3rd International Conference on Information Science and Technology Innovation (ICoSTEC)
July 27, 2024, Yogyakarta, Indonesia

Machine Learning Model for Diabetes Prediction Using Parallel Computing: Comparative Study

Shahd Alsaleh[1], Maha Alsayed[1], Ghadi Alkehily[1], Taif Alahmadi[1],
Lina Alrefi[1] and Malak Aljabri[2].

[1]*Department of Computer Science and Artificial Intelligence, College of Computing, Umm Al-Qura University*
Makkah 21955, Saudi Arabia

[2]*Department of Computer and Network Engineering, College of Computing Umm Al-Qura University*
Makkah 21955, Saudi Arabia

[1]shahdAls1@outlook.com
[1]maha.a.123.15@gmail.com
[1]ghadijalkehaily@gmail.com
[1]taif.alahmadi300@gmail.com
[1]l_alrefi@hotmail.com
[2]mssjabri@uqu.edu.sa

***Abstract*— Diabetes has become one of the most common and challenging health conditions in the world because it alters how the body uses glucose, an essential source of energy, and can damage organs including the kidneys, heart, eyes, and other issues in addition to the blood. Therefore, it is necessary to develop a system that can accurately identify diabetes patients using medical. indicators. Artificial intelligence (AI)-based techniques such as Machine Learning (ML) have proven to be effective in this regard. Sequential machine learning methods utilize a single underlying hardware processing element, thus having poor real-time prediction efficiency. Moreover, these approaches may struggle to handle large amounts of data due to their time-consuming nature. Parallel computing has been widely applied techniques that utilize multiple hardware processing elements to increase the application's computation time. In this study, we utilized parallel techniques in Python to train ML models and provided a comparative study for different parallel techniques. We used the Pima Indian Diabetes Dataset (PIDD), conducted five different experiments, and provided a comparative performance evaluation. We deployed two ML models which are Decision Tree (DT) and Linear Regression (LR). For each model, we compared the sequential execution with three different parallel Python techniques (multithreading, multiprocessing, and loky), each utilizing four cores. Our results showed that LR with multiprocessing technique achieved a higher accuracy of 78% and greater speedup of 39. The results in general indicated that parallel execution outperforms sequential execution in terms of speed. This comparative study provides valuable insights into how to optimize machine learning models for diabetes detection and highlights the usefulness of parallel computing technologies in healthcare applications.**

***Keywords*— parallel processing. Diabetes. machine learning. Decision Tree. Linear Regression**

I. INTRODUCTION

Diabetes affects 422 million people worldwide and is expected to increase to 490 billion by 2030, according to the World Health Organization [1]. Diabetes is a major chronic disease whose prevalence is continuously increasing. Diabetes impacts the body's ability to metabolize glucose, or sugar, which is a vital source of energy. Diabetes is classified into two types, which are type 1 diabetes and type 2 diabetes. Type 1 diabetes is an autoimmune condition in which the immune system mistakenly attacks and destroys insulin-producing beta cells in the pancreas. Type 2 diabetes is insulin resistance, where the body's cells can't process insulin properly, and it can affect children and adults. Over time, the pancreas stops producing enough insulin. Lifestyle factors, genetics, and obesity are common risk factors. Diabetes that goes undiagnosed and untreated can cause blood sugar levels to fluctuate and, in extreme cases, damage organs such as the kidneys and eyes. Early and precise diagnosis of diabetes mellitus is a significant challenge for healthcare professionals, particularly in its early stages [1].

The healthcare sector manages extensive databases, including structured, semi-structured, and unstructured data. These databases provide fertile ground for the application of big data analytics. By using machine learning techniques to analyze these healthcare databases, it becomes possible to develop AI models capable of detecting diabetes. These models have the potential to greatly enhance early diagnosis, and personalized treatment plans, and improve patient care.

