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

Comparative Analysis of XGBoost Performance for Text Classification with CPU Parallel and Non-Parallel Processing

Omer M. Ahmed[1], Subhi R. M. Zeebaree[2], Shavan Askar[3]
omar.alzakholi@dpu.edu.krd, subhi.rafeeq@dpu.edu.krd, shavan.askar@epu.edu.iq
[1]CIS Dept., Zakho Technical College, Duhok Polytechnic University, Duhok, Iraq
[2]Energy Eng. Dept., Technical College of Engineering, Duhok Polytechnic University, Duhok, Iraq
[3]ISE Dept., Erbil Technical Engineering College, Erbil Polytechnic University, Erbil, Iraq

Article Information

Submitted : 4 Mar 2024
Reviewed: 14 Mar 2024
Accepted : 1 Apr 2024

Keywords

Parallel Processing, Non-Parallel Processing, XGBoost, Classification, Text Classification

Abstract

This paper shows the findings of a study that looks at how CPU parallel processing changes the way Extreme Gradient Boosting (XGBoost) classifies text. XGBoost models can sort news stories into set groups faster and more accurately, with or without CPU parallelism. This is the main goal of the study. The Keras dataset is used to prepare the text so that the TF-IDF (Term Frequency-Inverse Document Frequency) features can be found. These features will then be used to train the XGBoost model. This is used to check out two different kinds of the XGBoost classifier. There is parallelism between one of them and not it in the other. How well the model works can be observed by how accurate it is. This includes both how long it takes to learn and estimate and how well predictions work. The models take very different amounts of time to compute, but they are all pretty close in terms of how accurate they are. Parallel processing on the CPU has made tasks proceed more rapidly, and XGBoost is now better at making the most of that speed to do its task. The purpose of the study is to show that parallel processing can speed up XGBoost models without affecting their accuracy. This is helpful for putting text into categories.

A. Introduction

Efficient classification and analysis of textual data has become essential for many applications in the information era, such as sentiment analysis and information retrieval [1][2]. Text classification is a machine learning technique that assigns a set of predefined categories to open-ended text. Particularly, articles cover an extensive range of subjects and viewpoints, which makes classifying them a difficult but essential work in the field of machine learning (ML)[3]. In addition to helping to organize large amounts of information, the ability to automatically classify articles into predetermined categories or subjects also makes it easier to deliver customized content and analyze trends across a variety of industries [4].

Of all the machine learning methods that may be used to text classification tasks, gradient boosting algorithms have proven to be particularly effective at managing sparse and high-dimensional text data [5]. A well-liked gradient boosting an approach XGBoost, has received praise and a lot of attention for its efficacy and efficiency in a variety of applications, including text classification [6]. Its appeal for text-based machine learning problems stems from its capacity to manage extensive datasets, feature interactions, and built-in parallelism [7].

Although XGBoost has demonstrated its effectiveness in a number of sectors, investigating how to improve its performance via parallel processing processes becomes a relevant research project [8]. The exponential increase in data volume demands solutions that are both scalable and computationally efficient [9]. Therefore, XGBoost's effectiveness in textual data classification can be studied in relation to CPU parallel processing in order to get insight into the relationship between algorithmic efficiency and computational resources.

The purpose of this study is to evaluate how CPU parallelism affects XGBoost's text classification performance. The main goals consist of: By using CPU parallel and non-parallel processing to categorize news articles into predetermined categories, assess the accuracy of XGBoost models. Calculate the computational efficiency for both model training and prediction with and without CPU parallelism in terms of processing time. For large-scale text categorization jobs, examine the trade-offs between computational resources and model performance to determine the practical implications.

Parallel processing involves making use of multiple processors or CPUs to efficiently handle different aspects of a single task [10]. Systems can significantly reduce a program's execution time by distributing the workload among multiple processors[11]. Multi-core processors, commonly found in modern computers, and any system with more than one CPU have the ability to execute parallel processing [12]. Multi-core processors are integrated circuit (IC) chips with two or more CPUs, offering improved speed, lower power consumption, and more effective handling of several activities[13]. Many computers are equipped with a range of cores, varying from two to four, and in some cases, even up to twelve to sixteen. Advanced algorithms and calculations are often executed using parallel processing techniques[14]. When it comes to distinguishing between parallel and serial operations, the way registers are used plays a crucial role[15]. Shift registers function by processing bits sequentially, handling one bit at a time. On the other hand, registers with parallel loading handle each bit of the word simultaneously. It is possible to optimize parallel processing by utilizing a range of functional units

that can perform various activities simultaneously, enhancing overall complexity management[16].

