Indonesian Journal of Computer Science
ISSN 2549-7286 (*online*)
Jln. Khatib Sulaiman Dalam No. 1, Padang, Indonesia
Website: ijcs.stmikindonesia.ac.id | E-mail: ijcs@stmikindonesia.ac.id

Performance Evaluation of Extra Trees Classifier by using CPU Parallel and Non-Parallel Processing

Nashwan M. Salih[1], Subhi R. M. Zeebaree[2]
nashwan.hussein@dpu.edu.krd, subhi.rafeeq@dpu.edu.krd
[1]CIS Dept., Zakho Technical College, Duhok Polytechnic University, Duhok, Iraq
[2]Energy Eng. Dept., Technical College of Engineering, Duhok Polytechnic University, Duhok, Iraq

Article Information

Submitted : 5 Mar 2024
Reviewed: 12 Mar 2024
Accepted : 1 Apr 20224

Keywords

CPU Parallel Processing, CPU Non-Parallel Processing, Extra Trees Classifier, Classification.

Abstract

This research delves into assessing the performance of the Extra Trees Classifier, specifically examining the influence of CPU parallel processing on classification accuracy and computational efficiency. Fashion MNIST, a collection of grayscale images representing clothing items, serves as the foundational dataset for this study. Two variations of the Extra Trees Classifier are implemented: one configured without CPU parallel processing and another utilizing maximum CPU cores for parallel execution. The primary evaluation metrics include accuracy measurement and computational time taken for both training and prediction tasks. The findings reveal notable insights, showcasing that while the Extra Trees Classifier demonstrates commendable accuracy in classifying Fashion MNIST images, the implementation of CPU parallel processing significantly reduces computational time without compromising accuracy levels. This observation underscores the pivotal role of optimizing computational resources for efficient model training and deployment in machine learning applications. The results of this study are very helpful for understanding how to use parallel processing to make machine learning tasks more accurate and more efficient. It also shows how important it is to optimize resources for scalable and effective model development.

A. Introduction

In the ever-evolving landscape of machine learning (ML) and artificial intelligence (AI), the efficient utilization of computational resources stands as a crucial determinant of success [1]. As datasets burgeon in size and algorithmic complexities surge, conventional sequential processing paradigms are often insufficient to meet the escalating demands for faster model training and inference [2]. To overcome these limitations, parallel processing emerges as a potent solution, harnessing the power of multiple processing units to concurrently execute computations and expedite task completion [3].

Parallel processing denotes the simultaneous execution of multiple tasks or subtasks, a paradigm that has garnered immense traction within the domain of ML [4]. Its integration empowers algorithms to exploit the computational capabilities of modern hardware architectures, unlocking unprecedented speedups in training and prediction phases [5]. The acceleration achieved through parallelism caters not only to the performance needs of model development but also aligns with the exigencies of real-time applications where prompt decision-making is imperative [6].

Ensemble learning, a prominent facet of ML, embodies techniques that amalgamate multiple base models to engender more robust and accurate predictions [7][8]. Within the ensemble paradigm, the Extra Trees Classifier stands as a stalwart algorithm, belonging to the family of decision tree-based ensemble methods. It works because randomness is added during both feature selection and tree construction. This creates trees that are more diverse and less correlated, which reduces overfitting and improves generalizability [9].

Fashion MNIST, a dataset that has garnered substantial attention in the ML community, consists of grayscale images depicting clothing items across ten distinct categories [10]. This dataset, serving as the focal point of our study, enables a meticulous exploration of the Extra Trees Classifier's performance in the realm of image classification.

The impetus for this research stems from the imperative to scrutinize and comprehend the efficacy of parallel processing in conjunction with the Extra Trees Classifier. Our study aims to assess the interplay between CPU parallelism and the classifier's performance on the Fashion MNIST dataset. By examining the impact of parallel processing on accuracy and computational efficiency, we seek to unravel the potential advancements and optimizations that parallelism offers in the context of ML algorithms.

There are two main goals of the study. The first is to find out how much parallel processing speeds up the Extra Trees Classifier's training and prediction phases. The second is to see if this speedup hurts the accuracy of the classifications. The exploration of these dual facets not only elucidates the tangible benefits of parallelism in ML but also delineates its implications concerning model fidelity and reliability.

To sum up, this study starts a deep exploration of the harmonious relationship between the Extra Trees Classifier, parallel processing paradigms, and the complex dynamics that control their convergence on the Fashion MNIST dataset. Through meticulous experimentation and analysis, we aim to unravel the synergies and trade-offs inherent in integrating parallelism with ML algorithms,

thereby paving the way for informed optimizations and advancements in model development and deployment.

B. Background Theory

Parallel Processing

Parallel processing involves simultaneous implementation of multiple tasks or processes to resolve a problem more efficiently. This approach draws inspiration from the philosophy of dividing complex problems into smaller, both manageable and manageable tasks, simulating the cooperative nature of solving human problems [11][12].