Artificial intelligence (AI) techniques such as machine learning (ML) have been increasingly utilized in the medical

sector and offered valuable support by providing a reference point for gaining preliminary insights into different diseases in terms of prediction, classification, or detection which greatly contributed to reducing the workload in healthcare. A considerable amount of research has been dedicated to automating diabetes prediction through ML techniques, often utilizing different open-source datasets for analysis. Predicting diabetes early and with accurate results can save many human lives. The purpose of such investigation is to assess the ML classifiers that can predict the probability of disease in patients with the greatest precision and accuracy.

Unfortunately, ML models need a lot of computational power, which results in a longer computation time. Usually, in single-processor environments, the ML algorithms cause a significant delay in model processing from training to classification. Sequential ML computing is often inappropriate for handling huge datasets. On the other hand, parallel computation offers exceptional opportunities to implement these large-scale problems due to their ability to efficiently exploit multiple processor environments. In our study, the main purpose is to increase the ML model prediction speed by applying different parallel computing techniques for the ML models and conducting a comparative evaluation study. We studied the three parallel ML techniques in Python namely: multithreading, multiprocessing, and loky which can be effectively deployed to reduce the execution time for the ML algorithms. Those three techniques were studied in Linear Regression **(**LR) and Decision Tree (DT) algorithms. LR and DT are both fundamental ML algorithms utilized in supervised learning, each offering unique approaches to predictive modelling. LR establishes optimal linear functions by analysing dataset relationships, while DT categorizes data and forecasts outcomes through structured flowcharts. Despite their, both methods share the goal of facilitating accurate predictions and informing decision-making processes.

The key contributions of this paper can be summarized as follows:

- Conducting comparative performance evaluation for the sequential and three different parallel Python techniques (multithreading, multiprocessing, and loky) utilizing four processing cores on two ML models (LR and DT) to enhance the detection of Diabetes.
- Making a significant contribution to AI, healthcare, and parallel computing literature, especially with limited studies available for comparing the performance of parallel techniques on ML in this domain.

This paper is structured as follows: Section II clarifies the previous studies that deployed sequential and parallel ML techniques. Section III presents the methodology followed to carry out the experiments. Section IV discusses the experimental setup and results obtained. Section V provides a conclusion and future work.

II. literature Review

Diabetes is a long-term medical condition characterized by persistently high blood sugar levels brought on by inadequate insulin production, inadequate insulin utilization, or both. The use of predictive ML algorithms in the context of diabetes mellitus is explored in this section. However, there is still room for improvement in the Diabetes prediction. The objective of this study is to develop ML algorithms in parallel. It also aims to highlight the significance of early detection to reduce complications and to clarify the critical role that predictive ML algorithms play in the management of diabetes. In this section, we discuss several related articles.

A study aimed to identify the most effective ML model for diabetes prediction by Kangra et al. [2] compared different algorithms including Naïve Bayes (NB), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Decision Tree (DT), and Logistic Regression (LR), to analyze performance indices and error rates. The used dataset was The German dataset and the Pima Indian Diabetic (PID) dataset from Kaggle, with analysis conducted in WEKA 3.8.6 software using Python. The PID dataset comprised 9 attributes and 768 instances, with the goal of diabetes classification. The German dataset also had nine variables. In both cases, the primary objective was diabetic status determination. SVM performed well with a 74% accuracy for the PID dataset, while KNN and RF excelled, achieving an impressive 98.7% accuracy for the German dataset. The researchers suggested exploring hybrid models and assessing their performance alongside other algorithms, providing valuable insights for further research.

A study by Sivaranjani S et al. [3] aimed to predict diabetes using ML algorithms, which support SVM and RF. They used the PIMA [4], dataset and performed feature selection using the "wrapper method" to increase the efficiency of the model and reduce its complexity. Special analytical tools Principal Component Analysis (PCA) was used for dimensionality reduction, optimizing dataset complexity effective modeling. The results showed that feature selection affected RF and SVM classifiers accuracy, the proposed technique achieved 83% accuracy with RF and 81.4% with SVM. Future research may refine the feature selection and explore new dimensionality reduction techniques to move forward and increase the efficiency of the model.