The paper proceeds with a literature review, then comprehensive methodology detailing the dataset, preprocessing techniques, and the experimental setup, followed by a presentation and analysis of the obtained results. Subsequently, a discussion section elucidates the implications of the findings, highlighting the significance of parallel processing in enhancing computational efficiency without compromising accuracy. The conclusion synthesizes the key insights gleaned from the research, emphasizing the practical implications and potential avenues for future investigations.

B. Background Theory

Parallel Processing

Parallel processing is a mathematical model that revolutionizes the traditional sequential approach by allowing multiple tasks to be carried out simultaneously. Parallel processing, rooted in both philosophy and technological progress, has become the cornerstone of addressing the increasing demands for complex calculations and data-intensive applications[17].

In essence, parallel processing is inspired by the idea that dividing the problem into smaller, easier to manage and solve simultaneously leads to faster and more efficient solutions. This philosophy corresponds to the old proverb, "divide and oppression," with emphasis on the possibility of parallelism to promote problem-solving strategies [18].

The philosophical basis for parallel processing also extends to the concept of cooperation, reflecting how human beings often work together to achieve a common goal. Through the use of parallel computing, systems simulate a collaborative effort, with workload distributed among multiple or core processors to accomplish tasks collectively. Parallel treatment of applications discoveries in a variety of areas, transforming how complex problems are addressed: Scientific simulation, big data analysis, schematic conversion, high-performance computing [19].

Approaches to Parallel Processing

Parallel processing can be achieved through different approaches, each tailored to specific accounting requirements [20]. The main approaches are:

Task Parallelism: In this approach, a larger function is divided into smaller sub-tasks that can be carried out independently. Each basic processor or processor performs a specific sub-function simultaneously, enabling efficient use of resources. The same functions are well suited for applications that by their very nature involve divisive workloads, such as scientific simulations and image processing.

Data Parallelism: This approach involves dividing the data set into smaller pieces and simultaneously processing them. Multiprocessors work on different parts of the data simultaneously, often with the same set of instructions. Data matching is usually used in applications such as parallel databases and multimedia processing, where processes can be applied uniformly to subsets of data.

Benefits of Parallel Processing:

The adoption of parallel processing offers a wide range of benefits [21], resulting in its widespread use in various areas [22]:

Increased speed and performance: The direct advantage of parallel processing is the significant enhancement of math speed. By dividing tasks or data into smaller parts, multiple treatments can work simultaneously, leading to faster implementation times [23].

Scalability: Parallel processing systems can easily expand by adding more processors or basics. Such interchangeability ensures that, as computational demands increase, the system can adapt to and deal with the larger workload without sacrificing performance [24].

Tolerance: Parallel systems often show tolerance for errors where tasks can be distributed to multiple processors. If one of the healers fails, others can continue to carry out the tasks assigned to them, while ensuring the continued operation of public order [25].

Resource efficiency: Parallel processing maximizes the use of resources by enabling multiple functions to function simultaneously. This efficiency is critical in scenarios where time-sensitive calculations or large data sets need to be processed quickly [26].

eXtreme Gradient Boosting (XGBoost)

The XGBoost has emerged as a powerful and efficient algorithm, particularly known for its effectiveness in forecasting modelling and data analysis. XGBoost belongs to the ensemble learning algorithm category, which combines the predictions of multiple vulnerable learners to create a strong and accurate model [27].

In essence, XGBoost is an extension of the progressive enhancement algorithm, a technology that builds a predictive model by combining the products of simpler models called vulnerable learners or decision trees. The difference between XGBoost is its focus on achieving the computational efficiency and predictive performance of the model through a process called boosting. The promotion involves the repeated addition of vulnerable learners to the model, with each new learner attempting to correct the mistakes made by the common model of previous repetitions. The "extreme" in XGBoost refers to its extreme focus on mathematical speed and model performance, which is achieved through parallel processing, organization techniques, and innovative tree-building algorithms.

