aci avances en ciencias e ingenierías

Artículo/Article
Vol. 17, nro. 1
e3707

Efficiency in the classification of chest X-ray images through generative parallelization of the Neural Architecture Search

Félix Armando Mejía Cajicá[1]*, John A. García Henao[2], Carlos Jaime Barrios Hernández [3], Michel Riveill[4]

[1] Universidad Industrial de Santander (UIS), Supercomputación y Cálculo Científico. UIS (SC3UIS), Grupo de Computación Avanzada y de Gran Escala (CAGE). Bucaramanga, Santander, Colombia.

[2] University of Bern: Bern, Switzerland.

[3] Universidad Industrial de Santander (UIS), Supercomputación y Cálculo Científico. UIS (SC3UIS), Grupo de Computación Avanzada y de Gran Escala (CAGE). Bucaramanga, Santander, Colombia.

[4]A Université Côte d'Azur, Laboratoire I3S, 06900 SA, France.

*Autor para correspondencia/Corresponding author: felix2067165@correo.uis.edu.co

Eficacia en la clasificación de imágenes de rayos X de tórax mediante la paralelización generativa de la búsqueda de arquitectura neuronal

Resumen

GenNAS permite la clasificación de radiografías de tórax en enfermedades pulmonares, aprovechando nuevos métodos de entrenamiento paralelo para una mayor precisión y eficiencia. La clasificación de imágenes médicas para patologías pulmonares a partir de radiografías de tórax tradicionalmente requiere mucho tiempo. GenNAS, utilizando las capacidades generativas de GPT-4, automatiza el aprendizaje de arquitecturas óptimas a partir de los datos. Este estudio utiliza la paralelización y los algoritmos generativos para optimizar las arquitecturas de redes neuronales para la clasificación de radiografías de tórax, analizando su impacto en el algoritmo Neural Architecture Search NAS utilizando el conjunto de datos de CheXpert. El estudio utiliza el conjunto de datos CheXpert con 224,316 radiografías de tórax para clasificar cinco patologías de enfermedades pulmonares. GenNASXRays evalúa 6561 posibilidades de arquitectura en un espacio de búsqueda de 8 capas, con gráficos AUC-ROC y Precision-Recall como métricas. El algoritmo secuencial, que se entrenó en 187,641 imágenes, tardó 190,2 horas para obtener un AUC-ROC de 0,869. En ejecución paralela en dos GPU, se logró un AUC-ROC de 0,87 en 127,09 horas, lo que resalta la eficiencia de la paralelización.

Palabras clave: Radiografías de pulmón, modelos generativos, arquitectura paralela

Abstract

Explore GenNAS for chest X-ray classification in lung diseases, leveraging novel parallel training methods for enhanced accuracy and efficiency. Medical image classification for pulmonary pathologies from chest X-rays is traditionally time-consuming. GenNAS, using GPT-4's generative capabilities, automates optimal architecture learning from data. This study investigates parallelization and generative algorithms to optimize

Editado por / Edited by:
Dennis Cazar

Recibido / Received:
25/11/2024

Aceptado / Accepted:
09/12/2024

Publicado en línea / Published online:
09/05/2025

 DOI: https://doi.org/10.18272/aci.v17i1.3707

neural network architectures for chest X-ray classification, analyzing their impact on the NAS algorithm using the CheXpert dataset. The study uses the CheXpert dataset with 224,316 chest X-rays to classify five lung disease pathologies. GenNASXRays evaluates 6,561 architecture possibilities in an 8-layer search space, with AUC-ROC and Precision-Recall plots as metrics. Training on 187,641 images, the sequential algorithm took 190.2 hours to achieve an AUC-ROC of 0.869. In parallel execution on two GPUs, an AUC-ROC of 0.87 was achieved in 127.09 hours, highlighting the efficiency of parallelization.

Keywords: Chest X-ray images, Generative Models, Parallel architecture

INTRODUCTION

Artificial intelligence (AI) has taken an important role in the field of medicine as a tool to help doctors make decisions, which is why the constant development of accurate and robust medical image classification models is necessary, especially in the diagnosis of lung pathologies from chest X-rays. Traditional medical image classification demands manual architecture design, a time-consuming process. GenNAS leverages GPT-4's generative capabilities to streamline this by learning optimal architectures directly from the data. This study investigates the potential of parallelization and generative algorithms to optimize neural network architectures for chest X-ray classification. It analyzes the impact of parallelization on the Neural Architecture Search (NAS) algorithm using the CheXpert dataset.