A study by Soni M et al. [5] suggested using various classification ML methods for early diabetes prediction, which are KNN, LR, DT, SVM, Gradient Boosting (GB), and RF. All these classifications and ensemble methods were applied to detect which of them gave the highest accuracy for diabetes prediction. The PIMA [4] was used to gather the data and underwent preprocessing in two steps, which were Missing Values removal and splitting of data, utilized in all the algorithms implemented using Python. While all these algorithms demonstrated accuracy exceeding 65%, RF stood out with the highest accuracy among them at 78%, making it the most accurate algorithm in predicting diabetes.

A study by Sarwar Muhammad et al. [6] proposed the deployment of ML algorithms to predict diabetes and help healthcare professionals make timely decisions on the health and treatment of the patient. The study compared the performance of six different algorithms which are SVM, DT, LR, RF, KNN, and NB, the dataset used was the Pima Indians Diabetes [4]. The dataset was divided into two sections training

data (70%) and data (30%). The results showed that SVM and KNN achieved the highest accuracy of 77% compared to other algorithms tested.

In a study conducted by Sierra-Sosa et al. [7], researchers proposed using parallel deep-learning (DL) techniques on multiple Graphics processing units (GPUs) for predicting adverse events in patients with type 2 diabetes. The study used a database from the Basque Health Service that contained records of 150,156 patients diagnosed with type 2 diabetes mellitus. The algorithms implemented in this study included Logistic Regression (LR), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Support Vector Machines (SVM), and Recurrent Neural Network (RNN). The algorithms were implemented in TensorFlow using Python as the programming language. The results of the study showed that LDA and SVM outperformed the other algorithms in predicting Major Amputations and acute Myocardial Infarction with 97% accuracy. LDA balanced and SVM balanced achieved 92% accuracy in predicting Hospital Admission for avoidable causes. RNN with a balanced dataset performed best, achieving 94.6% accuracy in predicting at least one disease, surpassing LDA balanced and SVM weighted by 7.4%. The paper suggests expanding the dataset and exploring additional deep-learning techniques to enhance prediction accuracy in the future.

A study by Rani K [8] aimed to develop a highly accurate system for early diabetes prediction by combining various ML methods. The study used the John Diabetes Database[9], which had nine features, with 2000 data points. The study used John's diabetes database, which contains nine attributes, with 2,000 data points. The study compared the accuracy of training and testing diabetes prediction when implemented by Python on several classification methods, including KNN, LR, DT, SVM, and RF. In the study, KNN and LR achieved an accuracy of up to 78%, while (DT) on all previous algorithms with an accuracy of 99%, and the most influential attribute was "glucose". RF reported an accuracy of 94%, with "glucose" and "BMI" as key attributes. SVM delayed accuracy by 76%.

A study conducted by Shrivastava et al. [10] proposed a parallel algorithm based on an SVM to predict the likelihood of diabetes in people using a large dataset from S. S. Medical College, Rewa, Master Chart. The dataset was distributed over multiple machines to process it in parallel, and the algorithm was implemented in MATLAB 7.12.7, R2011B on a machine with 3GB RAM. The study showed that the parallel SVM has substantially lower training time compared to the sequential SVM while maintaining similar accuracy levels. The parallel implementation also showed the possibility for scalability with multiple machines, resulting in a 1/3 reduction in training time. However, for future improvements, researchers can enhance the accuracy by improving the selection of the starting point for K-means clustering when dividing data into large clusters.

Our work in this paper is similar to the previous related research in terms of using ML for Diabetes prediction. However, we provided a comparative performance evaluation study for different parallel techniques with ML models. This comparative study demonstrates the value of parallel computing technology in healthcare applications and offers insightful information about how to optimize ML models for diabetes detection.