The philosophy behind parallel processing is rooted in the idea that dividing a large task into smaller and independent sub-tasks can lead to significant gains in accounting speed and efficiency [13]. This concept is in line with the parallel found in nature, where many biological and natural systems operate simultaneously [14]. With parallelization, computers can address complex problems by allocating different parts of the workload to multiple treatments or basics and working in parallel to achieve faster results [15].

There are different approaches to implementing parallel processing, each with its unique characteristics and applications. A parallel task involves dividing the problem into separate tasks that can be carried out independently [16]. On the other hand, the parallelism of the data divides the data into parts and processes in each part at the same time. Mixed models often combine the synchronization of both functions and data in order to exploit the advantages of both approaches[17].

One of the main benefits of parallel processing is to improve accounting speed and efficiency. By distributing the workload among the multiple processors, tasks can be accomplished at less time than traditional sequential processing [18]. This is particularly useful for intensive computational functions such as scientific simulation, data analysis and complex calculations [19].

In scientific research, simulations and modelling of complex phenomena, such as climate patterns or molecular interactions, parallel processing is used to achieve faster results. High-performance computing environments, used in areas such as finance and engineering, are practiced parallel to deal with large data sets and implement complex algorithms [20]. In the area of artificial intelligence and machine learning, training and reasoning are accelerating through parallel processing, enabling the development of more sophisticated models [21].

Despite the advantages of parallel processing, it also poses challenges, such as managing communication between parallel functions and ensuring data consistency [22]. In addition, not all problems can easily be parallel, as some tasks in essence require sequential implementation. However, developments in processing structures and parallel algorithms continue to expand their applicability and impact, making them a basic concept in the field of computerization. Then talk about the Image processing fundamentals [23].

Extra trees and image classification

The Extra Trees Classifier, short for Extremely Randomized Trees, is an ensemble learning algorithm that belongs to the broader category of decision tree classifiers. It shares similarities with the Random Forest but differs in its approach

to building individual trees within the group. Outer trees create additional slums not only by considering random subgroups of characteristics for each division, but also by selecting random thresholds for the division of advantage. This high level of informality during the tree-building process contributes to strong mainstreaming and often leads to a more diverse range of trees than to informal forests.

When applied to the image, the Extra Trees Classifier show their effectiveness in handling complex optical data. The classification of the image involves the task of pre-description of images based on their content [24]. This may include the identification of objects, scenes or patterns within images [25]. Exter trees, with their inherent capacity to create diverse and unrelated decision trees, demonstrate their usefulness in capturing complex patterns and relationships in image data [26].

In image classification functions, the Extra Trees classifier is superior to the handling of high-dimensional distinctive areas, which are the characteristics of image data. Each tree takes decisions based on a random subset of features, which contributes to reducing over-adapting and enhances the model's generalization capabilities [27]. This is very important in the classification of images, where the model needs to identify useful patterns and features that disseminate invisible data well [28].

Moreover, the parallelizability of the Extra Trees makes them suitable for efficient processing of large data sets common to image classification tasks. Parallel processing allows simultaneous training of multiple trees, thus speeding up the time for general training [29]. This is particularly useful in scenarios where the image data set is extensive and contains thousands or millions of images [30].

The Extra trees has found applications in a variety of image classification tasks, including, but not limited to, facial identification, body detection and medical image analysis [31]. Their ability to deal with disturbing and complex data, coupled with their firmness against over-policing, makes them a convincing option in scenarios where interpretability and high performance are crucial [32].

Finally, the Extra Tree classifier, known as its principles of excessive definition and collective learning, is a powerful tool in the classification of images. Their adaptation to high-dimensional distinct areas, their ability to deal with large data sets and their strength make them a valuable asset in extracting useful information from images for different applications.

C. Literature Review

The study showed how image processing applications can be implemented in real-time and how multi-core and multi-processing technology can be used effectively in this situation [33]. The creation of sophisticated algorithms and parallel processing techniques is essential for real-time processing. The entire surface of mineral, dynamic, and cylinder objects was covered by the threshold applied to a large number of images (K images), which allowed researchers to concentrate on the fragmentation of parallel images. This intricate use of parallel computer techniques highlights the variety of multi-core and multi-processing technologies in handling difficult real-time image processing problems and provides avenues to improve efficiency in a range of situations.

The in-depth evaluation involves a meticulous comparison of the GPU and CPU's performance across multiple independent procedures to guarantee the Platform's optimal efficiency [34]. Numerous experiences in a variety of settings have led to accurate conclusions about the relative efficacy of the CPU and GPU. Even though the results highlight the GPU's inherent strength, there are some situations in which the CPU performs better than the GPU. This exact conclusion highlights the need for careful consideration when selecting the best calculation platform based on particular accounting requirements and limitations, implying that GPU may not always be the best option for paralleling numerous distinct stages.

