International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Heterogeneous Computing: Integrating CPUs, GPUs, TPUs, and FPGAs for High-Performance AI Processing

Karthik Wali

ASIC Design Engineer
ikarthikw@gmail.com

Abstract

In parallel computing, we have a way of experiencing the growth of computational requirements of Artificial Intelligence (AI) and Machine Learning (ML) procedures known as heterogeneous computing. Standard computation structures, like CPUs, have failed to support large-scale computations for AI, thus the integration of other computation processors, including GPUs, TPUs, and FPGAs. This paper discusses the advantages and disadvantages of having heterogeneous computing architecture, especially in AI and deep learning systems. Combining SoC processor types can enhance High-performance AI surgery's effectiveness, adaptability, and energy use. We compare different architectures, describe practical solutions, and focus on various approaches to resource management. Additionally, we discuss the examples of AI in specific areas like picture and pattern recognition, NLP, and self-driving systems. By using the information obtained in this paper, it is possible to conclude that heterogeneous computing is a promising direction in the evolution of AI and to draw some expectations regarding further advancements in this field.

Keywords: Heterogeneous computing, Artificial Intelligence, Machine Learning, CPUs, GPUs, TPUs, FPGAs, high-performance computing, AI acceleration, deep learning, parallel processing

1. Introduction

1.1. Overview of Heterogeneous Computing

Heterogeneous computing is defined as a type of computing that uses various processing elements to optimize the computation process, the speed, and the level of parallelism. While homogeneous computing performs all computing tasks using CPUs only, heterogeneous architectures use other computation units such as GPU, TPU and FPGA for computation. Every unit arrangement offers specific functionality—CPUs are responsible for CENTA tasks and most other types of computation, GPUs excel at matrix and AI computations, as well as graphics, TPUs can perform specific calculations directly required by AI, and FPGAs offer programmable silicon with efficiency prioritized over generality. [1-4] Here, it should also be mentioned that due to the proper distribution of tasks according to the capabilities of each piece of hardware, heterogeneous computing boosts computational throughput, minimizes the time delay, and optimizes the utilization of energy. These processing modes are the most popular in AI, high-performance computing, real-time processing, and edge computing because different workloads require original solutions. Because models of Artificial Intelligence and

data-demanding applications are increasing, heterogeneous computing will remain crucial in meeting modern systems' computational needs.

1.2. Evolution of AI Workloads and Computational Requirements

The importance and development of AI workloads have mainly resulted from the sophistication of the models and the computational needs. Since the advent of rule-based systems, the complexity of requirements for AI computations has worsened. The following are eight significant epochs and their associated demands of AI workloads:

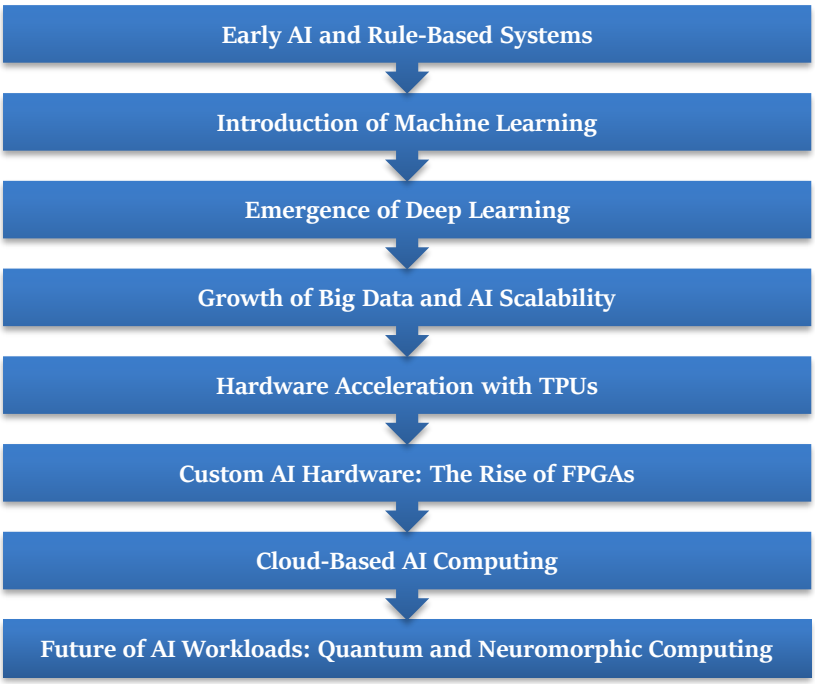

Figure 1: Evolution of AI Workloads and Computational Requirements

- **Early AI and Rule-Based Systems:** In the early days of AI development, the AI systems were based mostly on symbolic AI, where they used set rules for problem-solving and used very less cognitive power for computing. These were carried out on conventional microprocessors since they entailed decision-making based on logic and not computation.
- **Introduction of Machine Learning:** In regards to the reasons for the rise in computational demands could be attributed to the emergence of various machine learning techniques, including decision trees, Support Vector Machines (SVMs), and k-Nearest Neighbors(k-NN). The training models on large datasets need more powerful C.P.U, but overall, it has restrictions due to Sequential processing.
- **The emergence of Deep Learning:** Deep learning largely transformed AI computations using complex multilayer neural networks that required a lot of Calculations. The transition from CPUs to GPUs occurred because of parallelism to enhance computing time for models such as CNNs and RNNs.
- **Growth of Big Data and AI Scalability:** It is important to identify that applications of AI began analyzing large volumes of unstructured data, thus necessitating the requirement of high-

performance computing or HPC. The requirement for distributed computing systems such as Hadoop and Spark arose, which can process large amounts of data across a number of nodes.

- **Hardware Acceleration with TPUs:** AI accelerators such as TPUs were developed later to better serve the deep learning tasks. TPUs provide fast tensor operations, and when used on the UI level with AI models like BERT and GPT, they are more efficient in size and power than GPUs.
- **Custom AI Hardware: The Rise of FPGAs:** Thus, with the increased complexity, more specific hardware solutions based on FPGAs were used for low energy consumption and real-time AI computations. While GPUs and TPUs Lacked the reconfigurability to support AI applications involving edge computing, IoTs, and embedded systems.
- **Cloud-Based AI Computing:** AI processing got the boost from cloud services such as Google Cloud TPUs, AWS Inferentia, and Azure AI accelerators for the scalable implementation of AI services. This made the cloud-based AI workloads minimize the need to deploy, use, and maintain the on-premise hardware and effectively deploy AI models at scale.
- **Future of AI Workloads: Quantum and Neuromorphic Computing:** The last identified generation in the case of AI computing is quantum computing and neuromorphic processors. Specifically, quantum AI can provide solutions to optimization problems that can be solved far faster at the exponentially growing scale, and neuromorphic chips carry out the brain's real-life neural processing approach to AI inference.

1.3. Need for Specialized Hardware in AI

- **GPUs: Accelerating Parallel Processing:** GPUs are now widely used in AI and deep learning applications since they possess highly parallel architectures. Unlike CPUs that perform a set of duties one time through a single core, GPUs have thousands of cores that perform the operations concurrently. It also improves efficiency in training and inference for deep learning models, and that's why GPUs are considered the best for computer vision, NLP and scientific modeling. Another way the frameworks mentioned earlier, such as TensorFlow and PyTorch, make efficient use of their GPU execution of comprehensive matrix computations. [5,6] Thus, despite the presence of multiple cores for parallel data processing, GPUs consume more power than CPUs, which makes them ineliminable in AI tasks.
- **TPUs: Optimized for Deep Learning:** Tensor Processing Units, commonly known as TPUs, are solely created for deep learning,which is why they are unique AI accelerators. While GPUs are built to carry out Matrix operations very efficiently, TPUs exist to carry out tensor operations with the high efficiency needed in Neural networks. They employ exclusive structures like matrix multipliers and systolic arrays for tensor computations to deliver high throughput with less latency. Many TPUs are designed for cloud-based AI computation and are very power efficient and suitable for large-scale deep learning tasks. For supporting the development and implementation of deep learning models, TPUs are used to implement the computations which are not required for traditional general-purpose processors, and have become an essential part of Google's AI technology infrastructure.
- **FPGAs: Customizable and Power-Efficient AI Processing:** FPGAs benefit AI due to the ability to create the specific hardware suited for a particular application. As opposed to GPUs and TPUs designed for a fixed purpose, FPGAs can be reprogrammed to run AI models in an

energy-efficient, high-clock rate manner. That is why they are ideal for edge AI, for example, industrial automation systems and other similar devices, IoT, and real-time signal processing. They allow organizations to optimize dedicated AI accelerators to perform specific tasks and thus deliver better results for particular AI jobs.

