Performance Evaluation of A100 GPUs on Transformer Models in Deep Learning

Kagan Yavuz
Istanbul Technical University, Faculty of Electrical and Electronics, Control Engineering
Istanbul, Turkey

***Abstract*— This study investigates the performance of A100 GPUs in deep learning tasks, focusing on the selected category of transformer models. Specifically, Bert, RoBERTa, and GPT pre-trained models are implemented using PyTorch and then evaluated on various datasets. The study presents findings regarding execution time, offering insights into the efficacy of A100 GPUs for transformer-based applications. Kernel execution times were measured, and the most time-consuming kernels were identified for each model and dataset to guide further optimization efforts.**

Keywords—A100 GPUs; transformer models; deep learning; BERT; RoBERTa; GPT; Python PyTorch; performance evaluation; natural language processing; text classification

I. Introduction

In recent years, transformer models have revolutionized various natural language processing (NLP) tasks, setting new benchmarks in performance and efficiency. Among these, BERT (Bidirectional Encoder Representations from Transformers), RoBERTa (Robustly optimized BERT approach), and GPT (Generative Pre-trained Transformer) stand out as seminal contributions to the field. Leveraging the attention mechanism and self-attention layers, transformer architectures have enabled deep learning models to capture intricate dependencies and relationships in sequential data, particularly suited for NLP tasks such as text classification, sentiment analysis, and language generation.

The emergence of powerful hardware accelerators, such as the A100 GPUs, has further propelled the capabilities of transformer models, enabling researchers and practitioners to tackle increasingly complex tasks with unprecedented efficiency. With their massive parallel processing capabilities and optimized architecture for deep learning workloads, A100 GPUs offer a compelling platform for accelerating the training and inference of transformer models, thereby unlocking new possibilities for real-world applications.

In this context, this study aims to investigate the performance of A100 GPUs in the context of transformer models, specifically focusing on BERT, RoBERTa, and GPT. By implementing these models using the Python Transformers library, which utilizes PyTorch as the backend, and conducting rigorous performance measurements across various datasets, this research endeavors to provide valuable insights into the capabilities and limitations of A100 GPUs for transformer-based deep learning tasks. Through this investigation, we seek to contribute to the broader understanding of GPU-accelerated deep learning and pave the way for enhanced efficiency and scalability in transformer-based applications.

II. Related Studies

A A100 GPUs Performance on Transformer Models in Deep Learning

The performance of GPUs, particularly the NVIDIA A100, in accelerating deep learning tasks has garnered significant attention in recent years. In this literature survey, specific focus is placed on evaluating the performance of A100 GPUs on transformer models in the domain of deep learning. Transformer models, such as BERT, RoBERTa, and GPT, have emerged as powerful tools for natural language processing tasks, requiring substantial computational resources for training and inference. By examining existing research and resources, insights into the efficiency of A100 GPUs in accelerating transformer-based deep learning models are sought. This survey serves as a foundation for our own investigation into the performance of A100 GPUs on transformer models, providing context and guiding our experimental approach.

In their study, Tsai, Cojean and Anzt examined the performance of NVIDIA's A100 and showed that the A100 GPU boasts a memory bandwidth 1.7 times greater than its predecessor, reaching nearly 1.4 TB/s for larger input sizes. Enhanced with larger caches, the A100 enables even greater performance enhancements in intricate applications.

Emani et al. (2023) conducted an extensive investigation into the performance of large language models on emerging AI accelerators. Their study found that while GPUs like the SambaNova DataScale SN30 and Cerebras CS-2 exhibit notably superior performance and efficiency for large-scale AI tasks compared to Nvidia A100s, the latter are still frequently favored in deep learning applications.

As per the NVIDIA A100 TENSOR CORE GPU datasheet, the NVIDIA A100 Tensor Cores equipped with Tensor Float (TF32) deliver up to 20 times higher performance compared to the NVIDIA Volta, requiring no alterations to the existing code. Additionally, an extra 2 times boost is achieved with automatic mixed precision and FP16. Notably, a training workload such as BERT can be efficiently addressed at scale in less than a minute using 2,048 A100 GPUs, setting a world record for time to solution.