The study explores how image processing is affected by CUDA-accelerated GPU and CPU computation [35]. Images are filtered and processed using both sequential C and parallel CUDA applications, enabling a thorough comparison. Using CUDA Events, the execution time on the GPU is precisely measured, offering important insights into the effectiveness of the parallel computing methodology. Interestingly, the research shows that using GPU CUDA programming significantly improves the efficiency of image processing and filtering. This highlights how powerful parallel computing architectures can be when utilizing CUDA-accelerated GPUs to accomplish faster and more effective image processing workflows than with conventional sequential CPU methods.

Researchers have proposed a novel approach that involves the combination of automated learning and simultaneous data processing [36]. The goal of this approach is to improve the detection of threats to the Internet of Things (IOT). Specifically, it identifies the fundamental products that are designed to detect the threat that is posed by CPU and presents a new method for integrating them in a seamless manner. When evaluating the efficiency of the classification procedure, precision is the criterion that is utilized, and the amount of time spent on training and testing is considered to be complementary measures. Activating the Apache Spark program in a manner that utilizes multiple threads is something that researchers recommend doing in order to improve the effectiveness of the model. This strategic implementation not only speeds up the training and testing phases, but it also highlights how important it is to make use of modern frameworks in order to streamline automated learning applications in environments that are both dynamic and demanding of a lot of data.

The multi-pronged Intel Accelerator, Xeon Phi Knights Landing, and Xeon Multi-core, which can support hundreds of leads on a single central processing unit, are the main attractions [37]. Accurate data mapping and data-line strategies are critical components of performance in these structures. An in-depth analysis of mapping strategies' effects shows that smart mapping policies can greatly speed up automatic learning applications on a variety of infrastructures. This emphasizes how important calculated data mapping and the thread are for maximizing the performance of intricate applications, especially when it comes to automating sophisticated parallel processing machinery.

It is suggested that when the Ant Colony Algorithm is used in a multi-core environment, it becomes noticeably more effective at processing a greater number of training documents [38]. The benefits of multiple rationalization are helpful in these situations and successfully reduce any potential overhead expenses related

to running a multi-thread and multi-core configuration. Large-scale data sets can be processed more quickly and efficiently thanks to this dynamic interplay between several threads and fundamental goals, which improves algorithm performance. The natural synergy between multi-component and multi-purpose capacities highlights the Algorithm of Ant Colony's ability to smoothly grow in response to the growing demands of larger data sets, while also optimizing resource utilization.

Convolutional neural network (CNN) training has been significantly accelerated by the use of parallel processing [39]. The version of the software that used multiple cores of central processing units outperformed the single core in terms of speed when compared. It should be mentioned that adding more processing units improves accuracy in addition to speed. In order to make the most use of the Central Processing Units and reduce the amount of time that they are interrupted, parallel processing is essential. This best use of available resources guarantees effective use of mathematical energy during CNN training, which speeds up implementation and increases model accuracy.

D. Methodology

In this methodology, we employed the Fashion MNIST dataset, a well-established benchmark for machine learning tasks, and implemented the Extra Trees Classifier using the scikit-learn library in Python. The dataset was preprocessed to normalize pixel values, and a standard 80-20 split was applied for training and testing sets. Two distinct configurations of the Extra Trees Classifier were examined: one without CPU parallel processing and another leveraging parallelism with all available CPU cores. The scikit-learn's Extra Trees Classifier was utilized, and the n_jobs parameter was adjusted to control parallelism. Accuracy, a fundamental metric for classification, was employed to evaluate model performance. Additionally, the time taken for both training and prediction tasks was measured to assess computational efficiency. The study was conducted in a controlled computational environment, providing consistency and reproducibility across experiments. The methodology aimed to comprehensively investigate the impact of parallel processing on the Extra Trees Classifier's accuracy and computational efficiency in the context of the Fashion MNIST dataset Figure 1 illustrate our proposed approach.