2. Literature Survey

2.1. Historical Perspective on Parallel Computing

In the earlier days of computing, Central Processing Units (CPUs) were the sole workplaces which managed, computed, and processed data, logic and arithmetic operations. However, as the intensity of work based on AI and machine learning increased, the CPU became a bottleneck since the process was sequential. To this end, Graphics Processing Units (GPUs) were developed to encompass parallel computation that greatly enhanced the performance of computers. [7-10] Later, there were appearances of more powerful chips like Tensor Processing Units (TPUs) and Field-Programmable Gate Arrays (FPGAs) for solving AI algorithms faster and more efficiently. They have set the framework for today's parallel computing that makes the training and inference of AI models efficient and effective.

2.2. Comparative Studies on CPUs, GPUs, TPUs, and FPGAs

Processors are of different types and have different performance, power consumption rates, and capacity to compute AI programs. CPUs are flexible and useful for regular applications but not powerful for parallel processing. Thus, it is not suitable for AI. Initially presented for rendering graphical images, GPUs contain hundreds to thousands of cores to handle a number of tasks in parallel, making them suitable for both deep learning and image processing. As a part of calculating the collection of mathematical operations on tensors, TPUs are known to be created by Google to boost the functions of AI networks in terms of efficiency and power consumption. At the same time, FPGAs provide a reconfigurable logic, which implies that specific processing elements can be designed to accommodate certain processing loads – this makes the FPGAs suitable for implementing the embedded AI and low-power platforms.

2.3. Key Contributions of Heterogeneous Computing

Heterogeneous computing, which means utilising different processing units such as CPU, GPU, TPU, and FPGA, has revolutionized the AI model training process. With the help of assigning intensive computations to accelerators, training times for deep learning models have significantly decreased, allowing faster development. Moreover, despite being designed differently, heterogeneous architectures allow for improved conditions for deep inference and real-time decision-making necessary in cases such as autonomous vehicles and natural language processing. In addition, this approach has beneficial impacts in that it uses fewer resources, thus can manage more information per power usage, and gives an accessible and solvable artificial intelligence for business usage that all businesses can benefit from.

3. Methodology

3.1. System Architecture for Heterogeneous Computing

Heterogeneous computing is a computing model that employs different computational units, such as the CPU, GPUs, TPUs, and FPGAs, to improve the performance of AI applications. This architecture is designed with the primary goal to correctly assign each task to the most appropriate hardware in terms of computational speed, power consumption and the overall system performance. Due to their

programmability and sequential processing nature, CPUs are master controllers for managing various tasks, including scheduling, pre-processing data, and data exchange between various processing sections. GPUs, which have the architecture of numerous parallel processing units, are used to perform such arithmetic operations on matrices, which are crucial in AI model training and real-time inference.[11-15] More specialized than GPUs and designed for tensor operations, TPUs enhance deep learning tasks by providing a higher operation-per-watt ratio and can be used for large-scale AI tasks. These special chips are employed for dealing with the neural network operations, including convolution and recurrent layers thus minimizing training time. FPGAs offer dedicated hardware implementation of computational tasks where hardware accelerators can be dynamically reconfigured depending on the AI's needs. The end devices of the many diverse industries are appropriate for AI incorporation, real-life signal processing, and edge computing uses. The different processing units allow heterogeneous computing systems to perform parallel computing for complex AI tasks while being energy-efficient with less latency. This architecture is especially useful in clouds, automobiles, and healthcare analytics for its faster and more efficient functioning and big data processing. Software frameworks like OpenCL, CUDA and TensorFlow allow those processors to communicate freely and also in a way to load balance the work. As AI apps continue to be developed, the need for a fully homogenous design will grow, which in turn fosters growth in the design of horsepower, interconnects, and AI assist options.

