Proposal of an open-source accelerators library for inference of transformer networks in edge devices based on Linux

Propuesta de biblioteca de aceleradores de código abierto para inferencia de redes Transformer en dispositivos perimetrales

Alejandro Araya-Núñez[1], Justin Fernández-Badilla[2], Daniel González-Vargas[3], Jimena León-Huertas[4], Erick-Andrés Obregón-Fonseca[5], Danny Xie-Li[6]

Araya-Núñez, A; Fernández-Badilla, J; González-Vargas, D; León-Huertas, J; Obregón-Fonseca, E.A; Xie-Li, D. Proposal of an open-source accelerators library for inference of transformer networks in edge devices based on Linux. *Tecnología en Marcha*. Vol. 37, special issue. June, 2024. IEEE Latin American Electron Devices Conference (LAEDC). Pág. 118-125.

https://doi.org/10.18845/tm.v37i5.7225

1 Electronic Engineering School. Costa Rica Institute of Technology. Costa Rica.
ale6896@ieee.org
https://orcid.org/0009-0003-4741-303X
2 Academic Area of Computer Engineering. Costa Rica Institute of Technology. Costa Rica.
jfernandez@ieee.org
https://orcid.org/0009-0000-0350-1562
3 Electronic Engineering School. Costa Rica Institute of Technology. Costa Rica.
danielg1010@ieee.org
https://orcid.org/0009-0006-0994-0578
4 Academic Area of Computer Engineering. Costa Rica Institute of Technology. Costa Rica.
mena.leon@ieee.org
https://orcid.org/0009-0005-2730-6282
5 Academic Area of Computer Engineering. Costa Rica Institute of Technology. Costa Rica.
erickof@ieee.org
https://orcid.org/0009-0007-1046-0894
6 Computing Engineering School. Costa Rica Institute of Technology. Costa Rica.
dxie@ic-itcr.ac.cr
https://orcid.org/0000-0003-1878-9460

Keywords

Artificial intelligence; driver; FPGA; hardware accelerator; Linux; transformers.

Abstract

Transformers networks have been a great milestone in the natural language processing field, and have powered technologies like ChatGPT, which are undeniably changing people's lives. This article discusses the characteristics and computational complexity of Transformers networks, as well as, the potential for improving its performance in low-resource environments through the use of hardware accelerators. This research has the potential to significantly improve the performance of Transformers in edge and low-end devices. In addition, Edge Artificial Intelligence, Hardware Acceleration, and Tiny Machine Learning algorithms are explored. The proposed methodology includes a software and hardware layer, with a Linux-based minimal image built on top of a synthesized RTL. The proposal also includes a library of hardware accelerators that can be customized to select the desired accelerators based on the device's resources and operations to be accelerated.

Palabras clave

Inteligencia artificial; driver; FPGA; acelerador por hardware; Linux; transformers.

Resumen

Las redes de Transformers han sido un gran hito en el campo del procesamiento del lenguaje natural y han impulsado tecnologías como ChatGPT, que indudablemente están cambiando la vida de las personas. Este artículo discute las características y la complejidad computacional de las redes de Transformers, así como el potencial para mejorar su rendimiento en entornos con pocos recursos mediante el uso de aceleradores de hardware. Esta investigación tiene el potencial de mejorar significativamente el rendimiento de los Transformers en dispositivos de *edge* y de gama baja. Además, se exploran la Inteligencia Artificial en el *edge*, la Aceleración de Hardware y los algoritmos de *Tiny Machine Learning*. La metodología propuesta incluye una capa de software y hardware, con una imagen mínima basada en Linux construida sobre un nivel de transferencia de registro (RTL) sintetizada. La propuesta también incluye una biblioteca de aceleradores de hardware que se puede personalizar para seleccionar los aceleradores deseados según los recursos del dispositivo y las operaciones a acelerar.