TABLE I
Literature Review Summary

Year	Algorithm	Dataset	Performance measures
2023	NB, RF, SVM, KNN,DT and LR	Germany and Pima Indian diabetic (PID) diabetes datasets	Identifying the accuracy
2021	SVM and (RF).	The PIMA Indianas Diabetes (PID)	The performance measures mentioned in the paper are test accuracy, validation accuracy, sensitivity, and specificity
2020	KNN, LR, DT,RF ,GP and SVM	The PIMA Indianas Diabetes (PID)	Identifying the accuracy
2020	KNN, LR, DT,RF and SVM	John Diabetes Database	Identifying the accuracy
2019	SVM	From S. S. Medical College, Rewa, Master Chart	Identifying the accuracy and time in sec.
2018	SVM, DT, LR, RF, KNN, and NB	The PIMA Indianas Diabetes (PID)	Identifying the accuracy
2011	LR, LDA,QDA,SVM, and RNN	Form. The Basque Health Service	Identifying the accuracy, precision, recall, and F1-score.

III. Methodology

We trained the parallel ML models in Python using various multiprocessing and multithreading strategies. Then, we measured the training time in sequential execution and compared it to the parallel execution time, with a publicly accessible dataset, which distinguishes diabetes from other medical conditions. An overview of the methodology steps that were performed is demonstrated in Fig.1 and is covered in more detail in the following subsections.

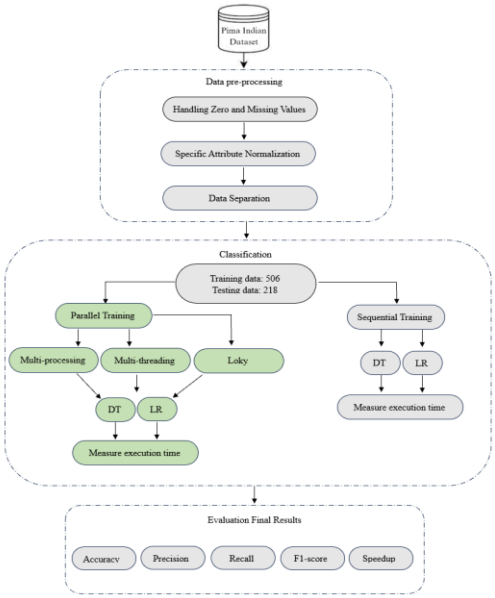

Fig.1 The methodology adopted

A. Dataset Description:

TABLE II
Data Set Statistics

PIDD	Dataset
768	Samples
8	I/P Attributes
2	O/P Classes
9	Total Attributes
Nil	Missing values
Nil	Noisy-Attributes

The dataset used was developed by the National Institute of Diabetes and Digestive and Kidney Diseases (PIMA), and its primary purpose is to predict the likelihood of a patient developing diabetes based on specific diagnostic tests. This dataset has been updated biennially since 1965[7].

Tables II and III, show the diabetic dataset characteristics, data type, and statistics. The classifier predicts whether the patient has diabetes based on the dataset characteristic, which is intended for binary classification, with class value 1 showing persons who have gotten a “diagnosis of diabetes” and class 0 standing for those who have not obtained a diabetes diagnosis.

There are 768 records in the dataset, 500 for training and 268 for testing, with no missing values. It is also worth noting that the dataset holds impractical values like zero for body mass index and insulin. There are a total of nine features, eight of which are independent, and one is target.