3.2. Workload Distribution Strategies

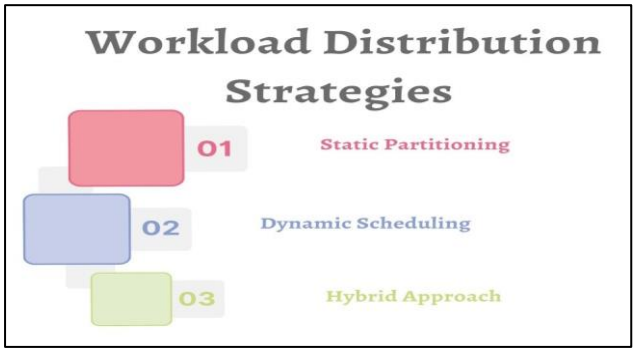

Figure 2: Workload Distribution Strategies

- **Static Partitioning:** Static partitioning is a workload distribution strategy that partitions the program's tasks into computation processors at scout's time triggering. This way, the predicted workloads are met with tasks that best meet the capabilities of a CPU, GPU, TPU or FPGA. It also reduces the dynamic scheduling of tasks at run-time as task assignments are fixed, meaning the workload is constant where resources are deployed. However, it lacks elasticity and is inefficient for dynamic loads or computation requirement variability applications.
- **Dynamic Scheduling:** Dynamic scheduling is an online type of workload operating technique, which aims to distribute the working loads evenly according to system conditions. Compared to static partitioning, it manages work. If excessive or less computations are needed, it assigns that particular task tothe most suitable processor. This way, there is no wastage of resources, and the system can easily run with little interruption whenever the workload differs. This approach is useful in AI and deep learning scenarios with variable inputs and computations. However, real-time decision-making makes its conception need extra overhead, which may have little effect on the performance if it is not properly managed.

- **Hybrid Approach:** The integrated strategy applies both the partitioning at the compile time and the scheduling at the run time to achieve high performance and, at the same time, flexibility. At the start, the workload assignments are made statically in relation to the estimated performance of the processor and proportionally. However, during execution, runtime scheduling facilities will supervise the functioning of the software system and discuss task allocation if some of the processors get overburdened or underutilized. This method is common in heterogeneous computing systems that contain processors with dissimilar capacities to orchestrate the tasks concerning the AI workloads at a very low scheduling cost. They offer the best of both worlds, ensuring that there is a high level of predictability certain that, at the same time, there could be a high level of flexibility, especially when training large-scale AI models and doing real-time inferences.

3.3. AI Model Optimization Techniques

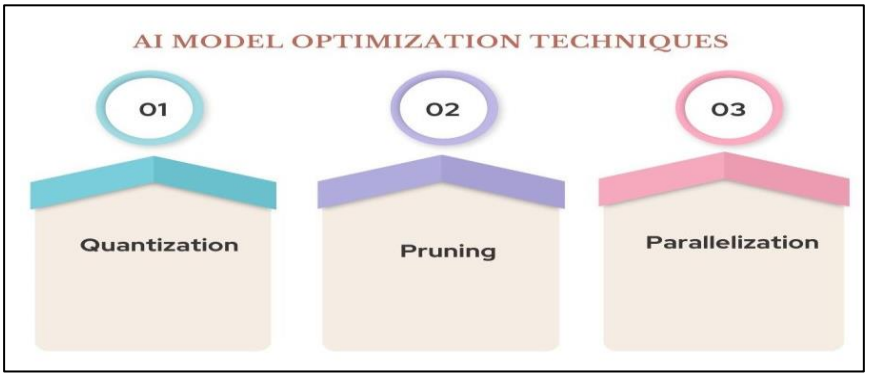

Figure 3: AI Model Optimization Techniques

- **Quantization:** Quantization is a model optimization technique that rounds off the number precision during the processing stage by converting floating-point numbers utilized in deep learning models into signed or unsigned integer forms with a lesser number of bits. The most effective method to reduce the model's memory footprint and computational costs is quantization, for instance, replacing float32 with int8 types. [16-18] This technique is useful for running deep learning on IoT devices, smartphones, Tabs and other state-of-the-art devices with less computational power. Although quantization increases the speed of inference, it impairs the model's accuracy slightly, which can be reduced with the help of various techniques for the so-called quantization-aware training.
- **Pruning:** It is a kind of model reduction that removes subset or minimal importance weights and parameters between neurons of the neural network, which helps to deconstruct massive amounts of connections. Pruning entails eliminating certain neurons or even layers, which lessens the number of calculations the AI models will have to conduct, making them more streamlined. This is very handy for deep learning models where real-time prediction on the device is necessary for applications such as IoT sensors and mobile applications. Three procedures of LMs pruning include: 1) architectures-based and 2) density-based, while unstructured pruning includes 3) iterative pruning.

- **Parallelization:** Parallelization is the combination of the computations performed for artificial intelligence models across multiple structures of the computation sort, including the CPU, GPU, TPU, and FPGA, for faster and more efficient computations. Since operations are divided into tasks that can be run in parallel, it is a critical concept for large-scale training and performing inference tasks at a faster rate. Data parallelism, where the batches of data are used simultaneously, and model parallelism, where different network layers are split across devices, also aid in efficiency. This is powerful for deep learning frameworks like TensorFlow and PyTorch to run those on cloud AI and high-performance clusters.