Indeed, parallelization is a critical technique in AI for healthcare that enables faster model training, efficient data processing, real-time decision-making, and the advancement of precision medicine. By harnessing the power of parallelization, AI systems can transform healthcare delivery and improve patient outcomes. However, it is important to highlight that parallelization and support of high-performance computing (HPC) architectures are just two aspects of AI for healthcare. These techniques are significant in medical imaging and diagnostics, virtual patient care, medical research and drug discovery, patient engagement and compliance, rehabilitation, and other administrative applications. Integrating AI into healthcare settings can revolutionize clinical decision-making, hospital operations, medical diagnostics, patient care, and more.

This study assesses the effectiveness of generative algorithms as a black box for designing optimal neural network architectures. Additionally, parallelization strategies with generative algorithms are implemented to expedite the search for accurate neural network architectures.

STATE OF ART

Several research studies utilize the capabilities of Large Language Models (LLMs), such as GPT-4, to search for neural network architectures. Thanks to the powerful generative capability of GPT-4, a pioneering study called GENIUS [1] was conducted, which uses GPT-4 to design neural architectures for convolutional neural networks (CNNs). The main

DOI: https://doi.org/10.18272/aci.v17i1.3707

idea is to allow GPT-4 to learn from the feedback of the generated neural architectures and generate better ones. Experimental results show that GPT-4 can find better-classified architectures after several repetitions of prompts. Subsequently, Automatic Machine Learning (AutoML)-GPT articles [2] design cues for GPT to automatically complete data processing tasks, model architecture design, and hyperparameter tuning. Some papers explore the mutual enhancement between LLM and AutoML articles [3].

In the present work, tests were conducted with Llama 2 and GPT-4. Among the issues with Llama 2 were that the results obtained were not formatted in a standard way, which made automation quite complex. Additionally, the model structures proposed by Llama 2 were not of the same quality as those obtained through GPT-4. GPT-4 allowed for standard formatting of outputs, and the proposed model structures were more robust, yielding better results.

HPC involves the use of multiple cores to support parallel processing. This processing can be massive or intensive. HPC supports AI in computer architecture design and organization. The well-known expression *Cambrian Explosion of Computing* for AI and Big Data [4] allows customizing computer architectures following requirements, in this case, with AI needs. Then, there are key points to observe:

- Next-Generation HPC Systems: Next-generation HPC systems and HPC architectures are raising the bar for computing and memory performance, I/O, and flexibility. These systems are designed to meet the increasing demands of HPC workloads, including AI and analytics. They draw from the cloud, supporting a diverse range of architectures, and compute models and scalings.

- Unified and Open Programming Models: Developers can use unified and open programming models, such as OneAPI, CUDA, or directives such as OpenACC to create a single codebase that can be used across different architectures, including CPUs, GPUs, and FPGAs. This approach enables more productive and performant development in the HPC and AI domains.

- Convergence of AI and HPC: The convergence of AI and HPC is an ongoing development. In real-world scenarios, AI architectures can be noisy, incomplete, and heterogeneous, and they may perform differently from benchmark results. Creating better mathematical frameworks for selecting AI architectures and optimizing their performance is an area of focus for future development.

- Hybrid Workflows: Hybrid workflows that combine traditional computing methods (i.e., simulation) with data science methods are becoming more efficient. For example, coupling legacy computational fluid dynamics (CFD) codes with AI libraries for forecasting or astrophysics can be a challenge, but advancements in HPC architectures, such as the use of GPUs or TPUs, enable the integration of AI and HPC more efficiently.

- Challenges and Trade-offs: There are challenges and trade-offs when it comes to AI and HPC configurations. AI-heavy workloads often prioritize speed over core count, while HPC workloads prefer greater compute performance with a high core count

DOI: https://doi.org/10.18272/aci.v17i1.3707

and more core-to-core bandwidth. Developers face challenges in transitioning software to function on HPC clusters and efficiently programming high-performance parallel computing. It can require a significant time investment, and compatibility across different hardware types and computing models is important.