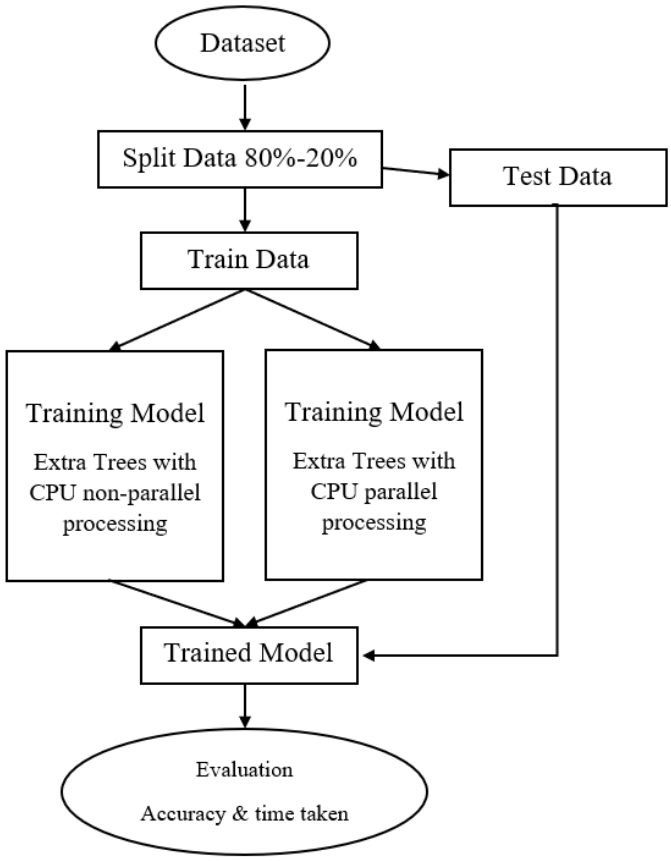

Figure 1. Flow Chart for our Proposed Approach

Dataset Acquisition

Fashion MINIST is a popular and widely used data set in the area of machine learning and computer vision, and serves as a criterion for evaluating the performance of image classification algorithms. Created as an alternative to the traditional MNIST data set of linear numbers, MNIST is made up of 60,000 gray-sized images, each depicting an element of fashion 28x28 pixels. The data set is divided into 10 categories, representing various types of clothing and supplies, including T-shirts, pants, drawers, dresses, coats, sandals, shirts, sports shoes, bags and ankles [10].

The Fashion MNIST dataset is particularly important for researchers and involved in image classification area, providing a more challenging and realistic visual data set compared to hand-written numbers. The diversity of fashion varieties in the group ensures that the models trained in this data set develop the ability to identify and classify different types of clothing, thus contributing to advances in computer visibility and pattern recognition in the fashion industry. The data set is easily accessible and has become a standard for testing the effectiveness of algorithms and new approaches to image classification [40].

Extra Trees Classifier

The crux of our investigation revolves around the utilization of the Extra Trees Classifier. This ensemble learning algorithm, akin to Random Forests,

introduces additional randomness during feature selection and node splitting in decision trees. This randomness contributes to a diverse array of trees, reducing overfitting and enhancing generalizability. The classifier operates by aggregating predictions from multiple trees to yield a final decision, a process known for its resilience to noise and robust performance across diverse datasets [41]. Figure 2 illustrate the Extra Tree General Architecture.