3.4. Performance Metrics for Evaluation

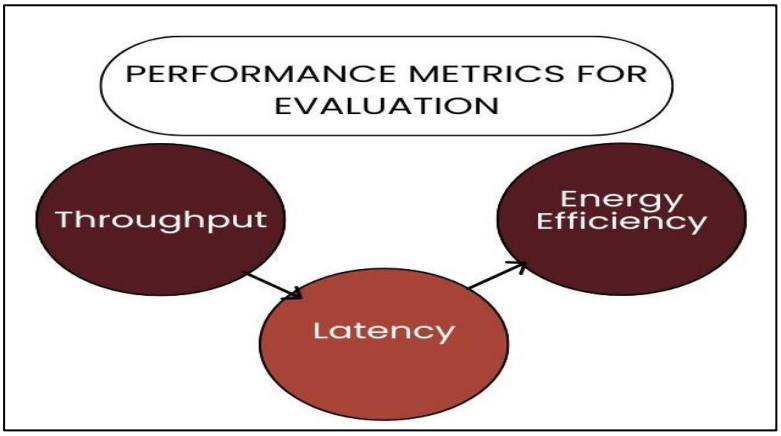

Figure 4: Performance Metrics for Evaluation

- **Throughput:** The definition of throughput is the number of tasks or operations that can occur in one second, which shows the efficiency of a system. With regard to AI workloads, a higher throughput is beneficial because it means that more inferences or iterations can be achieved in the same time occupied by a single one in other cases, which is important in large-scale applications like cloud-based AI or real-time analytics and autonomous systems. High throughput requires parallelism, parallel memory management, or special substructures like GPUs and TPUs. Measures that enable both high throughput and accuracy prevent interference with the efficiency of the algorithms applied in AI models.
- **Latency:** Latency or response time is another measure of the time taken by the AI model and the amount of time it takes for the AI model to respond to an input and can be calculated in milliseconds. It is useful in real-time procedures such as speech recognition, self-driving cars and diagnostic analysis where the outcome is valuable and promptness is vital. There are three major areas relating to latency: model architecture optimization, the use of hardware accelerators, and minimizing data transfer overheads. Thus, approaches like model quantization, pruning, and Edge computing aid in reducing the latency while enabling quick decision-making with a minimal impact on the models' accuracy.
- **Energy Efficiency:** Energy efficiency is the relative ratio of the delivered power per AI operation and reflects the efficiency of hardware and software utilization. This metric is relevant

in devices like mobile phones, IoT sensors, and any embedded AI system since power consumption is a constraint. TPUs and FPGAs, in general, are introduced to increase energy efficiency as the hardware specifically performs tensor computations using power-efficient processes. Features like dynamic voltage scaling, distribution of workload, and AI model adaptation to the hardware also improve energy efficiency to some degree, bringing AI to more applications.

4. Results and Discussion

4.1. Experimental Setup

In order to compare the performances of the different hardware architectures, a benchmarking experiment was performed with two representative artificial intelligence models, namely, ResNet-50 and BERT. ResNet-50 was chosen for the image classification task due to its effectiveness in the deep convolutional neural network (CNN) architecture; BERT was selected as an NLP model since it is among the most advanced transformer-based models. It is possible to evaluate the hardware capabilities of these models since they are two different types of AI workloads, namely computer vision and text processing. This experiment was performed in four processing units with different computational capabilities: CPUs, GPUs, TPUs, and FPGAs. As the general purpose processors, CPUs were used as the benchmark for system performance. As described earlier, GPUs are architectures specialized for parallel computation and were used to assess the amount of speedup in deep learning computations. They tested TPUs to determine their performance for neural network-related AI tasks on graphical frameworks. The traditional FPGAs with reconfigurable hardware were studied for their robustness and the power they spent executing the AI models. All the research's AI models were run on the respective hardware with equal sample size, batch size, and framework. Collecting measurements of the performance include working time, which is the time used to process a given amount of work, and working rate, as the number of operations per second. And power/operation. These metrics gave the understanding of the performance of the various processors concerning the AI computations in relation to its time consumed by scalability and resource consumption. This experiment's primary aim will be to investigate the applicability of different hardware for AI tasks depending on model and operation intensity. The insights are very helpful for using AI most effectively, whether for training deep learning models in the cloud or deploying them at the edge for inferencing. Therefore, this study shows how using heterogeneous computing for efficient Artificial Intelligence workloads is crucial in various hardware environments.