- Specialized Processing Units Acceleration**: GPUs, TPUs, or QPUs play a crucial role in accelerating AI workloads within HPC environments. NVIDIA, for example, offers a complete end-to-end stack and suite of optimized products, infrastructure, and services for high-performance computing and AI workflows. Their full-stack architectural approach ensures optimal performance, fewer servers, and lower energy consumption, resulting in faster insights at lower costs [5]. On the other hand, we can observe the same for other companies, such as Intel. However, the more impressive option is using QPUs (Quantum Processing Units) to support AI algorithm implementations.
  Energy Sustainability: The convergence of energy sustainability and AI/HPC is an area of growing interest. Green energy and AI synergy, energy efficiency and sustainability [6], and frugality for AI/HPC implementations are popular topics in the specialized community [7].

HPC and AI revolutionize the healthcare sector, enabling faster and more accurate diagnoses. By leveraging the power of HPC and AI, healthcare professionals can improve patient care, enhance research capabilities, and optimize healthcare delivery. In computing terms, parallelization plays a crucial role in the field of AI, because it is unlikely to implement the different AI algorithms in today's efficient mode without HPC support. Observing AI for healthcare [8, 9, 10], the junction enables faster and more efficient model training, essential for developing and deploying AI models in healthcare applications. Some key points about the role of parallelization in AI for healthcare are:

- Accelerating Model Training: Parallelization allows for the simultaneous execution of multiple computational tasks, significantly speeding up AI model training. This is particularly important in healthcare applications requiring real-time responsiveness, such as medical image analysis, for quicker diagnosis and treatment planning.

- Handling Large Datasets: Healthcare generates and deals with large amounts of data, including medical records, imaging data, and genomic data. Parallelization enables the efficient processing and analysis of these large datasets, leading to more accurate and meaningful insights.

- Improving Efficiency: By leveraging parallelization techniques, AI systems can process and analyze data faster and at lower costs than traditional on-premises infrastructure [11]. This improved efficiency can help healthcare organizations optimize operations, reduce costs, and enhance patient care.

- Real-Time Decision-Making: Parallelization allows AI models to be trained and deployed in real-time, enabling healthcare professionals to make faster and more informed decisions. This is particularly valuable in critical situations where timely interventions can significantly impact patient outcomes.

DOI: https://doi.org/10.18272/aci.v17i1.3707

- Advancing Precision Medicine: Parallelization supports the development of precision medicine, which aims to provide tailored healthcare interventions based on individual characteristics and needs. By accelerating model training, parallelization enables the analysis of large-scale datasets and the identification of personalized treatment strategies.

It is important to note that parallelization is just one aspect of AI in healthcare. AI also plays a significant role in medical imaging and diagnostics, virtual patient care, medical research and drug discovery, patient engagement and compliance, rehabilitation, and other administrative applications. Integrating AI into healthcare settings can revolutionize clinical decision-making, hospital operations, medical diagnostics, patient care, and more.

EXPERIMENTATION

We use a simple node of a specific HPC testbed machine and propose using specific materials and methods, as shown in the next subsections.

HPC Infrastructure Testbed

We use a node with Intel i9-9900K 5.0 GHz processor, two (2) NVIDIA RTX 3090 24 GB Cards - GDDR6X, 384 bits CUDA, 10496 cores. The processor, the Intel Core i9-9900K, is a 9th-generation Intel Core i9 series processor with the key specifications:

- Cache: It has a 16 MB cache.
- Clock speed: It can reach a clock speed of up to 5.00 GHz.
- Cores: It has 8 cores and 16 threads.
- Manufacturing technology: It is manufactured using the 14 nm process.
  Compatibility: It is compatible with the LGA 1151 socket.

It is important to observe that cache coherence is an important feature in this configuration because the node is a multiprocessor system where multiple processors or cores share a common memory. In the AI applications of this work, cache coherence plays a crucial function in ensuring the consistency of shared data across different caches, which is essential for accurate and reliable computation.

The GeForce RTX™ 3090 is a big ferocious GPU (BFGPUs) with TITAN class performance. Powered by Ampere—NVIDIA's 2nd-generation RTX architecture—they double down on ray tracing and AI performance with enhanced Ray Tracing Cores, Tensor Cores, and new streaming multiprocessors. Plus, they feature a staggering 24 GB of G6X memory, all to deliver the ultimate experience for gamers and creators. These are some of the specifications of the NVIDIA RTX 3090:

- Memory: They have 24 GB of GDDR6X memory.
- Memory bandwidth: They have a 384-bit bus and a memory bandwidth of 936.2 GB/s.
- CUDA cores: They have 10496 CUDA cores.
- Tensor Cores: They have 328 3rd-generation Tensor Cores.
  RT Cores: They have 82 2nd-generation RT Cores.