TABLE III
Attribute Description

Sl.No	Attribute	Description
1	No. of times Pregnant	Discrete data of type int64
2	Plasma Glucose Concentration	Discrete data of type int64
3	Diastolic Blood Pressure (mm Hg)	Discrete data of type int64
4	Skin Thickness (mm)	Discrete data of type int64
5	Insulin mu U/ml	Discrete data of type int64
6	BMI (Weight/ Height) (kg/m2)	Continuous data of type int64
7	Diabetes Pedigree Function	Continuous data of type int64
8	Age	Discrete data of type int64
9	Outcome class	Discrete data of type int64

B. Data Preprocessing

We noticed biologically implausible zero values in the PIMA dataset, so we employed a hybrid approach to manage zero values, eliminating missing values from the parameters that have the greatest effect on the outcome and normalizing the remaining parameters using the median, to create efficient models capable of accurately detecting cases of diabetes and non-diabetes. We normalized Skin Thickness and Insulin attributes by filling null values by median and then eliminating zero values by median because blood pressure and glucose are critical for deciding diabetes and their null fraction is extremely small, so it should be better to remove invalid entries. We then

separated the clean dataset into two datasets: training 70% and testing 30%, using the sk-learn train split technique[11].

C. Classification Phase

We assessed two ML, which are DT and LR, to train and test the dataset for predicting diabetes. Both sequential and parallel training methods were used for all the models. In sequential training, only one execution thread was used to train the models; in parallel training execution, 4 threads were used to train the models.

1) Sequential Training: In terms of sequential training, The DT and LR models were sequentially trained to gather performance metrics and keep track of execution time.

DT is a supervised learning algorithm used to solve classification and regression problems. It is usually preferred in such cases. The classifier has a tree-like structure, where the internal nodes represent the dataset's features, the branches decide the decision-making process, and each leaf node stands for the classification result [12].

LR is a type of supervised ML algorithm that decides how dependent a variable is, and how one or more independent features are related linearly. The algorithm seeks out the best linear equation that can predict the value of the dependent variable based on the independent variables [13].

2) Parallel Training: In the following subsections, the techniques adopted for parallel training of the DT and LR models are described. Threading Backend and Number of Cores in Python Parallel In this technique, the sklearn joblib library was imported into the working environment to use Python's built-in parallel backend threading features. The environment's ML models can be parallel trained by using the commands import joblib and from joblib import parallel backend. which enables them to utilize all the machine's available cores, speeding up the training process. Moreover, we added another argument called n_jobs that gives us more control over the number of active threads or CPU cores, which means that we use multi-threading to parallelize the training process and allocate a specific number of threads for that purpose. For example, n_jobs = -1 instructs the computer to use every available thread, while n_jobs = 1 runs the program in a single thread, and so on. Before fitting the model can be written this line of code "with parallel_backend('threading', n_jobs=#)" where # represents the number of cores or processes to utilize to train the models parallelly.

Multi-Processing Backend and Number of Cores in Python Parallel This technique makes use of the same code and libraries as those mentioned above. However, it makes use of processes rather than threads. The "multiprocessing" backend utilizes individual processes, making it a suitable choice for parallelism within a single host. However, it's considered a legacy approach. The code for the "multiprocessing" technique is implemented using "with parallel_backend ('multiprocessing', n_jobs=#) "Loky Backend and Number of Cores in Python Parallel. The parallel backend libraries and codes of Python are also used in this technique. The "loky" backend allows for adaptive parallelism and is particularly valuable for tasks that benefit from efficient multi-processing execution. It is based on single-host processes and is primarily based on multi-processing execution. To implement the "loky" technique, use this line "with parallel_backend ('loky', n_jobs=#) ".

D. Evaluation Phase

Different performance metrics can be used to compare ML classifiers. In this study, the models that were deployed were evaluated and their performances were compared using accuracy, precision, recall, and F1-score. The confusion matrix, which is used to evaluate the accuracy of the classifier, supplies a comparison of the model's predicted classifications alongside the actual classifications and it consists of four main values. In addition, the execution times for both sequential and parallel programs were given in seconds, and the speedup was calculated using this data.

- True Positive (TP): The model correctly predicts that a person has diabetes, and the person does have diabetes.
- False Positive (FP): The model incorrectly predicts that a person has diabetes when the person does not have diabetes.
- True Negative (TN): The model correctly predicts that a person does not have diabetes, and the person does not have diabetes.
- False Negative (FN): The model incorrectly predicts that a person does not have diabetes when the person does have diabetes.