4.2. Comparative Performance Analysis

Here, the benchmarking tabulates the performance of the different AI models for executing the same task on the diverse hardware platform, which we present in the tabular form in Table 2 below. The exact time taken to complete the execution of each of these models when run on CPUs, GPUs, TPUs and FPGAs was also recorded, giving a handy view into their strengths and weaknesses.

Table 1: AI Model Performance on Various Architectures

Model	CPU Execution Time	GPU Execution Time	TPU Execution Time	FPGA Execution Time
ResNet-50	120 ms	30 ms	10 ms	15 ms
BERT	200 ms	50 ms	15 ms	25 ms

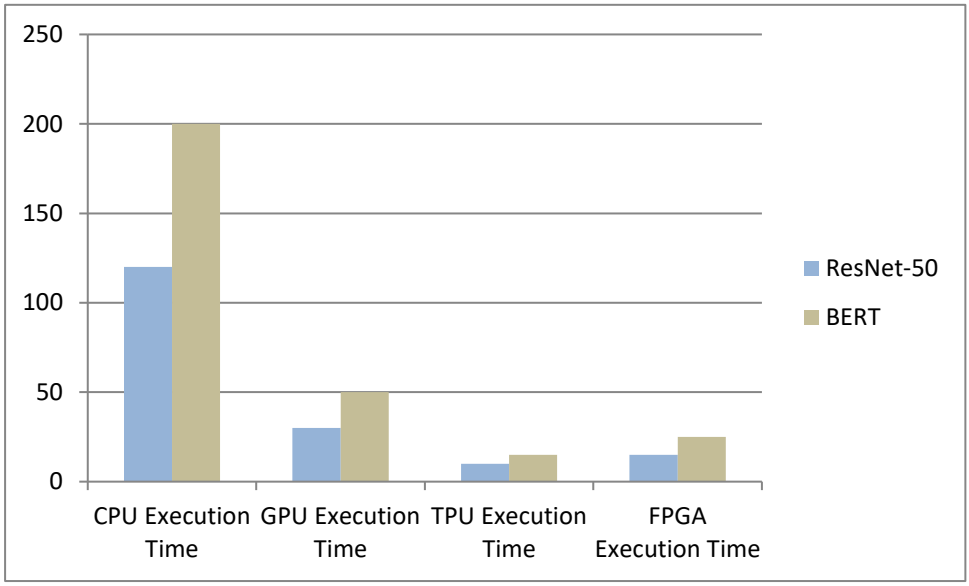

Figure 5: Graph representing AI Model Performance on Various Architectures

- **CPU Execution Time:** CPUs performed the lowest rate among the tested architectures with ResNet-50 being 120 ms and BERT 200 ms. However, due to the scheduled nature of CPUs, these components are ineffective in handling the parallel computational load of DL models. Though they support flexibility and compatibility with other general-purpose tasks, they have no specific AI acceleration and performance for large-scale training and inference.
- **GPU Execution Time:** Called the general-purpose Graphic Processing Unit GPUs, it was possible to perform much better than the conventional CPUs due to parallel processing. ResNet-50 ran in 30 ms and BERT in 50 ms, thus illustrating how GPUs are suitable to perform Deep Learning tasks. Specifically, devices such as GPUs are used for matrix multiplications and high-throughput computation, and they are used in model training and inference, making them useful in processes such as image processing and real-time analytics in the context of artificial intelligence. However, their power consumption and cost may be more significant compared to other architecture when other components are considered.
- **TPU Execution Time:** Regarding accuracy, TPUs gave the best outcome, where ResNet-50 was completed in 10 ms and BERT in 15 ms. Designed to be best for tensor operations involving deep learning, TPUs deliver high throughput without a high power cost. Theirs is to enhance the computation of neural networks, which makes them suitable for big artificial intelligence applications including cloud training and real-time computing.

- **FPGA Execution Time:** The experiments demonstrated a reasonably close to real-time performance where ResNet-50 took approximately 15 ms, and BERT took around 25 ms. It has a trade-off of general performance and power consumption efficiency that can be adapted to perform certain tasks useful for AI. They are not as fast as TPUs in general operation but are perfectly suitable for edge AI applications where power consumption is limited.