Introduction

The Transformers' architecture has achieved dominant results in various natural language processing (NLP) tasks. Usually, Transformer architectures have been trained on large GPU clusters. However, their quadratic computational complexity limits their usage in low-resource environments. One of the characteristics is that they process each input in parallel. This means each token of a sequence is stored simultaneously, which reduces execution time. An embedding is used to make a numerical representation of each token and in a positional codification module to indicate the relative position of every token in the sequence. The previous information is sent sequentially to six encoders. Each of them has an attentional module, with three matrices involved: query, key, and value vector. Attentional modules are used to analyze the text sequence and find relations among various words. To know the weight of each token in relation to the model, sinusoidal functions and a probabilistic function called Softmax are used [1].

Some important operations within the Transformer architecture include the scaled dot-product attention, which involves taking the dot product of the query, key, and value matrices within the attention mechanism. Transformers also require the normalization of certain data among the architecture, as it is used commonly in deep neural networks (DNN) to address gradient problems, leading to faster convergence. The multi-head attention mechanism involves the execution of the dot product attention among different dimensions, which allows the model to attend to more information given at different positions. Transformers are based on an encoder-decoder architecture. Generative pre-trained models (GPT) can be based only on decoder-only architecture, although some models might be based on different approaches depending on the task being performed [1, 2]. It uses stacked self-attention and point-wise, fully connected layers for both the encoder and decoder [1], shown in Figure 1.

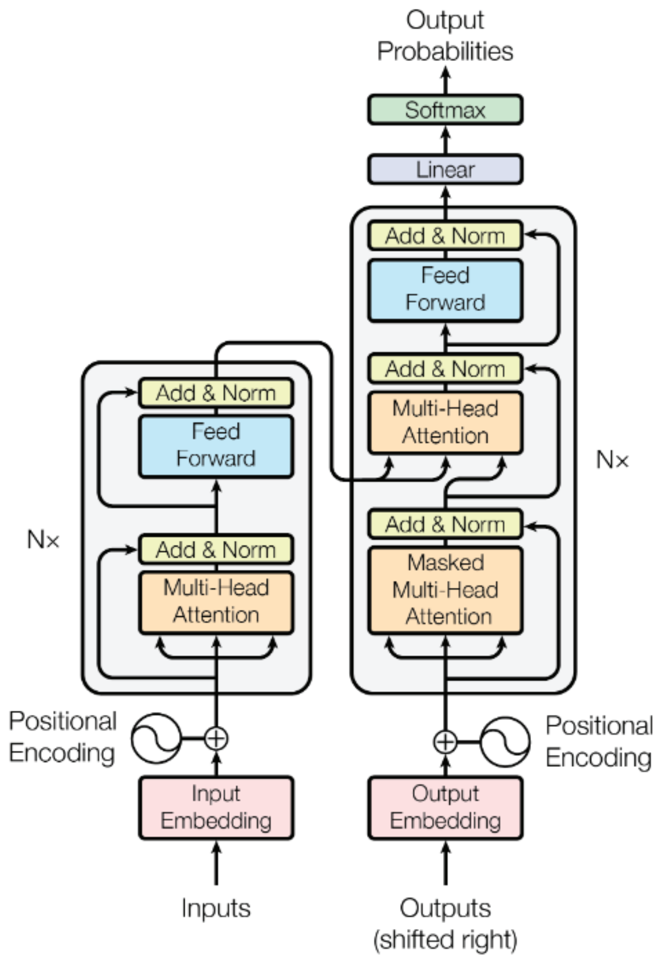

Figure 1. The Transformer – model architecture. Retrieved from Attention is all you need [4].

Hardware acceleration is a process where applications or systems delegate certain tasks to specialized hardware, giving more performance to the CPU and releasing the latter from that load [3]. The acceleration of DNN training is computationally expensive, requiring fast and efficient hardware acceleration [4]. Edge devices have limited computational capabilities, so energy-efficient accelerators and processors are needed. Lowering the access to external memory is challenging, but modern algorithms have reduced this access and others have focused on model parameters' reduction, like weight quantization and pruning [5, 6, 7]. The biggest constraint on the performance of inference accelerators is the limited bandwidth of the memory. Also, the GPU power consumption surpasses the power budget of a stand-alone embedded system. Since this, reductions in the precision of input data and hardware weight have been explored in the research field to reduce hardware accelerators' power consumption [8]. Tiny machine learning (Tiny ML) algorithms are a fusion of machine learning (ML) and the Internet of Things (IoT). This field is mostly used in edge computing, where systems have constraints in memory, power consumption, and computation time. This requires the utilization of approximation techniques, which are grouped into three families: pruning of processing layers, quantization of parameters, and activations with limited precision of binary parameters [9].