Accuracy is the percentage of how often a classification ML model is correct overall. By comparing the number of accurate predictions to the total number of instances in a dataset and calculated using the following formula:

$$Accuracy = (TP + TN)/(TP + FP + TN + FN)$$

Precision refers to the number of true positives divided by the total number of positive predictions and is calculated using the following formula:

$$Precision = TP/(TP + FP)$$

Recall is the proportion of correctly predicted classes to all positive classes and is calculated using the following formula:

$$Recall = TP/(TP + FN)$$

F1-score supplies a single metric that balances the trade-off between precision and recall, and uses the following formula:

$$F1 - score = (2 \times Precision \times Recall)/(Precision + Recall)$$

Speedup is the proportion of the time it takes to complete a task sequentially to the time it takes to complete the same task in parallel calculated by following the formula:

$$Speedup = (SequentialExecutionTime) /(ParallelExecutionTime)$$

IV. Result and Discussion

A. *Experimental Setup*

To conduct the experiments, we used Python 3.11.5 on the Visual Studio code platform. The computer that has been used runs Windows 11 operating system. The device processor was an AMD Ryzen7 5800H with 16 logical processors and eight cores with RAM 16 GB. The Pima Indian dataset selected features nine attributes as follows: number of pregnancies, plasma glucose concentration, diastolic blood pressure, triceps skinfold thickness, 2-hour serum insulin, Body Mass Index (BMI), Diabetes Pedigree Function, age, and class variable that indicates whether a patient has diabetes or not as shown in Table IV. After data processing the total amount of instances became 724, from this data 506 were used for training and 218 for testing. Overall, DT and LR were used in five experiments. Both sequential and parallel computing were included in each of them. Parallel computing was based on threading, multiprocessing, and loky.

TABLE IV
Evaluation Results Obtained

	DT		LR	
	Class 0 (true negative)	Class 1 (true positive)	Class 0 (true negative)	Class 1 (true positive)
Precision	74%	63%	77%	83%
Recall	81%	53%	94%	50%
F1-score	77%	57%	85%	62%
Accuracy	70%		78%	

B. *Result*

By applying different parallel techniques that remark on the methodology section, the accuracy, precision, recall, F1-score, execution time, and speedup that measured for each parallel technique and sequentially. The results of the evaluation metrics were stable, and no changes were observed while performing our experiments, which shows that our experiments did not suffer any trade-off between speed and performance. The Table IV shows a summary of the results of the evaluation metrics found for each model for two classes, including "true positive" and "true negative." Both models performed well. However, LR outperformed DT with 78% accuracy while DT had 70% accuracy. The precision results showed that LR has the highest precision in the "true negative" and "true positive" classes at 77% and 83%, respectively. In comparison, DT has the lowest precision in the "true negative" and "true positive" classes at 74%,63% respectively.

The recall that measures the model's ability to accurately identify all related instances of a given class contained by a dataset shows that LR outperformed DT in identifying "true negative" class with 94% while DT was 81%, whereas in identifying "true positive" class, DT was outstripped with 53% and LR with 50%. For F1-score that reflects the balance between precision and recall indicates that LR had a better score than DT, this concludes that LR was more accurate in predicting diabetes than DT.

