Attention – FPGA is all you need!!!

Abstract— Transformer models have emerged as pivotal assets in modern applications, with a notable stronghold in Natural Language Processing (NLP), celebrated for their adaptability and unrivaled predictive precision. Although Graphics Processing Units (GPUs) have traditionally served as the cornerstone for executing transformer models, a discernible shift in focus is evident, gravitating towards the utilization of Field-Programmable Gate Arrays (FPGAs). This transition is motivated by the commendable attributes of FPGAs, which encompass exceptional performance, rapid development cycles, and intrinsic reconfigurability. This paper explores the burgeoning interest in deploying transformer models on FPGAs, highlighting the unique advantages they offer over traditional GPU-based implementations. Through comprehensive experimentation and analysis, we demonstrate the potential of FPGAs in enhancing the efficiency and scalability of transformer-based applications, thereby paving the way for a new era in computational acceleration for NLP tasks.This paper introduces a pioneering FPGA-based transformer accelerator meticulously crafted to elevate the performance and efficiency of transformer models. The focal point of this endeavor is the implementation of the multihead attention mechanism detailed in the seminal paper "Attention is All You Need." Leveraging the capabilities of a Xilinx FPGA, our accelerator is developed utilizing a high-level synthesis tool, ensuring a seamless integration of advanced optimizations. To propel performance to new heights, the proposed accelerator deploys sophisticated techniques, including optimized loop unrolling, parallelization of loops, pipelining, and dataflow enhancements.

Through the strategic application of these advanced methodologies, our objective is to orchestrate a substantial boost in the overall efficiency and execution speed of transformer models on FPGA platforms. This innovative approach aims not only to meet but to exceed the expectations set by GPU-based implementations, positioning FPGAs as formidable contenders in the realm of transformer model acceleration.

I. INTRODUCTION

The Transformer model diverges from traditional recurrent and convolutional architectures by relying solely on self-attention mechanisms, eliminating the need for recurrence and enabling parallelization across sequence elements. This departure from sequential processing brings about significant advantages, allowing the model to capture long-range dependencies more effectively and facilitating training on larger datasets.

Implementing the Transformer architecture on Field-Programmable Gate Arrays (FPGAs) offers several advantages, making it a compelling decision for certain applications. Some highlights of using FPGAs –

1. Parallelization and Efficiency:

Field-Programmable Gate Arrays (FPGAs) distinguish themselves through their remarkable proficiency in parallel processing, a capability that enables the simultaneous execution of multiple operations. The inherent parallelism inherent in FPGAs empowers them to handle and process diverse tasks concurrently, showcasing their versatility in orchestrating intricate computational workflows through the simultaneous execution of numerous operations. This unique attribute of FPGAs, facilitating the parallelized execution of operations, not only enhances computational efficiency but also contributes to the acceleration of complex algorithms and applications. The parallel processing prowess exhibited by FPGAs, with their ability to handle a multitude of operations concurrently, underscores their adaptability and effectiveness in meeting the demands of modern computational challenges.

2. Customization and Flexibility:

Field-Programmable Gate Arrays (FPGAs) distinguish themselves by virtue of their remarkable programmability, providing a versatile platform that can be finely tailored to meet specific and intricate application requirements. The high level of programmability inherent in FPGAs empowers developers and engineers with a granular level of control, allowing for the customization of these devices to precisely match the unique demands and specifications of diverse applications. This adaptability extends beyond basic functionality, enabling the optimization of resource utilization, power consumption, and overall performance to align seamlessly with the nuanced needs of the targeted application. The extensive programmability of FPGAs not only enhances their versatility but also positions them as a dynamic solution capable of evolving and adapting to the evolving landscape of technological and application-specific demands.

3. Low Latency and Real-Time Processing:

FPGAs offer low-latency processing, making them suitable for real-time applications. The parallel nature of the Transformer architecture, coupled with FPGA's ability to handle multiple computations simultaneously, contributes to reduced inference time, crucial for applications with stringent latency requirements.

4. Energy Efficiency:

FPGAs are known for their energy efficiency, especially in comparison to traditional CPUs and GPUs. With the Transformer's self-attention mechanism requiring significant

computational resources, FPGAs can provide a power-efficient solution, making them suitable for deployment in resource-constrained environments.

5. Scalability:

FPGAs allow for scalable designs, enabling the deployment of Transformer models across a range of computational requirements. This scalability is particularly beneficial when implementing larger Transformer models for more complex tasks or when dealing with varied input data sizes.

6. High Throughput:

The remarkable parallel processing capabilities inherent in Field-Programmable Gate Arrays (FPGAs) play a pivotal role in significantly enhancing overall throughput. This capacity enables the expeditious execution of inference tasks, proving particularly advantageous in scenarios characterized by the imperative need to swiftly process a substantial volume of data. The inherent efficiency of parallelization on FPGAs not only expedites the computational workflow but also ensures that complex inference tasks can be seamlessly handled with accelerated speed and responsiveness. This heightened throughput, a direct result of FPGA's parallel processing prowess, establishes them as a robust solution for addressing the challenges associated with processing large datasets in a timely and efficient manner. In applications where rapid data processing is of paramount importance, the incorporation of FPGAs stands out as a strategic choice, exemplifying their ability to deliver not just speed but also the agility required to meet the demands of data-intensive scenarios.

7. Cost-Effective Hardware Acceleration:

While ASICs (Application-Specific Integrated Circuits) can offer high performance, they are expensive to design and manufacture. FPGAs provide a more cost-effective solution for hardware acceleration, offering a good balance between performance and development cost.

8. Adaptability to Future Architectural Improvements:

As the Transformer model evolves and new architectural improvements are introduced, FPGAs provide a platform that can be easily reconfigured to accommodate these changes without the need for a complete hardware overhaul.