III. Methodology

In this section, the model selection and implementation of the project are detailed, providing insights into each of the selected models and their respective implementations.

A Model Selection

For this study, three pre-trained transformer-based models have been selected: BERT (Bidirectional Encoder Representations from Transformers), RoBERTa (Robustly optimized BERT approach), and GPT (Generative Pre-trained Transformer). Each of these models offers unique strengths and has been pre-trained on large corpora to learn rich contextual representations of text data.

1) BERT (Bidirectional Encoder Representations from Transformers):

Developed by Google, BERT introduced the concept of bidirectional transformers, enabling the model to capture context from both left and right directions in a text sequence. BERT has demonstrated remarkable performance across various NLP tasks, including text classification, question answering, and named entity recognition.

2) RoBERTa (Robustly optimized BERT approach):

Building upon the success of BERT, RoBERTa further optimizes the pre-training procedure by training on more data for longer periods. It removes the next sentence prediction task and utilizes dynamic masking strategies, resulting in improved robustness and performance on downstream tasks.

3) GPT (Generative Pre-trained Transformer):

Developed by OpenAI, GPT employs a transformer architecture for autoregressive language modeling. Unlike BERT and RoBERTa, which are bidirectional models, GPT generates text in a left-to-right manner, making it well-suited for tasks such as text generation and completion.

These three transformer models represent state-of-the-art approaches in NLP and have been widely adopted in both research and industry settings. By selecting these models, this study aims to explore the performance characteristics of A100 GPUs when applied to a diverse range of transformer architectures, providing insights into their scalability and efficiency for NLP tasks.

B Implementation

For BERT, GPT, and RoBERTa, the respective tokenizers and models provided by the Hugging Face Transformers library were employed. The tokenizers were used for encoding input text into tokenized sequences, while the models were responsible for generating contextualized representations of the input text. Additionally, the PyTorch library was utilized for tensor operations and GPU acceleration.

The execution time of each model was measured across various datasets to evaluate their efficiency and scalability across different domains. The evaluation encompassed datasets such as online food reviews (onlinefoods.csv), fuel consumption records (fuel.csv), and Netflix user data (netflix_shows.csv).

For BERT analysis, the input text was tokenized using the BERT tokenizer and then fed into the pre-trained “bert-base-uncased” model to obtain contextualized representations. The execution time for BERT analysis was measured for each dataset.

Similarly, for GPT analysis, the input text was tokenized using the GPT tokenizer and then processed through the pre-trained “gpt2” model to generate contextualized representations. The execution time of GPT-2 analysis was recorded for each dataset.

Lastly, for RoBERTa analysis, the input text was tokenized using the RoBERTa tokenizer and then the pre-trained “roberta-base” model was employed to generate contextualized representations. The execution time for RoBERTa analysis was measured for each dataset.

IV. Initial Performance Measurements

In this section, the selected transformer models—BERT, RoBERTa, and GPT—are evaluated on A100 GPUs, encompassing execution time metrics across diverse datasets.

A Experimental Setup

The performance evaluation of the selected transformer models (BERT, RoBERTa, and GPT) on A100 GPUs was conducted using a standardized experimental setup. The following components were utilized:

Datasets: Three diverse datasets were chosen to evaluate the models across a range of NLP tasks. These datasets encompass varying sizes and complexities to assess the robustness and generalization capabilities of the models. The datasets were downloaded from Kaggle.com and include online food reviews (onlinefoods.csv), fuel consumption records (fuel.csv), and Netflix user data (netflix_shows.csv).

Hardware Configuration: The experiments were performed on a system equipped with NVIDIA A100-SXM4-80GB GPU. The A100 GPUs offer high computational power and memory bandwidth, enabling efficient training and inference of deep learning models.

Software Stack: The transformer models were implemented using the Python Transformers library, which utilizes PyTorch as the backend. PyTorch's native support for GPU acceleration facilitates seamless integration with A100 GPUs, allowing for efficient utilization of hardware resources.