Machine Learning and Text Classification

machine learning, a subset of artificial intelligence, involves the development of algorithms that enable systems to learn and make predictions or decisions without explicit programming [28]. Text classification is a common application of automatic learning, where algorithms are trained in the classification and attribution of textual data based on its content [29].

The purpose of the classification in the text is to automatically assign predetermined categories or insignia to the text documents. This could include tasks such as e-mail mail disclosure, analysis of feelings in social media sites, or classification of news articles [30]. Automated learning algorithms, including

XGBoost algorithms, are superior to the complexity of the natural language and the extraction of meaningful patterns of textual data [31].

XGBoost can be used effectively in text classification functions by representing text data as digital features. Techniques such as the frequency of wordbag documents or the frequency of reverse-frequency documents (TF-IDF) are usually used to convert the text into an appropriate form for automated learning algorithms. Once the data are converted, XGBoost can be trained in the tagged data set, learning patterns and the relationships between features and signs.

The advantages of using XGBoost to classify texts include its ability to deal with high-dimensional data, capture complex relationships, and provide accurate forecasts even in the presence of irregular noise or text. Its robustness and efficiency make it a preferred option for a wide range of applications in which the classification of textual information is crucial.

In conclusion, XGBoost stands as a powerful machine learning algorithm, especially as a genius in handling complex tasks such as disaggregating texts. Its emphasis on parallel enhancement, organization and processing contributes to its success in creating accurate and efficient predictive models, making them a valuable tool in ever-expanding automation.

C. Literature Review

The implementation of parallel processing has greatly accelerated training on the convolutional neural network (CNN) [32]. When paralleled with the software, the version using a multiple core of central processing units showed a high-speed advantage over the same core. It should be noted that the addition of additional processing units not only increases speed but also increases accuracy. Parallel processing plays a pivotal role in maximizing the use of the Central Processing Units and minimizing the time it is disrupted. This optimal allocation of resources ensures efficient utilization of mathematical energy throughout the CNN training process, contributing to both faster implementation and improved accuracy of models.

It is proposed that the use of the Algorithm of Ant colony become significantly more efficient in processing more training documents when implemented in a multi-core environment [33]. In such circumstances, the advantages of multiple rationalization are useful and effectively mitigate the potential overhead costs associated with the operation of a multi-thread and multi-core configuration. This dynamic interaction between multiple threads and basic intentions enhances algorithm performance, enabling the processing of large-scale data sets in a more streamlined and time-effective manner. The inherent synergy between multi-component and multi-purpose capacities not only optimizes the use of resources, but also emphasizes the ability of the Algorithm of Ant colony to adapt to smooth growth to the increasing demands of larger data sets.

The main focus is the multi-pronged Intel Accelerator, Xeon Phi Knights Landing, and Xeon Multi-core, which boasts the ability to support hundreds of leads on one central processing unit [34]. The key elements of performance in these structures are accurate data mapping and data-line strategies. A careful examination of the impact of mapping strategies reveals that automatic learning

applications on many infrastructures can be significantly accelerated through the implementation of smart mapping policies. This underscores the crucial role played by calculated data mapping and the thread in optimizing the performance of complex applications, particularly in the area of automation of advanced parallel processing equipment.

In their innovative approach [35], researchers call for the integration of automated learning with simultaneous data processing in order to enhance the detection of threats to the Internet from objects (IOT). It identifies the basic products designed to detect the threat posed by CPU and introduces a new way of integrating them smoothly. The effectiveness of the classification process is measured by precision, with both training and testing time being considered as complementary measures. In order to enhance the model efficiency, researchers recommend that the Apache Spark program be activated in a multi-threaded manner. This strategic implementation not only accelerates the training and testing phases, but also underlines the importance of taking advantage of contemporary frameworks to streamline automated learning applications in dynamic and data-intensive environments.