Related Work

Hardware Accelerators for Transformers

Numerous hardware accelerators have been proposed for Convolutional Neural Networks (CNN) model inference [14, 15]. However, there has been limited research on Transformer accelerators. In one study [10], authors introduced a hardware accelerator for the multi-head attention Resblock and the position-wise feed-forward network Resblock layers. This approach efficiently partitions large matrices to share hardware resources, optimizes the nonlinear functions, and achieves high hardware utilization using a systolic array. The hierarchical pruning framework in [13] presents a hierarchical pruning framework that optimizes the sparse matrix storage format to reduce memory usage for FPGA implementation. The framework's goal is to select the best device among multiple options for deploying a model while satisfying latency and accuracy constraints.

To address the computation and memory demands of transformers, a Sparse Transformer accelerator has been developed [11]. This accelerator utilizes a sparsity structure and features a specialized computing engine capable of handling sparse matrix multiplications. It includes a scalable softmax module to minimize latency from off-chip data communication. In the context of vision applications, a row-wise scheduling technique efficiently executes the major operations by treating them as a single dot product primitive. This approach promotes weight sharing in columns, leading to data reuse and reduced memory usage. Furthermore, it leverages a low gate count and SRAM buffer for improved performance [12].

Hardware Back-end

The hardware where the transformer model is running take a significant role to increase the time response of the Edge device. Different hardware backends such as GPUs and FPGAs can help accelerate the DNN processing. For the present study, FPGAs are the hardware backend due to the architectural flexibility that allows the design to handle multiple instructions [16].

Methodology

Modules

In general, the project has two layers. The first layer is the hardware layer. A minimal system is synthesized with memory and CPU, and the hardware accelerators selected by the user. The other layer is the software. A Linux-based minimal image is built on the synthesized RTL. Inside it, the Transformer model and API are added to the Linux image using Yocto recipes, as well as a driver for each synthesized accelerator instance, which is loaded in the Linux kernel. The model interacts directly with the API to transmit each layer's inputs and get each output result from it. The API is also in charge of scheduling the data transmission between the model and the different hardware accelerators, in the most efficient possible way. This will distribute the processing load in all the accelerators to take advantage of the available resources. In Figure 1, the different components of the projects are shown, as well as in Figure 2, the tools used for each part are also shown.

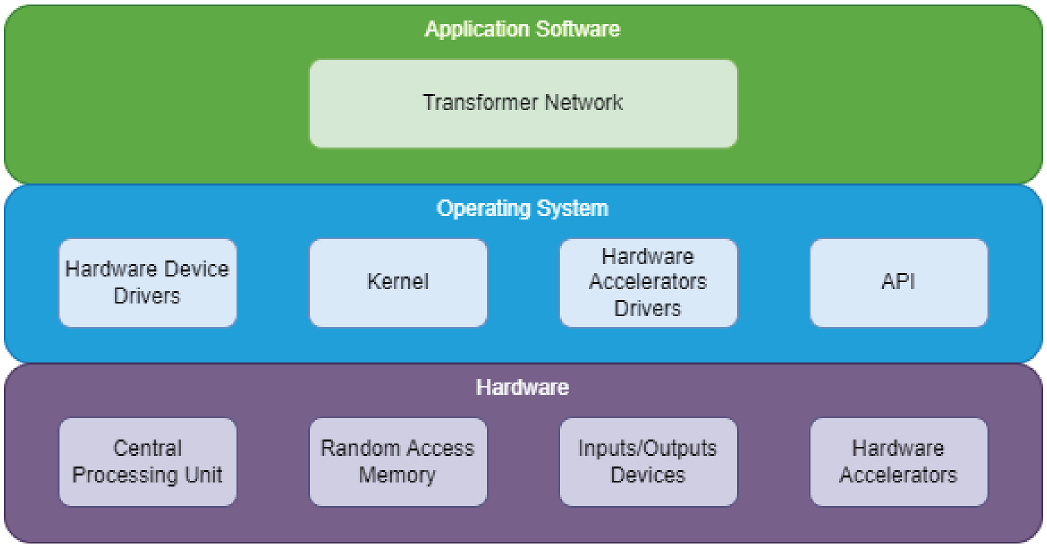

Figure 2. Project modules for the proposed architecture.