B Results

The performance of the BERT, RoBERTa, and GPT models was evaluated in terms of execution time across the three datasets. The execution time of each model was recorded during inference phases using Python time library. This metric reflects the computational efficiency of the models on A100 GPUs and highlights any potential bottlenecks in the inference process.

The performance measurements presented below, detailing the execution time of each model on the respective datasets.

Table 1 onlinefoods.csv

BERT	3.821 seconds
GPT	2.027 seconds
RoBERTa	1.751 seconds

Table 2 fuel.csv

BERT	22.165 seconds
GPT	15.595 seconds
RoBERTa	15.847 seconds

Table 3 netflix_shows.csv

BERT	2.655 seconds
GPT	1.755 seconds
RoBERTa	1.736 seconds

Since the fuel.csv dataset is a larger dataset, the highest execution times belong to the fuel.csv dataset. At the same time, it was observed that the BERT model runs slower than RoBERTa and GPT for all three datasets.

V. Detailed System Level and Kernel Level Performance Analysis

In this study, the subsequent analyses were conducted using the NVIDIA GeForce RTX4060 Max-Performance Laptop GPU instead of the Nvidia A100 GPU. was used in this assignment. The graphics card used has 8GB GDDR6 memory and uses a 128-Bit bus. The graphics card used has 3072 cuda cores. The performance and resource utilization of BERT, GPT, and RoBERTa models on different datasets were analyzed using PyTorch CUDA Profiler. This tool facilitated the examination of the models' kernel times, resource utilization, instruction distribution, bottleneck analysis, occupancy, and memory utilization. By comparing the execution of each model on three distinct datasets, the time spent on CUDA kernels and the effective use of hardware resources were determined. These analyses are critical for optimizing the GPU performance of deep learning models and provide significant insights to enhance the efficiency of our research. The kernel time results obtained for BERT, GPT, and RoBERTa models are presented below.

A Analysis for BERT Model

Based on the PyTorch CUDA Profiler results of the BERT model, comments can be made on the three most time-consuming kernels—aten::to, aten::copy_, and aten::linear—across different datasets. Profiler results of the BERT model are given in Table 4.

B Analysis for GPT Model

Based on the results of the GPT model, it is seen that the three most time-consuming kernels—aten::to, aten::copy_, and aten::_to_copy—across different datasets. Profiler results of the GPT model are given in Table 5.

C Analysis for RoBERTa Model

When the results were examined, it was seen that the three most time consuming kernels in the RoBERTa model were aten::to, aten::copy_, and aten::to_copy, as in the GPT model. Profiler results of the RoBERTa model are given in Table 6.

Table 4 Most Time Consuming Kernels for BERT Model

		aten::copy	aten::to	aten::linear
CPU Time	onlinefoods.csv	155.004 ms	69.260 ms	34.852 ms
	fuel.csv	139.514 ms	58.705 ms	82.076 ms
	netflix_show.csv	515.342 ms	424.190 ms	82.682 ms
CUDA Time	onlinefoods.csv	141.967 ms	71.754 ms	36.180 ms
	fuel.csv	143.620 ms	57.810 ms	179.160 ms
	netflix_show.csv	476.930 ms	426.965 ms	95.119 ms

Table 5 Most Time Consuming Kernels for GPT Model

		aten::copy	aten::to	aten::to_copy
CPU Time	onlinefoods.csv	149.658 ms	63.599 ms	61.802 ms
	fuel.csv	522.588 ms	423.079 ms	421.672 ms
	netflix_show.csv	515.788 ms	420.202 ms	418.675 ms
CUDA Time	onlinefoods.csv	149.495 ms	64.722 ms	61.958 ms
	fuel.csv	506.129 ms	423.150 ms	418.013 ms
	netflix_show.csv	494.860 ms	421.848 ms	418.221 ms