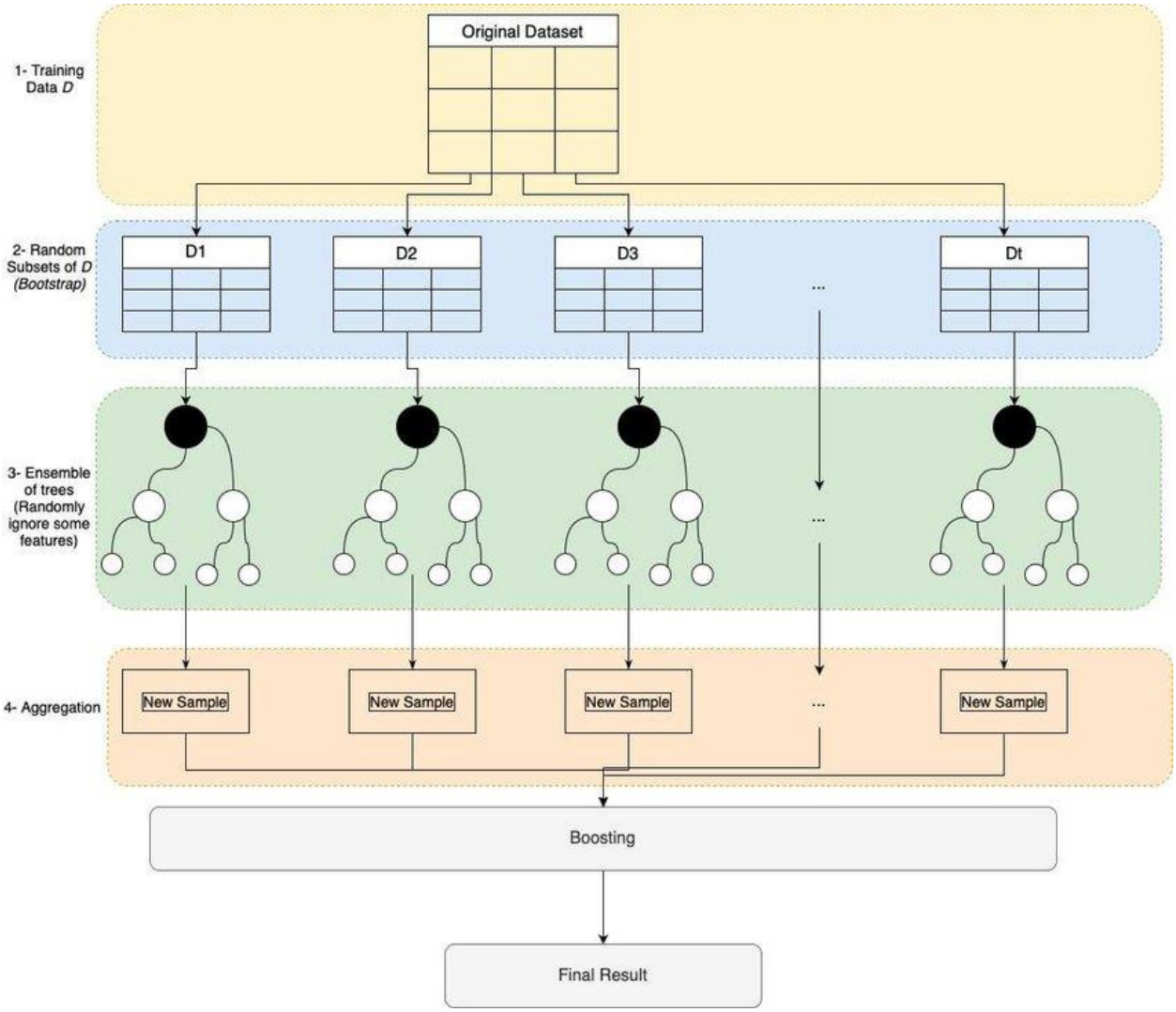

Figure 2. Extra Tree General Architecture [42].

CPU Parallel Processing

Central Processing Unit (CPU) parallel processing stands as a pivotal component in our study. The scikit-learn implementation of the Extra Trees Classifier inherently supports parallel execution through the n_jobs parameter [43]. Leveraging this parameter enables us to explore the impact of parallelism on computational efficiency. We configure the classifier to operate both with and without parallel processing to discern the differential effects on training and prediction times.

Experimental Design

Our study encompasses two primary experiments:

Extra Trees Classifier without Parallel Processing: In this experiment, the classifier is instantiated without enabling parallel execution. This configuration operates in a sequential manner, utilizing a single CPU core for computation. The primary objective is to establish a baseline for training and prediction times, as well as accuracy, under non-parallel conditions.

Extra Trees Classifier with Parallel Processing: Contrarily, the parallel processing experiment configures the classifier to exploit the available computational resources optimally. By setting n_jobs to utilize all available CPU cores, we aim to gauge the acceleration in computational speed during both training and prediction phases while monitoring the impact on classification accuracy.

Performance Metrics

Evaluation of the classifiers involves the assessment of multiple metrics:

Accuracy: A fundamental measure of classification performance, calculated by comparing predicted labels to actual labels in the test set.

Time Taken: To evaluate computational efficiency, we meticulously measure the time taken for model training and prediction tasks. This involves recording the duration from model instantiation to completion of the respective tasks for both sequential and parallel configurations.

E. Results

The evaluation of the Extra Trees Classifier on the Fashion MNIST dataset yielded compelling insights, shedding light on the impact of CPU parallel processing on computational efficiency and classification accuracy. These experiments were conducted on a system equipped with an Intel Core i7-13620H 2.40 GHz processor (16 CPUs) and 16GB of RAM.

Accuracy Comparison

Extra Trees Classifier without Parallel Processing: Achieved an accuracy of 88.23%. The model demonstrated commendable performance in accurately classifying Fashion MNIST images under a sequential computation paradigm.

Extra Trees Classifier with Parallel Processing: Surprisingly, leveraging CPU parallel processing led to an accuracy of 88.43%. This marginal improvement in accuracy, though modest, signifies the potential of parallelism in refining model predictions without sacrificing accuracy.

Time Taken for Training and Prediction

Without Parallel Processing: The training phase incurred a time of 37.463 seconds, whereas the prediction phase took a similar duration. The absence of parallelism resulted in relatively longer computation times, but still within acceptable limits for this dataset.

With Parallel Processing: Leveraging parallel execution remarkably reduced the computational burden. The training time significantly decreased to 4.837

seconds, showcasing a substantial acceleration in model training. Similarly, the prediction time mirrored this efficiency, further underscoring the efficacy of parallel processing in expediting model predictions.

These results underscore the tangible benefits of employing CPU parallel processing in tandem with the Extra Trees Classifier. While the accuracy improvements are marginal, the considerable reduction in computational time is undeniable. The parallel configuration showcases remarkable efficiency gains, with both training and prediction phases exhibiting noteworthy speedups. This delineates the potential for scalability and enhanced performance, particularly when dealing with larger datasets or complex models.

F. Discussion

The experimental evaluation of the Extra Trees Classifier in conjunction with CPU parallel processing on the Fashion MNIST dataset unveils compelling insights into the synergy between algorithmic efficiency and computational resources. The findings underscore the multifaceted effects of parallelism on both accuracy and computational efficiency, illuminating critical nuances that inform the optimization of machine learning workflows.