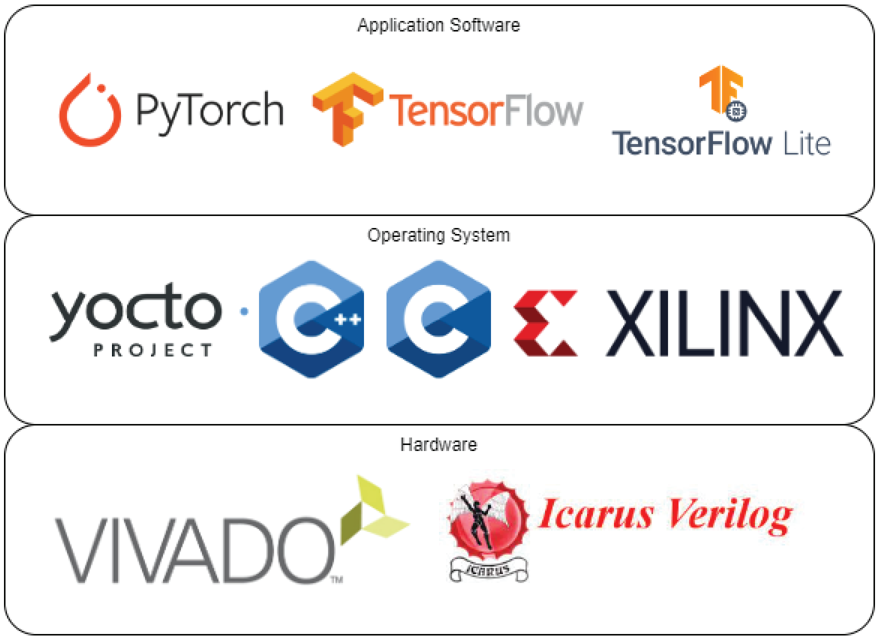

Figure 3. Project tools for each module.

API

The API is required to initialize the run time environment or context, which starts detecting the available hardware accelerators and allocates the memory for each of them. To send commands to the hardware accelerator is a requirement to define a command queue in which the context and device need to be set.

After that, allocating memory for the inputs and outputs is needed for each layer of the Transformer model. For these tasks, buffer objects are created and during the process of creation, the size of the buffer must be specified, based on the inputs and outputs size. With a buffer created, it is possible to write or read from memory to the CPU. A parameter should be set as true or false according to the task.

To process the data for the layer of the model, the CPU's buffer could be copied to the hardware accelerator to get a result to be returned to the model by doing the same process in the opposite direction. The API is in charge of copying a source to a destination buffer without losing the data to get a result to return to the Transformer model.

For the ending of the execution, the API must deallocate memory that was previously allocated by buffers. Furthermore, it is necessary to release the command queue and the environment that was created for the instance of the accelerator.

Hardware accelerators library

The hardware accelerators library consists of several accelerator instances that can be chosen and synthesized from a wide range of high-end to low-end FPGAs. A hardware accelerator is implemented for each operation of the Transformer model. Thanks to the high parallelization of this network architecture, multiple instances of the same accelerator can be synthesized by the library. This will let the user decide which operations can have more or null accelerators, and prioritize resources for a wide range of FPGAs.

Linux Drivers

A Linux driver is developed per hardware accelerator, and each instance has its own driver to transmit the data efficiently. When the Linux image is built, the accelerator instance is added with its corresponding driver loaded in the kernel. The API will take care of scheduling the data through the different hardware accelerators to take advantage of the resources properly.

Future work

The complete implementation of the proposed architecture is planned for future work. The design and implementation of the hardware accelerator library will leverage the state-of-the-art techniques explored in this research. In addition, the definition of validation tools for RTL will be explored later on, covering different open-source tools for static and dynamic verification to be used in the best scenarios.