DOI: https://doi.org/10.18272/aci.v17i1.3707

Tensor and RT Cores are specialized hardware components found in NVIDIA GPUs, specifically in their Turing architecture and later generations. These cores serve different purposes and are designed to accelerate specific tasks in AI applications and graphics rendering. Observing the Tensor Cores, they are processing units that are specifically optimized for matrix multiplication and deep learning operations. They can perform mixed-precision matrix operations at a much faster speed than traditional GPU cores. Tensor Cores benefit AI applications involving neural networks and deep learning algorithms. They can significantly accelerate tasks such as training and inference in deep learning models, resulting in faster performance and improved efficiency. Frameworks like TensorFlow utilize Tensor Cores to accelerate AI computations. They are available in limited GPUs, including the GeForce RTX, Quadro RTX, and Titan families. By leveraging the power of Tensor Cores, these GPUs can deliver enhanced performance in AI, gaming, and content creation applications.

On the other hand, RT Cores are specialized cores designed for real-time ray tracing. Ray tracing is a rendering technique that simulates the behavior of light in a scene, resulting in more realistic and accurate graphics. RT Cores are responsible for accelerating the ray tracing calculations and improving the efficiency of rendering complex lighting effects. They can calculate the paths of light rays and handle tasks such as intersection testing and shadow calculations, which are essential for realistic rendering. Like the Tensor Cores, the RT Cores are a key feature of NVIDIA's Turing architecture and subsequent generations. They enable real-time ray tracing capabilities in GPUs, allowing for more immersive and visually stunning graphics in games and other applications. In this work, we do not use the RT Cores, but the Tensor Cores.

Because Tensor Cores are specialized hardware components optimized for matrix multiplication and deep learning operations, while RT Cores are designed for real-time ray tracing, both Tensor Cores and RT Cores contribute to NVIDIA GPUs' enhanced performance and capabilities in AI applications and graphics rendering. However, for this study, the performance guarantee by the Tensor Cores is more relevant.

The NVIDIA RTX 3090s are high-performance graphics cards primarily designed for creation and intensive gaming tasks. They are also used in specialized clusters for AI, visualization, and scientific computing. This node was selected because it could correspond to a professional workstation configuration. Other nodes in a cluster can be used to increase the amount of data or intensive processes. Still, for this work, only one was selected since we were interested in the parallelization factor in the processing cores for both the CPU and the GPU/TPU. However, several aspects related to internode interconnectivity must be considered.

We use a hybrid platform based on CPU/GPU/TPUs, and it is important to discuss the CPU at this point. The Intel Core i9-9900K processor has a base clock speed of 1.2 GHz and is an octa-core processor with 16 threads, belonging to Intel's 9th generation of CPUs. The i9-9900K is known for its high performance and is often used in high-performance computing, low-cost nodes, gaming, and high-end desktop systems.

The processor features Intel Turbo Boost Technology 2.0, which allows it to dynamically increase its clock speed when needed, boosting performance for demanding tasks. It

DOI: https://doi.org/10.18272/aci.v17i1.3707

also supports Intel Hyper-Threading Technology, which enables each physical core to handle two threads simultaneously, improving multitasking capabilities. For this study, this is important due to its support for intensive calculations.