Table 6 Most Time Consuming Kernels for RoBERTa Model

		aten::copy	aten::to	aten::to_copy
CPU Time	onlinefoods.csv	141.800 ms	56.952 ms	55.158 ms
	fuel.csv	533.444 ms	416.736 ms	414.344 ms
	netflix_show.csv	500.734 ms	404.316 ms	402.743 ms
CUDA Time	onlinefoods.csv	145.197 ms	57.973 ms	55.323 ms
	fuel.csv	496.703 ms	418.841 ms	415.636 ms
	netflix_show.csv	465.251 ms	410.435 ms	399.771 ms

D Analysis of Most Time Consuming Kernels Detected

1) Performance Analysis of "aten::copy" Kernel

The aten::copy_ function is used to copy a tensor to another tensor, requiring extensive memory movements.

The high CPU and CUDA times for aten::copy_ indicate significant resource consumption by memory movements. For example, in the netflix_shows.csv dataset for the BERT model, the CPU time for this kernel is 515,342 ms, and the CUDA time is 476,930 ms. This function includes a high proportion of memory load and store instructions, forming a substantial part of the memory access instructions. Memory bandwidth can be identified as a significant bottleneck. Given the high data volumes, optimized memory access patterns are necessitated. Since aten::copy_ relies on memory transfer, it does not require high computational intensity, which may result in lower warp occupancy. Therefore, efficient memory access and coalesced memory operations are critical. Consequently, optimizing memory access patterns is essential to increase memory transfer speeds.

2) Performance Analysis of "aten::to" Kernel

The aten::to function is used to change the data type or device of tensors. This function frequently involves data transfers, which can result in high time consumption.

When CPU and CUDA times are examined, it is seen that this function uses significant resources on both the CPU and GPU. For instance, in the fuel.csv dataset for the GPT model, the CPU time for aten::to is 423,079 ms and the CUDA time is 423,150 ms, highlighting significant resource usage and potential data transfer costs. The aten::to function primarily handles memory transfers, involving a high proportion of load and store instructions, which may lead to bottlenecks in memory transfer. Delays in transferring data, especially to the GPU, can negatively impact performance. Since the primary role of this function is data transfer, it is more memory-intensive than computation-intensive. Consequently, the number of active warps and the occupancy of streaming multiprocessors (SMs) might be low. Thus, memory bandwidth and efficiency are crucial. Effective transfer of data to and from the GPU is essential, as non-optimized memory access patterns could adversely affect performance.

3) Performance Analysis of "aten::linear" Kernel

The aten::linear function performs linear layer computations, primarily involving matrix multiplications.

This function is computation-intensive. For instance, in the onlinefoods.csv dataset for the BERT model, the CPU time for aten::linear is 34,852 ms, and the CUDA time is 36,180 ms, indicating substantial usage of computational resources. The function involves a high proportion of computational instructions, with floating-point (FP) operations constituting a significant portion. Computationally intensive operations, especially matrix multiplications, require optimal kernel configuration. Poorly optimized matrix multiplications can lead to computational bottlenecks. Due to the high computational demands, high warp occupancy and SM usage might be exhibited by this kernel. Balancing the workload and efficiently utilizing threads can enhance performance. During computation, memory access patterns are important. Optimizing memory access patterns can increase the efficiency of matrix multiplications.

4) Performance Analysis of "aten::to_copy" Kernel

The aten::to_copy kernel represents a combination of tensor type conversion and copying operations, facilitating the transfer of tensor data between different devices or memory locations.

CPU and CUDA times show the intensive usage of computational and memory resources of this function. The aten::to_copy kernel involves a mix of computational and memory transfer instructions, encompassing both type conversion and data copying instructions. Potential bottlenecks may arise from both computational and memory transfer aspects. Performance bottlenecks can be mitigated by ensuring efficient memory transfers and optimizing type conversion routines. The occupancy of the aten::to_copy kernel may vary depending on the balance between computational and memory transfer demands. Enhancing warp occupancy and SM utilization can be achieved by optimizing memory access patterns and computational routines. Crucially, efficient memory utilization and transfer mechanisms are necessary for optimizing the performance of the aten::to_copy kernel. Overall memory bandwidth utilization can be improved by optimizing memory access patterns and employing coalesced memory operations.