The investigation delves into the impact of CUDA-accelerated GPU and CPU computation on image processing [36]. Both sequential C and parallel CUDA applications are employed to filter and process images, allowing for a comprehensive comparison. The execution time on the GPU is meticulously measured using CUDA Events, providing valuable insights into the efficiency of the parallel computing approach. Notably, the studies reveal a significant enhancement in image processing and filtering efficiency with GPU CUDA programming. This underscores the potency of leveraging parallel computing architectures, particularly CUDA-accelerated GPUs, to achieve accelerated and more efficient image processing workflows compared to traditional sequential CPU approaches.

Its comprehensive analysis entails a careful comparison of the performance of the GPU and the CPU through several independent processes to ensure that the Platform is the most efficient [37]. With many experiences in diverse contexts, these experiences reveal accurate views on the relative efficiency of GPU and CPU. While the results emphasize the inherent strength of GPU, which are curious surfaces - there are cases where the CPU exceeds the GPU treated in performance. This precise conclusion implies that GPU may not always be the definitive solution to parallel many separate stages, emphasizing the importance of careful scrutiny when choosing the optimal calculation platform based on specific accounting requirements and limitations.

The study not only demonstrated the real-time implementation of image processing applications, but also the effective use of multi-core and multi-processing technology in this context [38]. The key to real-time processing lies in the development of advanced algorithms and parallel approaches. Specifically, researchers focused on the fragmentation of parallel images through the threshold applied to a large number of images (K images), covering the entire surface of mineral, dynamic and cylinder objects. This complex application of parallel computer techniques emphasizes the diversity of multi-core and multi-processing technologies in addressing complex challenges in real-time image processing and opening up ways to enhance efficiency in diverse scenarios.

D. Methodology

The methodology used in this research involves a systematic approach to assessing the performance of XGBoost in the classification functions using the Reuters Newswire Dataset. The process begins with obtaining data and pre-processing, including uploading and dividing the data set, deciphering word indicator codes and converting text data into TF-IDF vectors. Subsequently, two XGBoost models are being developed and trained: one through non-parallel treatment by the central processing unit and the other through parallel processor treatment. The models are then evaluated on the basis of an independent test package and key performance measures, including accuracy and processing time, are recorded. The experimental composition includes a coherent environment, and the analysis focuses on comparing the accuracy and computational efficiency of models. This methodology ensures a thorough examination of the behavior of XGBoost in the classification of news articles, highlighting the implications of parallel processing of text classification functions by central processing units. Our proposed methodology is illustrated in Figure 1.