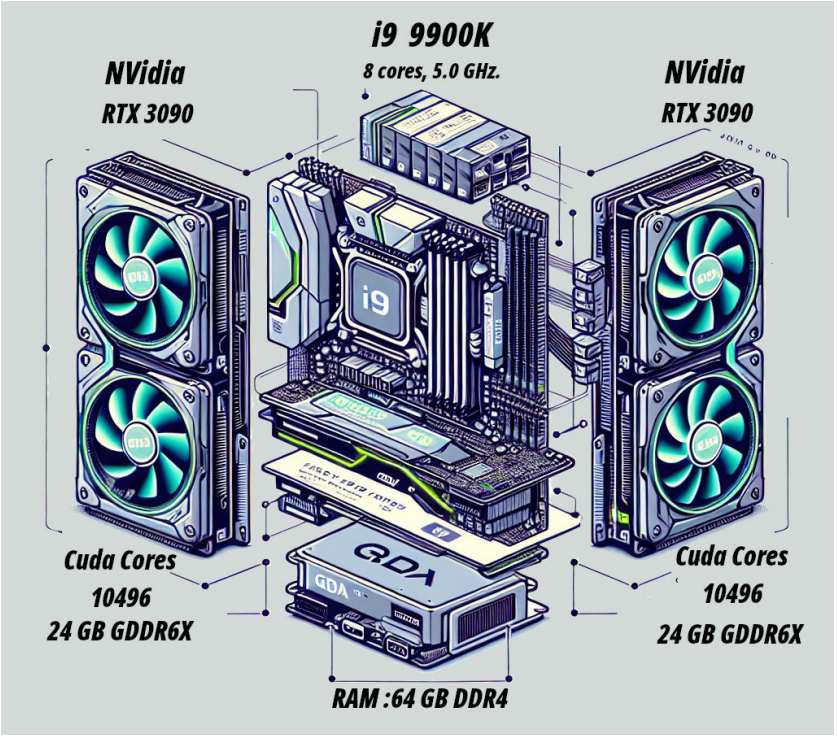

FIGURE 1. Computer Node Testbed Architecture: A hybrid computer using Intel CPUs and NVIDIA GPU/TPUs.

MATERIALS AND METHODS

The CheXpert dataset contains 224,316 chest X-rays from 65,240 patients who underwent radiographic examination at the Stanford University Medical Center between October 2002 and July 2017. Both inpatient and outpatient X-rays are included, labeled according to the presence of 14 observations as positive, negative, or uncertain. The prediction task classifies five lung disease pathologies: atelectasis, cardiomegaly, consolidation, edema, and pleural effusion.

DOI: https://doi.org/10.18272/aci.v17i1.3707

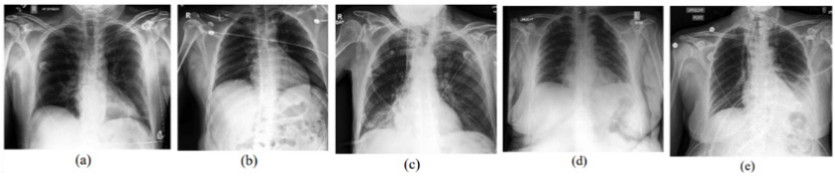

FIGURE 2. X-rays for each of the selected pathologies. (a) Atelectasis, (b) Cardiomegaly, (c) Consolidation, (d) Edema, (e) Pleural Effusion.

The structure of the prompt supplied to GPT-4 is the following:

Operations that allow use for each layer, the architecture of the 12 layers determining which are already defined and which are not, as shown in Table 1, and requesting the best architecture as a list of IDs based on the 3 operations, which allows obtaining better model performance. The results obtained with already trained models are provided. The next table shows the macrostructure of the search space.

TABLE 1. Macro structure of search space.

n	Input	Block	Channel	Stride
1	32x32x3	3x3	32	1
2	32x32x32	Choice block	64	2
3	32x32x64	Choice block	128	2
3	8x8x128	Choice block	256	2
1	4x4x256	1x1 conv	1280	1
1	4x4x1280	Global avgpool	-	-
1	1280	FC	5	-

The performance of the optimal architecture identified by the GenNASXRays generative model was evaluated in a benchmark test widely used in the NAS field. The search space is made up of eight search layers, where each layer contains three candidate blocks. These blocks are marked as zero (0) for Identity connection (encoded as '0'), one (1) for MobileNetV2 block with kernel size 3 and expansion ratio 3 (MB3_K3), and two (2) for MobileNetV2 block with kernel size 5 and expansion ratio 6 (MB6_K5). This creates a search space of 38 = 6,561.

With this experimentation over the selected testbed computing platform, interesting results can be shown and a discussion started, as shown in the next section.

RESULTS AND DISCUSSION

The training dataset used is 187,641 images for training and 7,505 images for testing. Each architecture is trained for 50 epochs using a standard SGD optimizer with a learning rate of 0.0004, a decay factor of 0.97, a batch size of 24, and a momentum of 0.9. These models

DOI: https://doi.org/10.18272/aci.v17i1.3707

were trained in their sequential version on a PC with processor i9-9900K, memory RAM of 64 Gigabytes, and GPU RTX 3090 24 GB, with 50 epochs because this was sufficient enough in terms of precision. Additional epochs did not improve the final precision and instead increased the computational cost. In the sequential execution of GenNAS with 10 iterations, the optimal neural network architecture was determined in the fifth iteration. This process took a total machine time of 190.2 hours, resulting in an AUC-ROC of 0.869, with the identified neural network structure being [2,2,1,2,1,2,1,1], shown in Table 2.