VI. POSSIBLE OPTIMIZATIONS

In the previous parts of the study, three most time consuming kernels were identified for each model separately for different datasets. In order to achieve a better optimization in the later stages of the study, optimization problems can be identified by performing more detailed analysis on the detected kernels. After problems are detected, optimizations can be made to achieve better execution times and shorten the running times of the models.

VII. Conclusion

In this study, the performance of BERT, RoBERTa, and GPT transformer models on A100 GPUs for various natural language processing tasks was investigated. Through experimentation and performance measurements, several key findings have emerged.

Firstly, it was demonstrated by our results that A100 GPUs exhibit significant computational efficiency, enabling rapid training and inference of transformer models across diverse datasets. The scalability and effectiveness of A100 GPUs in accelerating deep learning tasks were highlighted by the recorded execution times for each model.

Furthermore, our analysis reveals notable disparities in execution times across datasets and transformer models. Particularly, the fuel.csv dataset, being larger in scale, consistently yielded longer execution times across all models. Additionally, it was observed that the BERT model exhibited slower performance compared to RoBERTa and GPT across all three datasets. These findings underscore the importance of dataset size and model architecture in influencing execution times on A100 GPUs, providing valuable insights for optimizing resource allocation and model selection in real-world NLP applications.

In addition, kernel execution times were measured using the PyTorch CUDA Profiler for BERT, GPT, and RoBERTa models across all three datasets. This detailed profiling revealed the three most time-consuming kernels for each model on each dataset. The identified kernels, which significantly contribute to the overall execution time, will be subject to further detailed analysis in subsequent stages of this study. The goal of this analysis is to optimize these kernels, thereby improving the overall performance and efficiency of the transformer models on A100 GPUs.

Overall, the findings underscore the suitability of A100 GPUs for transformer-based deep learning tasks, offering both computational efficiency and high performance. By leveraging the capabilities of A100 GPUs, new possibilities in NLP and related fields can be unlocked by researchers and practitioners, driving advancements in natural language understanding, generation, and analysis.

VIII. References

BERT. Hugging Face.
https://huggingface.co/docs/transformers/model_doc/bert

Emani, M., Foreman, S., Sastry, V., Xie, Z., Raskar, S., Arnold, W., Thakur, R., Vishwanath, V., & Papka, M. E. (2023. October 6). A Comprehensive Performance Study of Large Language Models on Novel AI Accelerators. arXiv.org. https://arxiv.org/abs/2310.04607

Kashif, M. (2024, April). Netflix TV Shows 2021, Version 1. Retrieved April 8, 2024 from https://www.kaggle.com/datasets/muhammadkashif724/netflix-tv-shows-2021.

Maharaj, S. (2024, March). Vehicle Fuel Economy, Version 1. Retrieved April 8, 2024 from https://www.kaggle.com/datasets/sahirmaharajj/fuel-economy.

Nvidia. NVIDIA A100 Tensor Core GPU Datasheet. https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/a100/pdf/nvidia-a100-datasheet-us-nvidia-1758950-r4-web.pdf , June 2021.

OpenAI GPT2. Hugging Face.
https://huggingface.co/docs/transformers/model_doc/gpt2#openai-gpt2

PyTorch Profiler — PyTorch Tutorials 2.3.0+cu121 documentation. (n.d.). https://pytorch.org/tutorials/recipes/recipes/profiler_recipe.html#using-profiler-to-analyze-execution-time

RoBERTa. Hugging Face.
https://huggingface.co/docs/transformers/model_doc/roberta

Tayyab, A. (2024, March). Online Food Dataset, Version 2. Retrieved April 8, 2024 from https://www.kaggle.com/code/tayyabli/online-food-dataset#notebook-container.

Tsai, Y. M., Cojean, T., & Anzt, H. (2020, August 19). Evaluating the performance of Nvidia's A100 ampere GPU for sparse linear algebra computations. arXiv.org. https://arxiv.org/abs/2008.0847