Machine Learning Aids Predict the Execution Times of Equations for Predictive Performance Analysis

Yasmin Makki Mohialden[١], Nadia Mahmood Hussien[٢], Zeyad Farooq Lutfi[٣], Samira Abdul Kader Hussain[٤], Ethar Abdul Wahhab Hachim[٥]

[١,٢,٣,٤,٥] **Computer Science Department, Collage of Science, Mustansiriyah University, Baghdad-Iraq**

[١]ymmiraq٢٠٠٩@uomustansiriyah.edu.iq

[٢]nadia.cs٨٩@uomustansiriyah.edu.iq

[٣]zeyadfa٦@uomustansiriyah.edu.iq

[٤]samiracs@uomustansiriyah.edu.iq

[٥]ethar٢٠١١٢٤@uomustansiriyah.edu.iq

Abstract

This paper develops a machine learning system which creates forecasts about equation runtime duration. The solution aims at simplifying resource utilization forecasts and scalability assessment for computational programs. The proposed method utilizes standard supervised learning algorithm linear regression to create predictions regarding runtime from sent equation lengths. Python code production involves generating random data followed by runtime equation resolution timing and linear regression model development which utilizes input size as the model independent variable while using expected runtime as the dependent variable. The analysis indicates that the developed model achieves proper runtime prediction capabilities across various input data ranges. The research evaluates the potential of this application method to help select algorithms and determine their complexity scales. Machine learning techniques used for runtime estimation enhance computing environments by enabling improved performance analysis and decision-making through machine learning predictors in combination with algorithm scaling methods and runtime estimation and performance assessment capabilities.

Keywords: Machine learning predictor, Estimation of running times, Performance analysis, Runtime prediction, Linear regression.

١. Introduction

It's essential to know how long programs take to run in computer systems if you want to get the most out of your resources and do a better job analyzing performance. To scale algorithms and make decisions, it is essential to correctly estimate how long a complex equation will take to run. The runtime is often figured out through theory analysis or real-world observations, which may not always give accurate predictions for changing input sizes. Many people are interested in using machine learning methods to improve runtime estimation [١, ٢, ٣].

Because time-dependent mathematical models are harder to simulate numerically, they may use too many computer resources in processing power or memory storage. Numerical simulations of a given model must often be run more than once (called "multi-query") and with many different inputs for various applications, such as sensitivity analysis, optimization, control, uncertainty quantification, or dealing with multiscale issues. Even if complex mathematical models are accurate and reliable, they may not be helpful for forecasting in computational medicine and meteorology if they can't be solved almost instantly [٤, ٥, ٦].

When domain knowledge isn't enough, for example, to pick the correct parameters or models, machine learning techniques are often used to replace those parts of the process. This leads to hybrid methods, which can be new models for reducing model order or faster, more reliable solvers [٧, ٨, ٩].

This study looks at how machine learning can predict complicated equations' runtime based on their inputs' size. The idea is to make a predictive model that can accurately predict how long equations will take to run. This will allow computer systems to scale algorithms more successfully and analyze resource allocation more effectively. Using linear regression, the suggested way connects the input size and the time it takes for the equations to run. It is a popular supervised learning method to find a link between the input size and the time it takes for equations to run [١٠, ١١].

This paper is set up as follows: In this section, an introduction to the study question and its importance to computational systems are given. In Section ٢, it talks about the related work. In section ٣ "Methodology" explains the techniques used to make the predictive model for predicting equation runtimes. Results and Analyses, ٤th Section In this part, the results show how well the models work with statistical analyses and visuals are provided. Lastly section ٥ presents the conclusion and suggestions for future work.