For execution in serial mode, the LLM is asked to recommend a neural network structure based on the macrostructure shown in Table 1, considering the three candidate blocks for each layer. Once a structure is recommended, it is trained and validated to determine the accuracy achievable with the generated model, as shown in Table 2.

Once this new model is trained, the LLM is asked again to recommend a new structure that might achieve better accuracy, given the results obtained with the previous architectures. The process is repeated until the accuracies no longer improve and plateau. At this point, the best model is determined as the one with the highest accuracy based on the AUC-ROC.

TABLE 2. Experiment results serial GenNAS.

Iteration	Architecture	AUC-ROC	Parameters	Memory [MBytes]	Time [Hours]
1	1,1,2,1,2,1,2,2	0.86	540741	2.16	15.139
2	2,1,2,1,2,1,2,1	0.82	750981	3	36.592
3	1,2,1,1,2,1,2,2	0.86	2274629	9.1	94.161
4	2,1,1,2,1,2,1,1	0.86	2234021	8.94	140.744
5	2,2,1,2,1,2,1,1	0.87	2702405	10.81	190.231
6	1,1,2,1,2,1,1,2	0.85	1806245	7.23	234.292
7	2,2,1,2,1,1,2,2	0.86	2253701	9.02	285.153

For parallel training execution, for each iteration, the LLM recommends two model structures which are then trained and evaluated on each GPU, determining the AUC-ROC for each trained model. The LLM is then asked again to recommend two new model structures based on the accuracies obtained previously, and this process continues until no further improvements in accuracy are made.

In contrast, during parallel execution of GenNAS with five iterations and the same computational specifications, including adding a second GPU (RTX 3090), two models were concurrently trained per iteration. This parallel approach yielded excellent results, achieving an AUC-ROC of 0.87 in the third iteration and completing in an execution time of 127.09 hours. Notably, this parallel strategy allowed for obtaining the optimal neural network structure [1,2,1,2,2,1,2,2] in a significantly shorter time than the sequential approach, as shown in Table 3. This demonstrates a substantial improvement in efficiency when employing parallelization, resulting in faster convergence to an accurate neural network architecture.

DOI: https://doi.org/10.18272/aci.v17i1.3707

TABLE 3. Experiment results parallel GenNAS.

Iteration	Architecture	AUC-ROC	Parameters	Memory [MBytes]	Time [Hours]
1	1,0,1,0,0,1,0,0	0.82	540,741	2.16	19.59
	1,0,2,0,0,2,0,0	0.82	750,981	3	
2	1,2,1,2,2,1,2,1	0.86	2,274,629	9.1	77.36
	2,1,2,1,1,2,1,2	0.86	2,234,021	8.94	
3	1,2,1,2,2,1,2,2	0.87	2,702,405	10.81	127.09
	2,1,2,1,1,2,1,1	0.85	1,806,245	7.23	
4	1,2,2,1,1,2,2,1	0.85	2,253,701	9.02	177.95
	2,1,1,2,2,1,1,2	0.86	2,254,949	9.03	
5	1,2,2,1,1,2,2,2	0.86	2,681,477	10.73	225.51
	2,1,1,2,2,1,1,1	0.86	1,827,173	7.31	

The classification of chest X-ray images has been approached using different algorithms and machine-learning techniques. One common approach is the use of convolutional neural networks (CNNs). CNNs are deep neural networks that are well-suited for image classification tasks. They can automatically learn features from the input images and make predictions based on them.

Generative parallelization of the Neural Architecture Search (NAS) is a technique that aims to improve the efficiency of the classification process. NAS involves automatically searching for the optimal neural network architecture for a given task. By parallelizing the search process, we can explore a larger space of possible architectures and find more efficient models. However, it is necessary to observe computer features to support parallelization in an energy-efficient manner.

Many research studies have focused on developing techniques to improve the efficiency of chest X-ray image classification. These include the generative parallelization of Neural Architecture Search and the augmentation of training data using generative adversarial networks. These techniques aim to enhance the performance of CNN classifiers and enable more accurate and efficient diagnoses of chest X-ray images.

Further Work

As known, the convergence of HPC infrastructures and AI can potentially revolutionize healthcare. By leveraging HPC's speed and power, combined with AI's capabilities, significant advancements can be made in various aspects of healthcare.