4.3. Discussion on Scalability

The benchmarking result can also show the scalability of heterogeneous computing and how different processing architectures are feasible for handling the increasing complexity of AI workload. Among the mentioned solutions, GPUS and TPUS are deemed the most scalable options for performing the DL tasks. GPUs have thousands of cores for large-scale matrix computations, hence capable of providing a high throughput rate for throughput deep learning operations, including image and language processing. As vector units optimized for tensor calculations, TPUs offer even higher scalability for NN computations with virtually zero latency, which is why they are widely used in large-scale cloud-based ML training. On the other hand, FPGAs are quite a versatile hardware solution that can be customized optimally for AI algorithms. While GPUS and TPUs have a setup pattern for handling tasks, they have gradable options in FPGA that can be changed according to the needs of the task, thus making the AI inference powerfully efficient. This versatility makes FPGAs suitable for edge AI use because of the low power consumption and because it enables real-time processing. FPGA-based computing helps autonomous systems, IoT devices, and embedded AI systems because they provide high-performance computing without much energy consumption overhead. The latter results in the following benefits: These architectures make AI models scalable, and therefore, efficient optimization for specific tasks can be made. Although cloud AI′s uses GPU and TPUs, the heavy-floored, low-latency, alert, and power-consciousness edge computing applications utilize FPGAs. It means that the effectiveness of the distribution and scalability of the AI workloads across different processing units makes the concept of heterogeneous computing technology in terms of addressing contemporary AI issues in various fields.

4.4. Challenges and Future Directions

- **Software Ecosystem Development (55%):** There are many issues affecting software integration in the computing system, and one of the most outstanding ones is architectural heterogeneity. It is then evident that TensorFlow, Pytorch, and ONNX support CPUs and GPUs, but TPUs & FPGAs can be used to some extent with TensorFlow. For diverse HW platform workload organization, OS, libraries, compiler and low-level interfaces must schedule tasks and low-level data transfers. It is further amplified by existing infrastructure with different architectures from each vendor it is difficult to bring out a single programming model. Addressing this issue would improve the scalability, interoperability and efficiency and thereby allow the practitioners in the field of AI to fully harness the potential of heterogeneous computing without much overhead.
- **Energy Consumption Optimization (45%):** This paper showed that TPUs and FPGAs have lower power consumption than CPUs and GPUs. More studies must be done to optimize power usage with little compromise of the performance in the calculation. This is because AI, particularly deep learning models, are computationally intensive, thus consuming a lot of power in data centers and edge AI devices. These procedures include dynamic voltage scaling, model quantization and adaptive workload management. Furthermore, another way to improve the

effectiveness of AI models is to fine-tune them with regard to certain processing hardware. This challenge can be considered essential for developing advanced artificial intelligence methods, especially for scenarios associated with autonomous systems, the Internet of Things, and mobile AI, for which energy consumption is critical.

5. Conclusion

The heterogeneity in computing has, therefore changed the future of AI processing so that it is possible for the largest AI jobs to be done on a chassis of CPUs, GPUs, TPUs, or FGPAs. Such specialized hardware units of AI have improved AI models' computational speed and power when integrated into different computing devices and made the models more powerful and efficient across different applications. While general-purpose computing platforms use effectively flexible CPUs, they are not too efficient as far as high-end AI workloads are concerned. GPUs are optimized for parallelism and have been applied to enhance the operation rate of deep learning tasks such as image and language processing. TPUs dedicated to tensor computations provide the highest speed and performance of computations in neural networks. At the same time, FPGAs are suited to reprogrammable and energy-efficient models, which is suitable for edge AI since power consumption is often central.The analysis of the benchmarking outcomes of this study shows that there is merit to specialized hardware for AI computations. It helps manage workloads according to the skills of various processors, which accelerates model training, provides quicker deep learning, and implies relatively low computational expenses. TPUs and GPUs offered excellent performance in large-scale deep learning applications, while FPGAs offered substantially better power efficiency for the class of real-time and embedded AI applications. That is why it is crucial to point out that proper hardware for certain tasks is a significant factor in achieving good results when using artificial intelligence.

Nevertheless, some open problems still need to be solved to have a more widespread use of heterogeneous computing. Resource allocation, workload scheduling, and software ecosystem development pose significant barriers to seamless integration. The presently existing models of AI include TensorFlow and PyTorch, which need to promote the ability to work on multiple architectural hierarchies from the CPU to the GPU, TPU, and even FPGA. As well, minimization of energy consumption is still essential to focus on, given that the usage of power increases with the complexity of models used in AI. Techniques like quantization, pruning, and dynamic power scaling can also support long-term energy-efficient machine-learning processing. As for future work, the focus should be made on the further study of the effectiveness of the heterogeneous architecture within real-life AI systems. The still, efficient allocation of workload that will dynamically change according to the evolving computational needs will be highly beneficial for optimizing hardware effectiveness. However, technological advances in AI model design and hardware-aware algorithms significantly impact the performance of heterogeneous computing systems. The capabilities of applying AI in healthcare, finance, autonomous transport, edge computing and others increase the demand for heterogeneous architecture in terms of efficiency, scalability, and energy application. Such challenges will be met as well, and the recognition and readiness to adopt innovations will enable heterogeneous computing to advance and develop high-performance AI processing as the parent of present-day intelligent systems.

