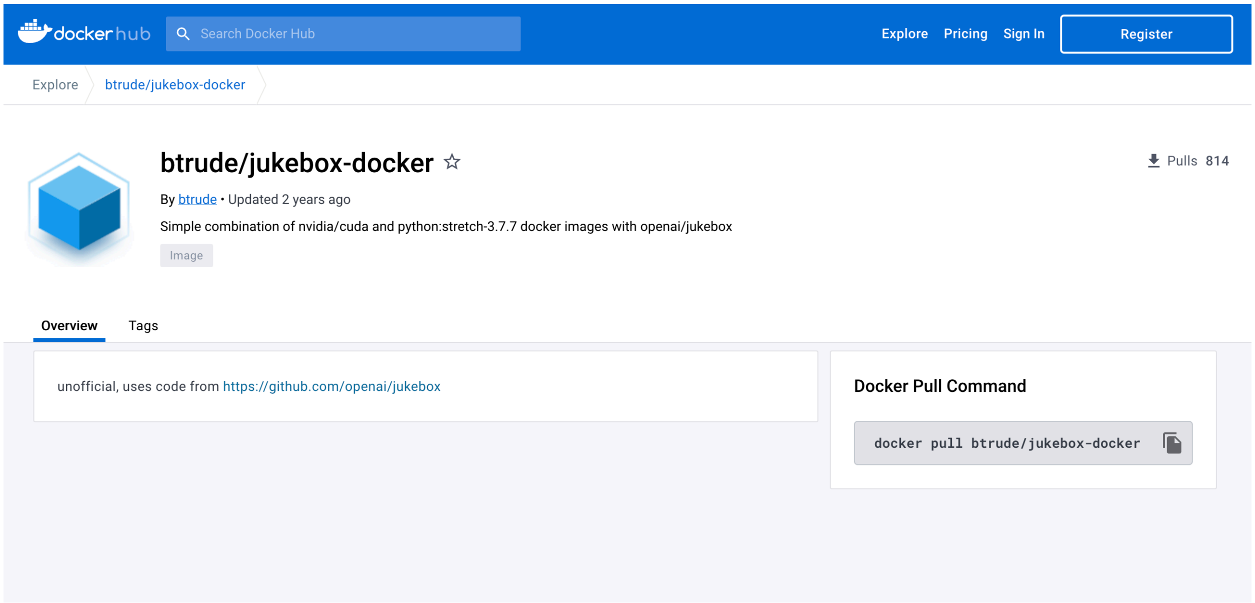
m:7735 host:54069 Flevoland, NL
1x RTX 3080 · 36.8 TFLOPS · Max CUDA: 12.0 · 10 GB 629.5 GB/s · vast.ai · Type #6214463
X99 · PCIE 3.0,8x 5.8 GB/s · Xeon® E5-2697 v3 · 9.3/56 cpu 11/64 GB
↑583 Mbps ↓630 Mbps · 166 ports · X3 · 1215 MB/s 80.6 GB
21.4 DLPerf · 213.5 DLP/$/hr
verified · Reliability 99.30%
$0.100/hr RENT

m:8645 host:51345 Greater Poland, PL
1x RTX 4090 · 103.2 TFLOPS · Max CUDA: 12.1 · 24 GB 3595.1 GB/s · vast.ai · Type #6316745
X570S AORUS P... · PCIE 4.0,4x 6.4 GB/s · AMD Ryzen 9 39... · 12.0/24 cpu 32/64 GB
↑180 Mbps ↓497 Mbps · 50 ports · Patriot · 1943 MB/s 732.8 GB
78.1 DLPerf · 226.2 DLP/$/hr
verified · Max Duration 6 days · Reliability 99.65%
$0.345/hr RENT

m:8831 datacenter:47223 Sweden, SE
2x RTX A5000 · 68.8 TFLOPS · Max CUDA: 12.0 · 24 GB 629.1 GB/s · vast.ai · Type #6143418
X79 · PCIE 3.0,8x 5.7 GB/s · Xeon® E5-2630 v2 · 6.0/24 cpu 4/16 GB
↑620 Mbps ↓663 Mbps · 25 ports · Qunion · 495 MB/s 19.8 GB
37.2 DLPerf · 183.1 DLP/$/hr
verified · Max Duration 2 mon, 28d · Reliability 99.54%
$0.203/hr RENT

m:6132 host:47196 Quebec, CA
1x RTX 3090 · 44.4 TFLOPS · Max CUDA: 11.7 · 24 GB 772.8 GB/s · vast.ai · Type #5516869
0NK5PH · PCIE 3.0,4x 3.0 GB/s · Xeon® E5-2609 v3 · 12.0/12 cpu 129/129 GB
↑627 Mbps ↓708 Mbps · 20 ports · PNY · 346 MB/s 762.2 GB
31.4 DLPerf · 171.1 DLP/$/hr
verified · Max Duration 26 days · Reliability 99.84%
$0.184/hr RENT

m:9415 datacenter:47223 Sweden, SE
2x A40 · 74.8 TFLOPS · Max CUDA: 12.0 · 45 GB 525.3 GB/s · vast.ai · Type #6116362
X79 · PCIE 3.0,8x 5.7 GB/s · Xeon® E5-2630 v2 · 6.0/24 cpu 4/16 GB
↑486 Mbps ↓510 Mbps · 25 ports · Qunion · 501 MB/s 19.8 GB
45.9 DLPerf · 188.7 DLP/$/hr
verified · Max Duration 2 mon, 28d · Reliability 99.25%
$0.243/hr RENT



btrude/jukebox-docker ☆

Pulls 814

By btrude • Updated 2 years ago

Simple combination of nvidia/cuda and python:stretch-3.7.7 docker images with openai/jukebox

Image

Overview Tags

unofficial, uses code from https://github.com/openai/jukebox

Docker Pull Command

```
docker pull btrude/jukebox-docker
```

Nonetheless, Jukebox is a considerably slow model. After dedicating 2-3 full days to training, I was only able to generate 30 pieces of 2.5-minute songs. Among these samples, a mere 3-5 outputs met my satisfaction. The rest either started promisingly but quickly veered off course, or struggled to formulate a reasonable structure. Interestingly, I observed that despite setting the generation length at 2.5 minutes, most songs began to fade out after the 2-minute mark and the quality noticeably declined after 1 minute. This highlighted the model's limitations, and I ultimately had to resort to editing the output samples in Ableton Live. I managed to use three generated outputs, discovering some clever tactics to weave different pieces together. While Jukebox excels at maintaining tempo and key, my final composition still exhibited some evident gaps between sections. Identifying these obstacles was an insightful part of the process.

Source

1. 15-steps - Radiohead: https://genius.com/Radiohead-15-step-lyrics
2. jukebox-docker: https://hub.docker.com/r/btrude/jukebox-docker/
3. btrude's github: https://github.com/btrude/jukebox-docker
4. Vast.ai: https://vast.ai/
5. Jukebox: https://github.com/openai/jukebox
6. Google colab: https://colab.research.google.com/drive/1YjOmczsWqPl3rIrBJ1I7PkzqUyRtOxVI
7. My Python lyrics analysis script (in the attachment)