Although we have been explicit in the application in the previous section, we are currently experimenting with four important aspects related to the architecture: energy

DOI: https://doi.org/10.18272/aci.v17i1.3707

efficiency, deployment, scalability (without sacrificing energy efficiency and accuracy), and hybrid processing. The latter involves the analysis of massively and intensive parallel workloads and the simultaneous support of different processing units.

AUTHOR CONTRIBUTIONS

Félix Armando Mejía Cajicá conducted the research on existing literature and also implemented the source code necessary for running the performance tests and generating deep learning model structures used in this study. John Anderson García Henao contributed to test design, deep learning model structure generation strategies, and the development of parallelization algorithms. Carlos Jaime Barrios Hernández and Michell Riveill supervised the research, participated in reviewing and refining this article, and helped define parallelization strategies.

CONFLICT OF INTERESTS

The authors declare no conflict of interest.

DOI: https://doi.org/10.18272/aci.v17i1.3707

REFERENCES

[1] Zheng, M., Su, X., You, S., Wang, F., Qian, C., Xu, C., & Albanie, S. (2023). Can GPT-4 perform neural architecture search? *arXiv preprint arXiv:2304.10970*. https://doi.org/10.48550/arXiv.2304.10970

[2] Zhang, S., Gong, C., Wu, L., Liu, X., & Zhou, M. (2023). AutoML-GPT: Automatic Machine Learning with GPT. *arXiv preprint arXiv:2305.02499*. https://doi.org/10.48550/arXiv.2305.02499

[3] Tornede, A., Deng, D., Eimer, T., Giovanelli, J., Mohan, A., Ruhkopf, T., Segel, S., Theodorakopoulos, D., Tornede, T., Wachsmuth, H., & Lindauer, M. (2023). AutoML in the age of Large Language Models: Current challenges, future opportunities and risks. *arXiv preprint arXiv:2306.08107*. https://doi.org/10.48550/arXiv.2306.08107

[4] Matsuoka, S. (2018, June). Cambrian explosion of computing and big data in the post-Moore era. In *Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing* (pp. 105-105). https://doi.org/10.1145/3208040.3225055

[5] Onwusinkwue, S., Osasona, F., Ahmad, I. A. I., Anyanwu, A. C., Dawodu, S. O., Obi, O. C., & Hamdan, A. (2024). Artificial intelligence (AI) in renewable energy: A review of predictive maintenance and energy optimization. *World Journal of Advanced Research and Reviews, 21*(1), 2487-2799. https://doi.org/10.30574/wjarr.2024.21.1.0347

[6] Hidalgo, I., Fenández-de_Vega, F., Ceberio, J., Garnica, O., Velasco, J. M., Cortés, J. C., Villanueva, R., & Díaz, J. (2023). Sustainable artificial intelligence systems: An energy efficiency approach. *TechRxiv.* https://doi.org/10.36227/techrxiv.24610899

[7] Rao, B. C. (2024). Frugal computing for artificial intelligence and other applications. In *Frugal engineering. Design science and innovation*. Springer. https://link.springer.com/chapter/10.1007/978-981-99-9700-8_11

[8] Al Kuwaiti, A., Nazer, K., Al-Reedy, A., Al-Shehri, S., Al-Muhanna, A., Subbarayalu, A. V., Al Muhanna, D., & Al-Muhanna, F. A. (2023). A review of the role of artificial intelligence in healthcare. *Journal of Personalized Medicine, 13*(6), 951. https://doi.org/10.3390/jpm13060951

[9] Bajwa, J., Munir, U., Nori, A., & Williams, B. (2021). Artificial intelligence in healthcare: transforming the practice of medicine. *Future Healthcare Journal, 8*(2), e188-e194. https://doi.org/10.7861/fhj.2021-0095

[10] Davenport, T., & Kalakota, R. (2019). The potential for artificial intelligence in healthcare. *Future Healthcare Journal, 6*(2), 94-98. https://doi.org/10.7861/futurehosp.6-2-94

[11] Calciu, I., Talha Imran, M., Puddu I., Kashyap, S., Al Maruf, H., Mutlu, O., & Kolli, A. (2023). Using local cache coherence for disaggregated memory systems. SIGOPS Oper. Syst. Rev. 57, 1 (June 2023), 21–28. *ACM SIGOPS Operating Systems Review, 57*(1), 21-28. https://doi.org/10.1145/3606557.3606561

 DOI: https://doi.org/10.18272/aci.v17i1.3707