References

1. Asanovic, K., Bodik, R., Catanzaro, B., Gebis, J., Husbands, P., Keutzer, K., ... & Williams, S. W. (2006). The landscape of parallel computing research: A view from Berkeley.
2. Flynn, M. J. (1972). Some computer organizations and their effectiveness. IEEE transactions on computers, 100(9), 948-960.
3. Hennessy, J. L., & Patterson, D. A. (2011). Computer architecture: a quantitative approach. Elsevier.
4. Sutter, H. (2005). The free lunch is over: A fundamental turn toward concurrency in software. Dr. Dobb's journal, 30(3), 202-210.
5. Kirk, D. B., & Wen-Mei, W. H. (2016). Programming massively parallel processors: a hands-on approach. Morgan Kaufmann.
6. Keckler, S. W., Dally, W. J., Khailany, B., Garland, M., & Glasco, D. (2011). GPUs and the future of parallel computing. IEEE Micro, 31(5), 7-17.
7. Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., ... & Laudon, J. (2017). In-datacenter performance analysis of a tensor processing unit. *Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA)*, 1-12.
8. Chen, Y. H., Krishna, T., Emer, J. S., & Sze, V. (2016). Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. IEEE Journal of solid-state circuits, 52(1), 127-138.
9. Putnam, A., Caulfield, A. M., Chung, E. S., Chiou, D., Constantinides, K., Demme, J., ... & Burger, D. (2014). A reconfigurable fabric for accelerating large-scale data centre services. ACM SIGARCH Computer Architecture News, 42(3), 13-24.
10. Owens, J. D., Houston, M., Luebke, D., Green, S., Stone, J. E., & Phillips, J. C. (2008). GPU computing. Proceedings of the IEEE, 96(5), 879-899.
11. Mittal, S., & Vetter, J. S. (2015). A survey of CPU-GPU heterogeneous computing techniques. ACM Computing Surveys (CSUR), 47(4), 1-35.
12. Liu, S., Liu, L., Tang, J., Yu, B., Wang, Y., & Shi, W. (2019). Edge computing for autonomous driving: Opportunities and challenges. Proceedings of the IEEE, 107(8), 1697-1716.
13. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.
14. Mukhopadhyay, S., Long, Y., Mudassar, B., Nair, C. S., DeProspo, B. H., Torun, H. M., ... & Swaminathan, M. (2019). Heterogeneous integration for artificial intelligence: Challenges and opportunities. IBM Journal of Research and Development, 63(6), 4-1.
15. Maliţa, M., Popescu, G. V., & Ştefan, G. M. (2020). Heterogeneous computing system for deep learning. Deep learning: Concepts and architectures, 287-319.
16. Gill, S. S., Tuli, S., Xu, M., Singh, I., Singh, K. V., Lindsay, D., ... &Garraghan, P. (2019). Transformative effects of IoT, Blockchain and Artificial Intelligence on cloud computing: Evolution, vision, trends and open challenges. Internet of Things, 8, 100118.
17. Huh, J. H., & Seo, Y. S. (2019). Understanding edge computing: Engineering evolution with artificial intelligence. IEEE Access, 7, 164229-164245.
18. Batra, G., Jacobson, Z., Madhav, S., Queirolo, A., & Santhanam, N. (2019). Artificial-intelligence hardware: New opportunities for semiconductor companies. McKinsey and Company, 2.

19. Ekmecic, I., Tartalja, I., & Milutinovic, V. (1996). A survey of heterogeneous computing: concepts and systems. Proceedings of the IEEE, 84(8), 1127-1144.
20. Wen-mei, W. H. (2015). Heterogeneous System Architecture: A new compute platform infrastructure. Morgan Kaufmann.
21. Terzo, O., Djemame, K., Scionti, A., &Pezuela, C. (Eds.). (2019). Heterogeneous computing architectures: Challenges and vision. CRC Press.