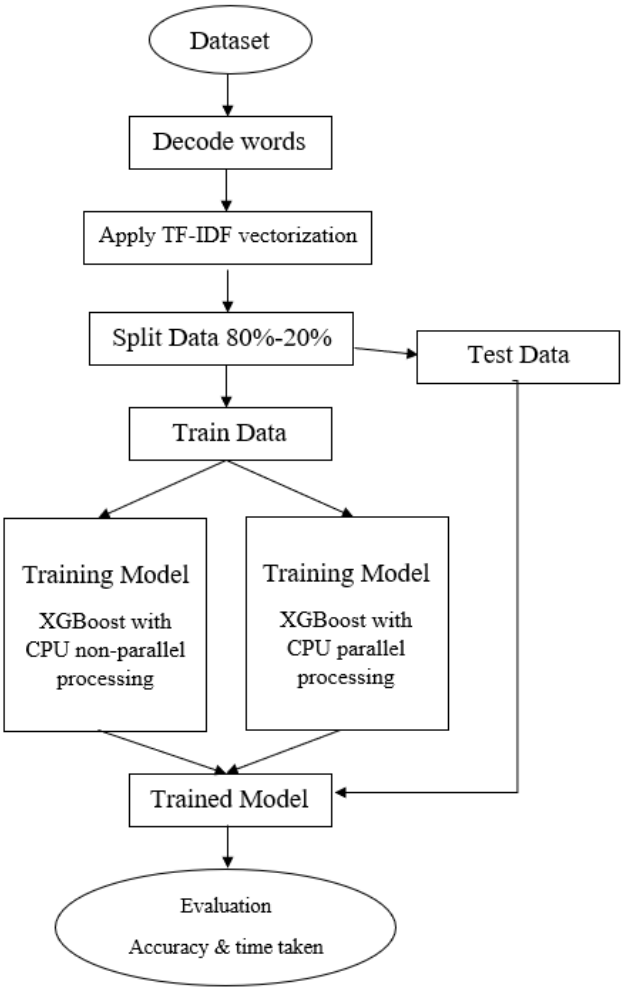

Figure 1. Flowchart for our proposed approaches

Dataset Description

The Reuters Newswire dataset [39] is used in this research, accessible through the TensorFlow Keras Library, is the cornerstone of natural language processing and machine learning research. This dataset includes a diverse and extensive collection of news articles, which are strictly organized in 46 predefined topics. It is a valuable resource for text classification tasks, allowing researchers to explore and develop models that can accurately classify news articles into distinct topics. The wealth of the data set lies not only in its size, but also in the importance of its content in the real world, where it covers a wide range of global events and issues. Each of the news articles in the data set is represented as a series of word indicators, facilitating their integration into machine learning workflows. The Reuters Dataset data set is especially equipped for training and algorithm evaluation because of its inherent nature in the Multi-class Classification, making it a standard for evaluating the performance of models designed to deal with complex and diverse textual data. This data set has played a pivotal role in advancing the area of classification of texts, as a benchmark for assessing the effectiveness of new algorithms and methodologies. Its availability within the Kiras ecosystem ensures accessibility, making it an indispensable asset for researchers in the processing of natural languages. The dataset is divided into training and test sets, with a predefined split of 80% for training and 20% for testing.

Preprocessing

The dataset is loaded using Keras, resulting in a collection of integer sequences representing the indices of words in the articles. To retrieve the actual text content from these indices, a mapping from index to word is created using the provided word index dictionary. Unknown words are denoted by a '?' character. This conversion facilitates the transformation of integer sequences back into human-readable text for analysis and feature extraction purposes [40].

The textual data is preprocessed to generate features suitable for training XGBoost models. To achieve this, the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization technique is employed. The TF-IDF Vectorizer from the scikit-learn library is utilized to convert the text corpus into a matrix of TF-IDF features. This transformation assigns weights to words based on their frequency in individual documents and their occurrence across the entire corpus, providing a numerical representation of the text suitable for ML algorithms [41].

XGBoost Models

XGBoost (Extreme Gradient Boosting) is a popular and powerful supervised learning algorithm used for both regression and classification tasks. It belongs to the family of gradient boosting algorithms and is known for its speed, scalability, and accuracy in handling structured/tabular data [27].

XGBoost builds an ensemble of weak predictive models, typically decision trees, in a sequential manner. Each new tree that is added to the ensemble corrects the errors made by the existing set of trees. It minimizes a predefined loss function by fitting new models to the residuals (the differences between the predicted and actual values). This iterative process continues until a specified number of trees is

reached or no further improvements can be made. Figure 2 illustrate the general structure of XGBoost.

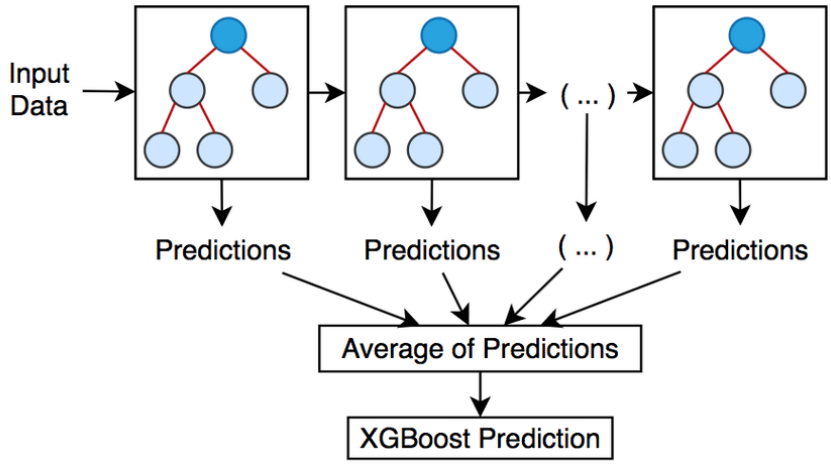

Figure 2:.General structure of XGBoost [42]

The objective function for the XGBoost classifier can be represented in equation 1:

$$Objective\ Function = \sum_{i=1}^{n} \ell(y_i, \hat{y}_i) + \sum_{i=1}^{k} \Omega(f_{\mathcal{K}}) \quad \text{.....(1)}$$

Where:

- ℓ is the loss function that measures the difference between the predicted $(\hat{y}_i)$ and actual (y_i) values.
- Ω is the regularization term that penalizes complex models to prevent overfitting.
- $f_{\mathcal{K}}$ represents the $\mathcal{K}$th tree in the ensemble.

