Guys Performing Transformations GPT

Group Name: AnimeLovers42
Names: Jerry Li, Pranesh Naagamuthu

Project Idea:

Created a decoder-only Transformer on transcribed text from a previous Zoom class recording to create a GPT-2 style Transformer in CUDA.The result of our project should be text generated based on speech in Zoom classes. Our goal for this project was to accelerate the model inference function by coding it in CUDA.

Acceleration from GPU:

Key Components:

- Matrix Multiplication (sgemm) - Basic functionality used everywhere. We implemented tiled matrix multiplication with shared memory for thread-level parallelism.

- Softmax -Utilized for many functions such as attention score, Layer Norm, and Generation. We parallelized it by using row-wise parallelism and max reduction for less memory usage and better parallelism.

- QKV Projection - Combines Query, Key, and Value weights. Threads independently copy blocks from the input into the QKV matrix

The stage of the software pipeline is parallelized

The above functions being accelerated speed up a lot of the process due to the fundamental requirement of those functions. The above functions apply to every other function in the codebase and significantly speed up the functions.

Implementation Timeline-

Week 6:
- We coded miniGPT in Python

Week 7:
- Coded LoadMatrix and tested the function

Week 8:
- Coded LayerNorm, Sgemm and Softmax

Week 9:
- Coded TransformerBlock

Week 10:
- Coded AttentionHead

Week 11:
- Coded MultiHeadAttention, Decoder, Embedding and Feed Forward

How to run the code:
Run the following two lines in the files :

nvcc transformer_block.cu -o transformer_block -I/usr/local/cuda-11.3/include -L/usr/local/cuda-11.3/lib64 -lcublas -lcudart -arch=sm_75 -std=c++14

./transformer_block

Evaluation:
Condition: Failed to implement the genator but all the other code was implemented

Problems Faced:

- Generate function was a bit of trouble and took a long time due to not understanding the fundamental structure of it. The structure being, one of the functions is making randomness, resulting in new outputs even if using same logits for the operations.
- The MultiHeadAttention was tricky since we had to turn it into a combined matrix to parallelize all at once.

Function	Jerry Li	Pranesh Naagamuthu
Python Implimentation	50%	50%
Sgemm.cu	100%	0%
Layer Norm.cu	0%	100%
softmax.cu	100%	0%
Transformer Block.cu	100%	0%
tools.cu	100%	0%
Generate	0%	100%
Positional Encoding	0%	100%
Forward	100%	0%
Report	25%	75%

Video
https://app.screencastify.com/watch/aWLGvekWK6zyhQWwj4TO