The data used shows how applying the parallelization technique to LR and DT models minimizes execution times. Three parallel approaches employed are Threading, multiprocessing, and multiprocessing with Loky, each of them *has its outcome. The speedup values for linear regression,* which range from roughly 3.1 to 3.9, show us that all parallelization techniques perform better than the sequential approach, with multiprocessing producing the maximum speed-up. While the decision tree model also satisfier results from parallelization, the speedup values are lower compared to LR, ranging from about 1.15 to 1.42, with Threading providing the best increase of the speed in this model. The calculated efficiency values, which range from approximately 0.7759 to 0.9853 for Linear Regression and from 0.2884 to 0.3555 for the Decision Tree, reveal how effectively the parallelization techniques utilize the available processors knowing that Values closer to 1 have higher efficiency, which makes the efficiency of LR is also better than DT. As a result, the Linear Regression model is more adaptable to parallelization than the Decision tree, especially the Multiprocessing technique. The Decision Tree still performs respectably with Threading. The execution times, speedups, and efficiency for various approaches and classifiers are displayed in the table below in Table V, also in Fig.2 and Fig.3.

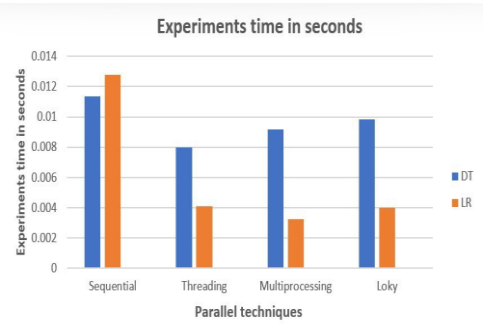

Fig.2 Execution time for both models

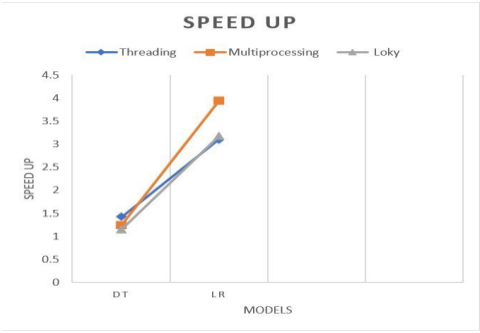

Fig.3 Compare the two models' speed up

TABLE V
The experiment results summary

	Sequential	Threading			Multiprocessing			Multiprocessing with Loky		
	Execution Time in(S)	Execution Time in(S)	Speedup	Efficiency	Execution Time in(S)	Speedup	Efficiency	Execution Time in(S)	Speed up	Efficiency
DT	0.011	0.008	1.422	0.356	0.009	1.238	0.309	0.010	1.153	0.288
LR	0.013	0.004	3.104	0.7759	0.003	3.9412	0.98533	0.0040	3.182	0.796

Although the performance of parallel techniques can vary amongst several hardware characteristics, it is important to know that not all models respond positively to a particular technique for instance, a single technique that produces accurate results for one model could produce poor results for a different model. Therefore, before putting parallel procedures into practice, it is necessary to study and evaluate their compatibility with various models carefully.

TABLE VI
Summarizes the Results of the Related Work

Study	Algorithm	Dataset	Performance measure	Result
[2]	SVM RF, KNN	PIMA German dataset	accuracy	74% 98.7%
[3]	SVM RF	PIMA	accuracy	81.4% 83%
[5]	RF KNN, LR, DT, SVM, GB	PIMA	accuracy	78% 65%
[6]	SVM, KNN	PIMA	accuracy	77%
[7]	LDA, SVM, RNN	Private dataset	accuracy	%97 %92 94.6%
[8]	KNN and LR DT RF	John Diabetes	accuracy	78% 99% 76%.
[10]	SVM	Private dataset	accuracy	Parallel 77% sequential 74%

According to the comparison in Table VI, we have found research studies related to computing. One of these studies is based on parallel computing with GPUs, while the others are based on sequential computing. Upon analysis, it was observed that even though the accuracy of both algorithms is good, there are certain issues with the amount of time consumed by each algorithm.