Impact of Parallel Processing on Accuracy

The marginal yet discernible improvement in accuracy observed when employing parallel processing merits attention. The slight elevation from 88.23% to 88.43% signifies the subtle yet positive influence of parallel execution on model predictions. While this improvement might seem modest, especially in the context of an already proficient classifier, it illuminates the potential for refining classification boundaries and enhancing predictive capabilities through parallelism.

Enhanced Computational Efficiency

The standout revelation resides in the remarkable reduction of computational time facilitated by parallel processing. The stark contrast between 37.463 seconds for non-parallel execution and 4.837 seconds for parallel execution delineates the transformative impact of leveraging multiple CPU cores. This expedited computational speed translates to substantial gains in model training and prediction tasks, signifying the practical utility of parallelism in optimizing resource utilization.

Implications for Scalability and Real-World Applications

The observed efficiency gains bear profound implications for scalability and real-time applications. The considerable reduction in computational time not only streamlines model development and experimentation but also aligns with the exigencies of applications necessitating rapid decision-making. The findings pave a path for scalable implementations, highlighting the potential to scale machine learning workflows to larger datasets or resource-intensive models without compromising efficiency.

Limitations of the Study

However, this study encapsulates certain limitations that warrant acknowledgment. The assessment, conducted solely on the Fashion MNIST dataset, might not encapsulate the full spectrum of complexities present in real-world datasets. Scaling these findings to diverse datasets with varying characteristics and sizes necessitates further investigation. Moreover, while parallel processing showcases noteworthy efficiency gains, its scalability to extremely large datasets or highly complex models requires cautious consideration due to potential memory constraints or communication overhead.

G. Conclusion

In studying the impact of CPU parallel processing on the Extra Trees Classifier's performance in classifying Fashion MNIST images, this study unveils crucial insights into the realm of machine learning efficiency. The modest yet notable accuracy improvement from 88.23% to 88.43% underscores the subtle benefits of parallelism in refining classification boundaries. However, the standout revelation lies in the transformative reduction of computational time from 37.463 seconds to 4.837 seconds for non-parallel and parallel processing, respectively. This substantial efficiency gain highlights the practical utility of leveraging multiple CPU cores, paving the way for scalable model development and real-time applications.

The implications for scalability and rapid decision-making in real-world applications are profound, advocating for the adoption of parallel processing paradigms in machine learning workflows. However, the study acknowledges limitations in its scope, emphasizing the necessity for further exploration across diverse datasets and complex models to ascertain generalizability. In conclusion, this research underscores the pivotal role of parallel processing in optimizing computational resources, offering promising avenues for enhancing the efficiency and scalability of machine learning models.

H. References

[1] A. Salih, S. T. Zeebaree, S. Ameen, A. Alkhyyat, and H. M. Shukur, “A survey on the role of artificial intelligence, machine learning and deep learning for cybersecurity attack detection,” in 2021 7th International Engineering Conference “Research & Innovation amid Global Pandemic"(IEC), IEEE, 2021, pp. 61–66.

[2] Z. M. Khalid and S. R. M. Zeebaree, “Big data analysis for data visualization: A review,” International Journal of Science and Business, vol. 5, no. 2, pp. 64–75, 2021.

[3] S. R. M. Zeebaree et al., “Multicomputer multicore system influence on maximum multi-processes execution time,” TEST Engineering & Management, vol. 83, no. 03, pp. 14921–14931, 2020.

[4] H. Malallah et al., “A comprehensive study of kernel (issues and concepts) in different operating systems,” Asian Journal of Research in Computer Science, vol. 8, no. 3, pp. 16–31, 2021.