٢. Related work

In [٢٠١٨] Image binarization is considered as the first step in analyzing old papers. Even though the ink is fading, there are spots, and it's bleeding through, it still makes a line between the center and the background. When studying extensive document image archives, it's essential to do binarization quickly because even small inefficiencies can waste years of processing time. Binarization is critical for companies and states that want to look at large groups of documents. This means that work needs to be sped up without slowing down binaryization. The authors get ٣,٥ times faster processing by correctly mapping a state-of-the-art binarization method to a heterogeneous CPU-GPU architecture. The authors' algorithm cuts execution time by ١,٧ compared to earlier ways of tweaking parameters. For the chosen method, machine learning-based parameter adjustment is faster regarding absolute processing times than heterogeneous computing [١٢].

In [٢٠٢٠] The authors provide MATH, a new dataset comprising ١٢,٥٠٠ complex competition mathematics tasks. Each MATH issue has a comprehensive step-by-step solution that may be used to train models to create answer derivations and explanations. They also contribute an extensive auxiliary pretraining dataset that helps teach models the principles of mathematics to facilitate future research and improve MATH accuracy. While we can improve MATH accuracy, our results reveal that accuracy remains relatively low, even with massive Transformer models. Furthermore, if scaling trends continue, just raising budgets and model parameter counts will be unfeasible for obtaining solid mathematical reasoning. While Scale Transformers automatically solves most other text-based jobs, it does not currently solve MATH [١٣].

In [٢٠٢١] Understanding complex materials is considered an essential aim for industrial and scientific growth, especially ones with solid-liquid interfaces like water on surfaces or in small areas. Established modeling methods have given us the necessary information about atoms. Still, initio methods have trouble with the time and length scales we need, and force field methods can be wrong. The authors show how a simple and automatic machine-learning method can solve these problems and give accurate an initio interaction models for a wide range of complex aquatic systems. Because of these breakthroughs, molecular simulations of many scientifically critical systems are now possible. To understand complicated systems like how solids and liquids interact, you need simulation methods that show potential energy surfaces quickly and adequately. A machine learning method is given for building and testing models of complicated water systems. Instead of trying to make machine learning work well everywhere, the authors suggest making models that are easy to use and work well at specific thermal state points. After an initio simulation, a data-driven operational learning process builds machine learning capabilities. These models can then be used in full simulations to answer the science question or look at the thermal performance of initio methods. The authors show how their process works in different water systems, like bulk water with many ions in solution, water on a titanium dioxide surface, and water between nanotubes and molybdenum disulfide sheets. The accuracy of our method in terms of the initio reference is shown by an automated validation method that looks at structural and dynamical properties as well as the accuracy of force forecasts. Lastly, the authors show that method to examine the structure and movement of water on the surface of rutile titanium dioxide (١١٠). Machine learning models add more time and length scales to models of complicated systems accurately. Machine learning models make time and length scales for modeling complex systems simply but accurately [١٤].

In our proposed model, we introduce a challenge in measuring the Equation's runtime in different inputs using machine learning and without it based on Python.

٢,١ Gap in Related Work

Research has succeeded in its targets but persistent barriers exist for precise and effective runtime estimation of mathematical expressions that span multiple input values. Research studies continue to depend heavily on established computer methods that fail to produce accurate outcomes when dealing with complex inputs. Real-world data application testing of proposed models exists insufficiently which reduces their reliability output in practical usage. Applications requiring fast and accurate runtime prediction cannot use machine learning-based models because they need extensive training data. A new model needs development to create accurate mathematical equation runtime estimates in addition to improving integration between machine learning methods and traditional computational analytics.

٣. Methodology

Machine learning is frequently employed in various fields to resolve complex issues that are not amenable to simple computer-based solutions. One of the simplest and most commonly used machine learning methods is linear regression [١٥]. It is a mathematical method for conducting predictive analysis. Continuous, natural, or mathematical variable projections are possible with linear regression [١٦, ١٧]. A linear regression test assesses and quantifies the relationship between the variables under consideration. Regression and partial correlation are techniques that help scientists determine how confusion affects the connection between two variables [١٧,١٨]. It is a technique for evaluating data and modeling that develops linear relationships between dependent and independent variables. Thus, this technique would simulate the relationships between dependent and independent variables [١٩].