V. Conclusion and Future Work

In conclusion, diabetes is a disease that causes many serious complications that greatly affect human health and should be predicted early with high performance, less computation time, and more accuracy to save many human lives. The present study was designed and focused on advancing the prediction and classification of diabetes through the implementation of parallel processing techniques to expedite machine learning model training. This study has presented the ML predictive algorithms, specifically focusing on DT and LR classifiers. The algorithms were executed sequentially, and subsequent parallel experiments were conducted utilizing multiprocessing, loky backend, and threading backend. The trial was conducted using the PIDD and executed through the utilization of Google Colab. This research also discussed the performance of each technique in both models was thoroughly evaluated, measuring execution

time and speed-up. The main finding can be summarized as employing parallel processing techniques significantly reduced the sequential execution time of the models, with a maximum speed-up of 3.9 achieved for the LR model using Python multiprocessing backend with four jobs. Remarkably, there was no compromise between performance and execution speed. Additionally, The LR model achieved the highest accuracy of 78%, surpassing the DT model, which attained 70%.

Future research should therefore concentrate on the investigation of improvements that can be made by utilizing larger datasets and exploring other machine-learning and deep-learning algorithms. This work lays the groundwork for future research and developments in the fields of machine learning process optimization and healthcare applications. The integration of deep learning techniques with parallel computing is expected to shape the future trend of diabetes machine learning algorithms, which will address diabetes production through parallel computing. This evolution has the potential to greatly advance medical applications and support the ongoing development of diabetes-related predictive models.

REFERENCES

[1] "Diabetes." Accessed: Jan. 24, 2024. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/diabetes

[2] K. Kangra and J. Singh, "Comparative analysis of predictive machine learning algorithms for diabetes mellitus", *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 3, pp. 1728–1737, Jun. 2023, doi: 10.11591/EEI.V12I3.4412.

[3] S. Sivaranjani, S. Ananya, J. Aravinth, and R. Karthika, "Diabetes Prediction using Machine Learning Algorithms with Feature Selection and Dimensionality Reduction", in *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2021, vol. 1, pp. 141–146, doi: 10.1109/ICACCS51430.2021.9441935.

[4] *Pima Indians Diabetes Database*. (n.d.). Retrieved February 27, 2024, from https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

[5] M. Soni and D. S. Varma, "Diabetes Prediction using Machine Learning Techniques", *International Journal of Engineering Research & Technology*, vol. 9, no. 9, Oct. 2020, doi: 10.17577/IJERTV9IS090496.

[6] M. A. Sarwar, N. Kamal, W. Hamid, and M. A. Shah, "Prediction of Diabetes Using Machine Learning Algorithms in Healthcare", *2018 24th International Conference on Automation and Computing (ICAC)*, pp. 1–6, 2018.

[7] D. Sierra-Sosa et al., "Scalable healthcare assessment for diabetic patients using deep learning on multiple GPUs," IEEE Transactions on Industrial Informatics, vol. 15, no. 10, pp. 5682–5689, Oct. 2019, doi: 10.1109/tii.2019.2919168.

[8] K. M. J. Rani, "Diabetes Prediction Using Machine Learning", *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pp. 294–305, Jul. 2020, doi: 10.32628/CSEIT206463.

[9] *diabetes*. (n.d.). Retrieved February 27, 2024, from https://www.kaggle.com/datasets/johndasilva/diabetes

[10] K. S. Naveen, P. Saurabh, and V. Bhupendra, "An efficient approach parallel support vector machine for classification of diabetes dataset," International Journal of Computer Applications, vol. 36, no. 6, pp. 19–24, Dec. 2011, [Online]. Available: https://research.ijcaonline.org/volume36/number6/pxc3976342.pdf

[11] "AnalyzingPima-Indian-Diabetes-dataset". https://medium.com/analytics-vidhya/analyzing-pima-indian-diabetes-dataset-36d02a8a10e5.

[12] "Decision Tree Algorithm in Machine Learning - Javatpoint". https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm.

[13] "LINEAR REGRESSION IN MACHINE LEARNING - GEEKSFORGEEKS". HTTPS://WWW.GEEKSFORGEEKS.ORG/ML-LINEAR-REGRESSION/.