The overall objective is to minimize this function by adjusting the model parameters during the training process.

XGBoost has hyperparameters that can be tuned to control the learning process, such as the number of trees, the learning rate (shrinkage), tree depth, regularization parameters, and others, allowing for optimization and improved model performance.

Two XGBoost classifiers are trained and evaluated to assess the impact of CPU parallel processing on model performance: The first XGBoost classifier is configured to operate without CPU parallelism, indicated by setting the n_jobs parameter to 1. This setting restricts the model to utilize a single CPU core for training and prediction. The second XGBoost classifier is configured to leverage CPU parallel processing, utilizing all available CPU cores for enhanced computational efficiency. This is achieved by setting the n_jobs parameter to -1, allowing XGBoost to utilize multiple CPU cores for parallelized computation.

Evaluation Metrics

The primary evaluation metric employed is accuracy, calculated as the ratio of correctly predicted labels to the total number of test samples. This metric assesses the models' ability to correctly classify news articles into their respective predefined topics [43]. Additionally, the computational efficiency of each model is measured in terms of the time taken for both training and prediction phases. Python's time module is utilized to record the elapsed time during these processes, providing insights into the computational overhead associated with CPU parallelism.

E. Results

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

The evaluation of the XGBoost models, one employing CPU parallel processing and the other without, yielded compelling insights into their performance on the Reuters Newswire Dataset. Both models exhibited noteworthy accuracy in classifying news articles into predefined categories. The achieved accuracy of 79.83% for both models showcase their efficacy in correctly categorizing news articles. Surprisingly, despite employing CPU parallel processing, the accuracy remains consistent with the non-parallel model, indicating that parallelism does not compromise the classification accuracy for this particular dataset and model configuration.

Computational Efficiency

The assessment of computational efficiency, measured in terms of the time taken for model training and prediction, demonstrates notable disparities between the models with and without CPU parallel processing. The computational time for model training and prediction indicates a substantial difference between the two XGBoost configurations. The non-parallel model required approximately 501.319 seconds, while the parallelized model showcased significantly enhanced computational efficiency, completing the same tasks in approximately 264.978 seconds.

F. Discussion and Comparison

The results highlight an intriguing scenario where both XGBoost models achieve identical accuracies in classifying news articles while showcasing considerable discrepancies in computational efficiency. Although both models attain an accuracy of 79.83%, the model leveraging CPU parallel processing demonstrates a remarkable reduction in processing time, completing the tasks approximately twice as fast as its non-parallel counterpart.

Comparison and Implications

The consistency in accuracy between the models underscores the robustness of XGBoost's classification capabilities, unaffected by the utilization of CPU parallel processing. However, the stark contrast in computational efficiency suggests a

tangible advantage in employing parallelism for reducing processing time, especially in resource-intensive tasks.

The reduction in processing time from approximately 501.319 seconds to 264.978 seconds indicates the substantial impact of CPU parallelism on computational efficiency. This enhancement in speed without sacrificing accuracy holds significant implications, particularly in scenarios where rapid processing of large-scale textual data is paramount.

Further Analysis and Considerations

While the study primarily focuses on accuracy and computational efficiency, additional investigations into varying hyperparameters, feature engineering techniques, or alternative parallel processing approaches could provide deeper insights into optimizing model performance and resource utilization. Furthermore, exploring the scalability of parallelism for more extensive datasets or diverse text classification tasks merits further exploration.