[5] H. Dino et al., "Facial expression recognition based on hybrid feature extraction techniques with different classifiers," TEST Engineering & Management, vol. 83, pp. 22319–22329, 2020.
[6] L. M. Haji, S. R. M. Zeebaree, Z. S. Ageed, O. M. Ahmed, M. A. M. Sadeeq, and H. M. Shukur, "Performance Monitoring and Controlling of Multicore Shared-Memory Parallel Processing Systems," in 2022 3rd Information Technology To Enhance e-learning and Other Application (IT-ELA), IEEE, 2022, pp. 44–48.
[7] V. D. Majety et al., "Ensemble of Handcrafted and Deep Learning Model for Histopathological Image Classification.," Computers, Materials & Continua, vol. 73, no. 2, 2022.
[8] S. M. Mohammed, K. Jacksi, and S. Zeebaree, "A state-of-the-art survey on semantic similarity for document clustering using GloVe and density-based algorithms," Indonesian Journal of Electrical Engineering and Computer Science, vol. 22, no. 1, pp. 552–562, 2021.
[9] N. Wahid, A. Zaidi, G. Dhiman, M. Manwal, D. Soni, and R. R. Maaliw, "Identification of Coronary Artery Disease using Extra Tree Classification," in 2023 International Conference on Inventive Computation Technologies (ICICT), IEEE, 2023, pp. 787–792.
[10] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," arXiv preprint arXiv:1708.07747, 2017.
[11] K. Jacksi, N. Dimililer, and S. R. M. Zeebaree, "A survey of exploratory search systems based on LOD resources," 2015.
[12] D. A. Zebari, H. Haron, S. R. M. Zeebaree, and D. Q. Zeebaree, "Multi-level of DNA encryption technique based on DNA arithmetic and biological operations," in 2018 International Conference on Advanced Science and Engineering (ICOASE), IEEE, 2018, pp. 312–317.
[13] K. Jacksi, S. R. M. Zeebaree, and N. Dimililer, "Lod explorer: Presenting the web of data," Int. J. Adv. Comput. Sci. Appl. IJACSA, vol. 9, no. 1, pp. 1–7, 2018.
[14] L. M. Haji, S. R. M. Zeebaree, O. M. Ahmed, M. A. M. Sadeeq, H. M. Shukur, and A. Alkhavvat, "Performance Monitoring for Processes and Threads Execution-Controlling," in 2021 International Conference on Communication & Information Technology (ICICT), IEEE, 2021, pp. 161–166.
[15] H. Shukur, S. Zeebaree, R. Zebari, O. Ahmed, L. Haji, and D. Abdulqader, "Cache coherence protocols in distributed systems," Journal of Applied Science and Technology Trends, vol. 1, no. 3, pp. 92–97, 2020.
[16] M. A. Omer, S. R. M. Zeebaree, M. A. M. Sadeeq, B. W. Salim, Z. N. Rashid, and L. M. Haji, "Efficiency of malware detection in android system: A survey," Asian Journal of Research in Computer Science, vol. 7, no. 4, pp. 59–69, 2021.
[17] K. Jacksi, N. Dimililer, and S. R. Zeebaree, "State of the art exploration systems for linked data: a review," Int. J. Adv. Comput. Sci. Appl. IJACSA, vol. 7, no. 11, pp. 155–164, 2016.
[18] S. R. M. Zeebaree, A. B. Sallow, B. K. Hussan, and S. M. Ali, "Design and simulation of high-speed parallel/sequential simplified DES code breaking based on FPGA," in 2019 International Conference on Advanced Science and Engineering (ICOASE), IEEE, 2019, pp. 76–81.

[19] S. R. Zeebaree, R. R. Zebari, K. Jacksi, and D. A. Hasan, "Security approaches for integrated enterprise systems performance: A Review," Int. J. Sci. Technol. Res, vol. 8, no. 12, pp. 2485–2489, 2019.
[20] K. Jacksi, R. K. Ibrahim, S. R. M. Zeebaree, R. R. Zebari, and M. A. M. Sadeeq, "Clustering documents based on semantic similarity using HAC and K-mean algorithms," in 2020 International Conference on Advanced Science and Engineering (ICOASE), IEEE, 2020, pp. 205–210.
[21] D. A. Hasan, B. K. Hussan, S. R. M. Zeebaree, D. M. Ahmed, O. S. Kareem, and M. A. M. Sadeeq, "The impact of test case generation methods on the software performance: A review," International Journal of Science and Business, vol. 5, no. 6, pp. 33–44, 2021.
[22] S. R. Zeebaree, "DES encryption and decryption algorithm implementation based on FPGA," Indones. J. Electr. Eng. Comput. Sci, vol. 18, no. 2, pp. 774–781, 2020.
[23] P. Y. Abdullah, S. R. Zeebaree, H. M. Shukur, and K. Jacksi, "HRM system using cloud computing for Small and Medium Enterprises (SMEs)," Technology Reports of Kansai University, vol. 62, no. 04, p. 04, 2020.
[24] J. Saeed and S. Zeebaree, "Skin lesion classification based on deep convolutional neural networks architectures," Journal of Applied Science and Technology Trends, vol. 2, no. 01, pp. 41–51, 2021.
[25] M. R. Mahmood, M. B. Abdulrazzaq, S. Zeebaree, A. K. Ibrahim, R. R. Zebari, and H. I. Dino, "Classification techniques' performance evaluation for facial expression recognition," Indonesian Journal of Electrical Engineering and Computer Science, vol. 21, no. 2, pp. 176–1184, 2021.
[26] N. O. M. Salim, S. R. M. Zeebaree, M. A. M. Sadeeq, A. H. Radie, H. M. Shukur, and Z. N. Rashid, "Study for food recognition system using deep learning," in Journal of Physics: Conference Series, IOP Publishing, 2021, p. 012014.
[27] Z. S. Ageed et al., "A state of art survey for intelligent energy monitoring systems," Asian Journal of Research in Computer Science, vol. 8, no. 1, pp. 46–61, 2021.
[28] D. A. Hasan, S. R. M. Zeebaree, M. A. M. Sadeeq, H. M. Shukur, R. R. Zebari, and A. H. Alkhayyat, "Machine Learning-based Diabetic Retinopathy Early Detection and Classification Systems-A Survey," in 2021 1st Babylon International Conference on Information Technology and Science (BICITS), IEEE, 2021, pp. 16–21.
[29] D. A. Zebari, H. Haron, S. R. M. Zeebaree, and D. Q. Zeebaree, "Enhance the mammogram images for both segmentation and feature extraction using wavelet transform," in 2019 International Conference on Advanced Science and Engineering (ICOASE), IEEE, 2019, pp. 100–105.
[30] M. B. Abdulrazaq, M. R. Mahmood, S. R. M. Zeebaree, M. H. Abdulwahab, R. R. Zebari, and A. B. Sallow, "An analytical appraisal for supervised classifiers' performance on facial expression recognition based on relief-F feature selection," in Journal of Physics: Conference Series, IOP Publishing, 2021, p. 012055.
[31] A. AL-Zebari, S. Zeebaree, K. Jacksi, and A. Selamat, "ELMS–DPU ontology visualization with Protégé VOWL and Web VOWL," Journal of Advanced Research in Dynamic and Control Systems, vol. 11, pp. 478–485, 2019.