Additionally, an extensive exploration of optimization strategies for the hardware accelerators will be conducted. These optimizations aim to enhance resource utilization and minimize power consumption. Moreover, efforts will be made to develop efficient device drivers that reduce communication overhead. These optimizations will contribute to the overall performance and efficiency of the hardware accelerators.

References

[1] A. Vaswani et al. "Attention is All You Need". 31st International Conference on Neural Information Processing Systems, Long Beach, California, 2017, pp. 6000–6010.

[2] OpenAI, "GPT-4 Technical Report," [Online], Mar 15 2023. Available: https://doi.org/10.48550/arXiv.2303.08774

[3] U. Farooq. (2021). What Is Hardware Acceleration and When Should You Use It? [Online]. Available: https://www.makeuseof.com/what-is-hardware-acceleration/

[4] A. N. Mazumder et al. (2021, Dec). "A Survey on the Optimization of Neural Network Accelerators for Micro-AI On-Device Inference". IEEE Journal on Emerging and Selected Topics in Circuits and Systems [Online]. Vol. 11, issue 4, pp. 532-547. Available: https://doi.org/10.1109/JETCAS.2021.3129415

[5] J. Lee and H. J. Yoo. (2021, Oct). "An Overview of Energy-Efficient Hardware Accelerators for On-Device Deep-Neural-Network Training". IEEE Open Journal of the Solid-State Circuits Society [Online]. Vol. 1, pp. 115-12. Available: https://doi.org/10.1109/OJSSCS.2021.3119554

[6] H. Yang and X. Lingao. (2023). "Structured Pruning for Deep Convolutional Neural Networks: A survey" [Online]. Available: http://doi.org/10.48550/arXiv.2303.00566

[7] Y. Hancheng, B. Zhang, T. Chen, and J. Fan. (2023, March). "Performance-aware Approximation of Global Channel Pruning for Multitask CNNs" [Online]. Available: https://doi.org/10.1109/TPAMI.2023.3260903

[8] H. Park and S. Kim, "Chapter Three – Hardware accelerator systems for artificial intelligence and machine learning," in Hardware Accelerator Systems for Artificial Intelligence and Machine Learning, S. Kim and G. C. Deka, Eds. Amsterdam: Elsevier, 2021, pp 51-95.

[9] S. Disabato and M. Roveri. (2022, Dec). "Tiny Machine Learning for Concept Drift". IEEE Transactions on Neural Networks and Learning Systems [Online]. Available: https://doi.org/10.1109/TNNLS.2022.3229897

[10] S. Lu *et al*, "Hardware accelerator for multi-head attention and position-wise feed-forward in the transformer," in *2020 IEEE 33rd International System-on-Chip Conference (SOCC),* 2020, .

[11] C. Fang et al, "An efficient hardware accelerator for sparse transformer neural networks," in 2022 IEEE International Symposium on Circuits and Systems (ISCAS), 2022, .

[12] H. Wang and T. Chang, "Row-wise accelerator for vision transformer," in 2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS), 2022, .

[13] P. Qi et al, "Accelerating framework of transformer by hardware design and model compression co-optimization," in 2021 IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2021, .

[14] L. Bai, Y. Zhao and X. Huang, "A CNN accelerator on FPGA using depthwise separable convolution," *IEEE Transactions on Circuits and Systems II: Express Briefs,* vol. 65, *(10),* pp. 1415-1419, 2018.

[15] A. Kyriakos *et al*, "High performance accelerator for cnn applications," in *2019 29th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS),* 2019, .

[16] Z. Azad, R. Sen, K. Park, and A. Joshi, "Hardware Acceleration for DBMS Machine Learning Scoring: Is It Worth the Overheads?," presented at the - 2021 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2021, pp. 243–253, doi: 10.1109/ISPASS51385.2021.00047.