G. Conclusion

The investigation into the impact of CPU parallel processing on the performance of XGBoost in text classification tasks using the Reuters Newswire Dataset yields valuable insights into the trade-offs between computational efficiency and accuracy. This study aimed to evaluate the influence of parallelism on model performance and computational speed, employing two XGBoost configurations with and without CPU parallelism. The attained accuracy of 79.83% for both models underscore XGBoost's robustness in accurately categorizing news articles, irrespective of CPU parallelism. Surprisingly, parallel processing did not yield a discernible increase in accuracy, indicating that the model's predictive capabilities remain consistent across configurations.

However, the significant reduction in processing time from approximately 501.319 seconds without parallelism to about 264.978 seconds with CPU parallelism highlights the pivotal role of parallel processing in enhancing computational efficiency. This enhancement in computational speed without compromising accuracy carries substantial implications for real-world applications, especially in scenarios demanding rapid analysis of voluminous textual data. In conclusion, while both XGBoost configurations showcase comparable accuracy, the adoption of CPU parallel processing demonstrates tangible advantages in computational efficiency, making it a favorable approach for accelerating text classification tasks without sacrificing predictive performance. This study emphasizes the significance of considering parallel processing strategies to optimize computational resources and expedite text analysis processes in practical applications. Further research exploring diverse datasets and alternative parallelization techniques could augment these findings and broaden their applicability in text classification and related domains.

H. References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] Z. Qi, "The text classification of theft crime based on TF-IDF and XGBoost model," in *2020 IEEE International conference on artificial intelligence and computer applications (ICAICA)*, IEEE, 2020, pp. 1241–1246.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.
[15] 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.
[16] 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.
[17] 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.
[18] H. M. Zangana and S. R. M. Zeebaree, "Distributed Systems for Artificial Intelligence in Cloud Computing: A Review of AI-Powered Applications and Services," *International Journal of Informatics, Information System and Computer Engineering (INJIISCOM)*, vol. 5, no. 1, pp. 1–20, 2024.
[19] 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.
[20] 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.
[21] K. Jacksi, N. Dimililer, and S. R. M. Zeebaree, "A survey of exploratory search systems based on LOD resources," 2015.
[22] 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.
[23] 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.
[24] 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.
[25] 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.
[26] 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.
[27] M. Nalluri, M. Pentela, and N. R. Eluri, "A Scalable Tree Boosting System: XG Boost," *Int. J. Res. Stud. Sci. Eng. Technol*, vol. 7, pp. 36–51, 2020.
[28] 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.
[29] 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.
[30] 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.
[31] V. D. Majety *et al.*, "Ensemble of Handcrafted and Deep Learning Model for Histopathological Image Classification.," *Computers, Materials & Continua*, vol. 73, no. 2, 2022.
[32] D. Datta, D. Mittal, N. P. Mathew, and J. Sairabanu, "Comparison of performance of parallel computation of CPU cores on CNN model," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, IEEE, 2020, pp. 1–8.
[33] 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.
[34] 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 *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, IEEE, 2018, pp. 329–333.
[35] 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, 2020.
[36] B. N. M. Reddy, S. Shanthala, and B. R. VijayaKumar, "Performance analysis of GPU V/S CPU for image processing applications," *Int J Res Appl Sci Eng Technol*, vol. 2, pp. 437–443, 2017.
[37] 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 & Information Technology (IJCSIT) Vol*, vol. 9, 2021.
[38] S. Aydin, R. Samet, and O. F. Bay, "Real-time parallel image processing applications on multicore CPUs with OpenMP and GPGPU with CUDA," *J Supercomput*, vol. 74, pp. 2255–2275, 2018.
[39] "Reuters newswire classification dataset." Accessed: Jan. 09, 2024. [Online]. Available: https://keras.io/api/datasets/reuters/
[40] 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.
[41] H. D. Abubakar, M. Umar, and M. A. Bakale, "Sentiment classification: Review of text vectorization methods: Bag of words, Tf-Idf, Word2vec and Doc2vec," *SLU Journal of Science and Technology*, vol. 4, no. 1 & 2, pp. 27–33, 2022.
[42] L. M. Demajo, V. Vella, and A. Dingli, "Explainable ai for interpretable credit scoring," *arXiv preprint arXiv:2012.03749*, 2020.

[43] 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.