II. Motivation

The limitations of GPUs in effectively accelerating transformer models emphasize the critical need to explore alternative platforms, with FPGAs emerging as a highly promising solution. FPGAs demonstrate unparalleled efficiency in the implementation of transformers, boasting superior memory bandwidth capabilities that significantly augment overall performance. This marks a pivotal shift in the pursuit of optimized hardware for transformer architectures, positioning FPGAs as a compelling choice to overcome the challenges posed by traditional GPU-based acceleration.

Furthermore, the dynamic nature inherent in the evolution of transformer models mandates platforms capable of swift adaptation to changes. FPGAs stand out against GPUs in this regard, offering a unique advantage in terms of reconfigurability. This attribute ensures optimal utilization, particularly when confronted with evolving model architectures, enabling seamless adjustments for peak efficiency. The growing scarcity and escalating costs associated with GPUs further underscore the appeal of FPGAs. As GPUs grapple with high demand and supply shortages, their cost-effectiveness diminishes. In stark contrast, FPGAs not only emerge as a more economical alternative but also empower users to deploy transformer models without succumbing to the constraints imposed by GPU availability. This dual advantage positions FPGAs as a robust solution for the ever-changing landscape of transformer model development.

Moreover, the inherent flexibility to scale down transformer models and accommodate smaller FPGAs enhances their suitability for embedded applications. This scalability feature not only underscores the adaptability of FPGAs but also positions them as a versatile and pragmatic choice for deploying transformers in resource-constrained environments. This unique capability extends the appeal of FPGAs beyond conventional computing scenarios, offering a tailored solution that aligns seamlessly with the demands of diverse and space-limited applications.

III. Proposed approach

To initiate our implementation, we adopt a multifaceted strategy that prioritizes offloading the Input and Output embedding matrices onto the FPGA, thereby unlocking superior performance capabilities. Our emphasis extends to tactically harnessing the inherent parallelism within the Feed Forward and Multi-Head Attention layers, with the overarching goal of optimizing their computational efficiency. The acceleration approach we employ involves a meticulous layer-by-layer enhancement, seamlessly integrating the FPGA-based accelerator into the Python/C++ code executed on the ARM core of the PYNQ board.

Throughout this transformative process, our paramount objective is to maintain a BLEU score comparable to that achieved by the native Transformer, ensuring that performance enhancements do not compromise model accuracy. Within the constraints of the Zynq board, we undertake a judicious downsizing of the Transformer architecture, accompanied by limitations on input word size to safeguard against exceeding 100% resource utilization. This thoughtful synergy between FPGA acceleration and strategic architectural adjustments is designed to strike an optimal balance, effectively enhancing the overall performance of the Transformer model on the designated platform. Our approach not only aims for computational efficiency but also underscores a meticulous consideration of resource constraints, thereby ensuring a robust and well-rounded enhancement of the Transformer's capabilities in real-world applications.

IV. EXPECTED RESULTS

First and foremost, our primary objective is to develop Intellectual Property (IP) blocks specifically designed for both the feed-forward layer and the Multi-Head Attention layer within the Transformer model. This intricate process involves a meticulous approach to crafting specialized components that optimize the performance of these layers, emphasizing efficiency and tailored functionality.

Additionally, our overarching goal extends beyond the creation of these IP blocks; we aspire to uphold the existing BLEU score, ensuring that the accelerated Transformer maintains the same level of language translation quality. This dual focus on acceleration and preservation of translation quality underscores our commitment to delivering advancements in computational efficiency without compromising the fundamental efficacy of the model.

Furthermore, we anticipate that the downscaled version resulting from our efforts will exhibit superior power efficiency, leveraging the inherent advantages of Field-Programmable Gate Arrays (FPGAs) in comparison to Graphics Processing Units (GPUs). The inherent power efficiency of FPGAs positions them as a compelling choice for computational tasks, particularly in scenarios where energy consumption is a critical consideration.

Moreover, we envision that the implementations developed in this endeavor possess a high degree of adaptability and reusability. These optimized IP blocks, tailored for the feed-forward and Multi-Head Attention layers, can be readily repurposed or modified for the enhancement of other architectures sharing the foundational Transformer model. This foresight not only streamlines the development process for future projects but also contributes to the broader applicability and sustainability of our innovations in the realm of advanced neural network architectures.

V. TASK DIVISION

This research endeavor involves a multi-faceted approach to accelerae transformer models using FPGA-based high-level synthesis (HLS). To lay the groundwork for FPGA acceleration, our first task is to transcend the prevalent paradigm of transformer model implementations predominantly in Python using the PyTorch framework. Instead, we pioneer a shift by implementing the transformer model in C++, setting the stage for seamless integration with HLS.

To comprehensively understand the performance bottlenecks in the context of real-world deployment, our second task involves running the C++-based transformer model on an ARM core. This provides valuable insights into the intricacies of execution, enabling a detailed analysis of both the results and the code structure. Through this analysis, we aim to unearth potential optimization opportunities and refine the implementation for optimal efficiency.

Subsequently, leveraging Xilinx pragmas and employing innovative acceleration methods become pivotal tasks in our pursuit of enhancing transformer model performance. The integration of Xilinx pragmas, coupled with other acceleration techniques, aims to unlock the full potential of FPGA-based acceleration. By strategically applying these optimizations, we endeavor to augment the computational prowess of the transformer model and significantly enhance its speed and efficiency.

In essence, this paper presents a holistic exploration, ranging from the fundamental shift in implementation language to the fine-tuning of code and execution on ARM cores, culminating in the deployment of Xilinx pragmas and advanced acceleration techniques. This comprehensive methodology is poised to make a substantive contribution to the field by advancing the state-of-the-art in transformer model acceleration.