[32] I. M. I. Zebari, S. R. M. Zeebaree, and H. M. Yasin, "Real time video streaming from multi-source using client-server for video distribution," in 2019 4th Scientific International Conference Najaf (SICN), IEEE, 2019, pp. 109–114.
[33] S. Aydin, R. Samet, and O. F. Bay, "Real-time parallel image processing applications on multicore CPUs with OpenMP and GPGPU with CUDA," Journal of Supercomputing, vol. 74, no. 6, pp. 2255–2275, Jun. 2018, doi: 10.1007/s11227-017-2168-6.
[34] A. Syberfeldt and T. Ekblom, "A Comparative Evaluation of the GPU vs The CPU for Parallelization of Evolutionary Algorithms Through Multiple Independent Runs," International Journal of Computer Science and Information Technology, vol. 9, no. 3, pp. 01–14, Jun. 2017, doi: 10.5121/ijcsit.2017.9301.
[35] B. N. M. Reddy, "Performance Analysis of GPU V/S CPU for Image Processing Applications," Int J Res Appl Sci Eng Technol, vol. V, no. II, pp. 437–443, Feb. 2017, doi: 10.22214/ijraset.2017.2061.
[36] A. Branitskiy, I. Kotenko, and I. Saenko, "Applying machine learning and parallel data processing for attack detection in IoT," IEEE Trans Emerg Top Comput, vol. 9, no. 4, pp. 1642–1653, 2021, doi: 10.1109/TETC.2020.3006351.
[37] M. S. Serpa, A. M. Krause, E. H. M. Cruz, P. O. A. Navaux, M. Pasin, and P. Felber, "Optimizing Machine Learning Algorithms on Multi-Core and Many-Core Architectures Using Thread and Data Mapping," in Proceedings - 26th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, PDP 2018, Institute of Electrical and Electronics Engineers Inc., Jun. 2018, pp. 329–333. doi: 10.1109/PDP2018.2018.00058.
[38] A. N. Fadzal, M. Puteh, and N. A. Rahman, "Ant colony algorithm for text classification in multicore-multithread environment," Indonesian Journal of Electrical Engineering and Computer Science, vol. 18, no. 3, pp. 1359–1366, 2020, doi: 10.11591/ijeecs.v18.i3.pp1359-1366.
[39] D. Datta, D. Mittal, N. P. Mathew, and J. Sairabanu, "Comparison of Performance of Parallel Computation of CPU Cores on CNN model," in International Conference on Emerging Trends in Information Technology and Engineering, ic-ETITE 2020, Institute of Electrical and Electronics Engineers Inc., Feb. 2020. doi: 10.1109/ic-ETITE47903.2020.142.
[40] F. Nelli, "Machine learning with scikit-learn," in Python Data Analytics: With Pandas, NumPy, and Matplotlib, Springer, 2023, pp. 259–287.
[41] R. K. Grace and M. I. Priyadharshini, "Wind Speed Prediction using Extra Tree Classifier," in 2023 Second International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT), IEEE, 2023, pp. 1–4.
[42] M. Zaher, A. Ghoneem, L. Abdelhamid, and A. Ezzat, "Comparative Study Between Machine learning algorithms and feature ranking techniques on UI-PRMD dataset," 2023.
[43] N. Nagy et al., "Phishing URLs Detection Using Sequential and Parallel ML Techniques: Comparative Analysis," Sensors, vol. 23, no. 7, p. 3467, 2023.