TEC | Tecnológico de Costa Rica

LAEDC2023

Proposal of an Open-Source Accelerator Library for Inference of Transformer Networks in Edge Devices based on Linux

Araya-Nuñez Alejandro ale6896@ieee.org
González-Vargas Daniel danielg1010@ieee.org
Fernández-Badilla Justin jfernandez@ieee.org
León-Huertas Jimena mena.leon@ieee.org
Obregón-Fonseca Erick erickof@ieee.org
Xie-Li Danny dnnxl@ieee.org

Instituto Tecnológico de Costa Rica

ABSTRACT

Transformers networks have been a great milestone in the natural language processing field, and have powered technologies like ChatGPT, which are undeniably changing people's lives. This article discusses the characteristics and computational complexity of Transformers networks, as well as, the potential for improving its performance in low-resource environments through the use of hardware accelerators. This research has the potential to significantly improve the performance of Transformers in edge and low-end devices. In addition, Edge Artificial Intelligence, Hardware Acceleration, and Tiny Machine Learning algorithms are explored. The proposed methodology includes a software and hardware layer, with a Linux-based minimal image built on top of a synthesized RTL. The proposal also includes a library of hardware accelerators that can be customized to select the desired accelerators based on the device's resources and operations to be accelerated.

Keywords. *Artificial Intelligence; Driver; FPGA; Hardware Accelerator; Linux; Transformers.*

I INTRODUCTION

Multi-Head Attention is the key component of transformer architecture. It allows the model to attend to different parts of the input sequence. The number of operations required to compute the attention weights grows quadratically with the length of the input sequence.

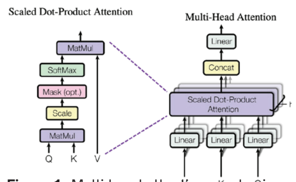

Figure 1. Multi-head attention mechanism. Recovered from [1].

III FUTURE WORK

- Designing and implementing the hardware accelerator library.
- Defining validation tools and methodology for RTL.
- Exploration of optimizations for the hardware accelerators to enhance resource utilization and reduce power consumption.
- Investigating optimization techniques to minimize the communication overhead in drivers.

II METHODOLOGY

The project consists of two layers:

1. **Hardware layer**: A minimal system is synthesized with memory and CPU, allowing users to select their desired hardware accelerators.
2. **Software layer**: A Linux-based minimal image is constructed using the synthesized RTL. Within this image, the Transformer model and API are integrated via Linux image using Yocto recipes. Additionally, a driver for each synthesized accelerator instance is loaded in the Linux kernel. The model interacts directly with the API to transmit each layer's inputs and get the corresponding output. The API also handles the scheduling of the data transmission between the model and the various hardware accelerators, ensuring optimal efficiency. This approach distributes the processing load across all the accelerators, effectively utilizing the available resources.

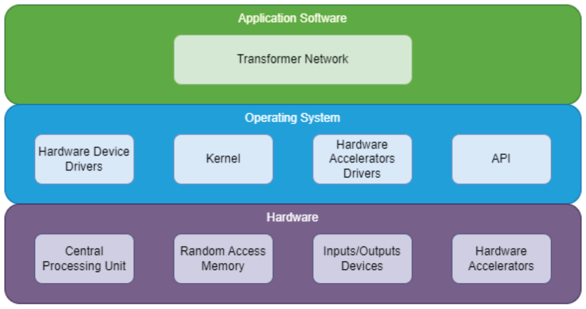

Figure 2. Project modules for the proposed architecture.

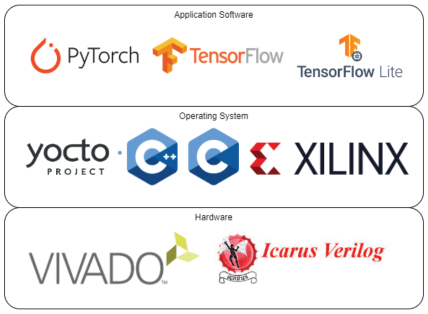

Figure 3. Project tools for each module.

REFERENCES

[1] A. Vaswani et al. "Attention is All You Need". 31st International Conference on Neural Information Processing Systems, Long Beach, California, 2017, pp. 6000–6010.

IEEE Advancing Technology for Humanity

2023 IEEE Latin American Electron Devices Conference (LAEDC)

Electron Devices Society