Figure ١ illustrates the proposed method. This study aims to investigate the viability and efficacy of machine learning for predicting equation runtimes to gain essential insights into improving performance analysis and decision-making in computational systems. Enhancing resource allocation, algorithm selection, and scalability analysis through the integration of machine learning techniques might result in computing systems that are more effective and optimized in the long run. The method examines the runtimes of equations with and without machine learning techniques to predict runtimes based on input sizes.

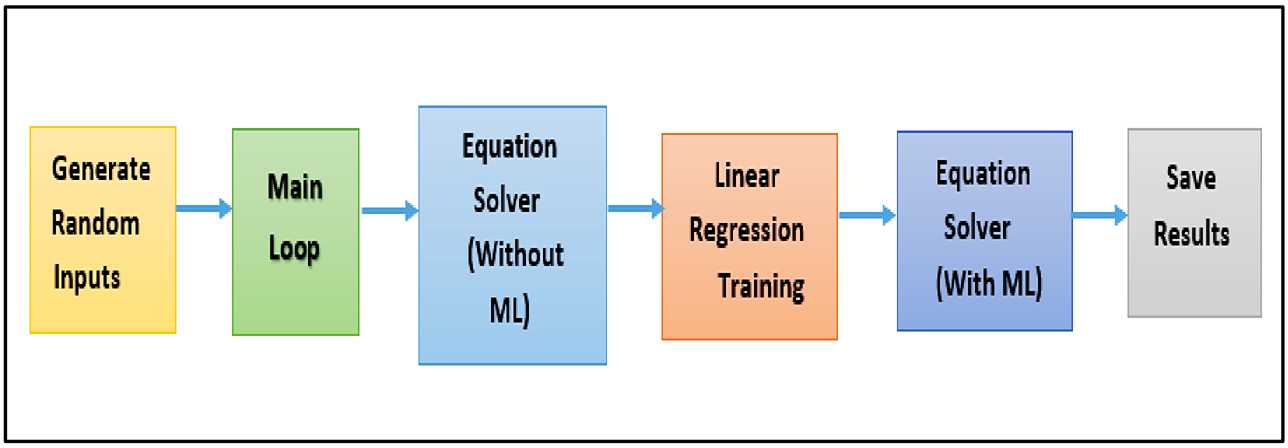
Generate Random Inputs
Main Loop
Equation Solver (Without ML)
Linear Regression Training
Equation Solver (With ML)
Save Results

Figure ١: General block diagram of the proposed model

The strategy takes an orderly approach to achieving this goal. To execute the linear regression model, the necessary libraries are imported. These include a time library to measure runtime, a library to generate random inputs, a pandas library to create a data frame for storing the results, and (sklearn.linear_model) for linear regression.

The critical equations are written as lambda functions in the list of equations. These equations show how to do a range of math operations and methods. Then, the process goes through each Equation one at a time. The runtime of each Equation is found by keeping track of when it starts, solving it with random inputs, and then keeping track of when it finishes. Without machine learning, the runtime of the Equation can be found by comparing the start and end times. This runtime has been printed so that it can be watched and studied.

The technique wrote down the Equation in the findings list, the Runtime without machine learning, and the expected Runtime with machine learning. This lets us compare and examine the two methods later on. The results list is then turned into a Pandas DataFrame so that more data can be viewed and played. The data frame is saved as an Excel file for later use and shared readily. The general steps for finding the runtime are:-

Table ١: Input output parameter

Step	Input Parameter	Description	Output Parameter	Description
Step ١	Required Libraries	time, numpy, pandas, LinearRegression from scikit-learn	Imported Libraries	Libraries available for use
Step ٢	x (array of values)	Randomly generated input values	Equation(x)	Computed equation output
Step ٣	np.random.rand(١٠٠٠٠٠٠)	Generates an array X of ١,٠٠٠,٠٠٠ random numbers between ٠ and ١	X	Random input values
Step ٤a	time.time()	Captures the start time	start_time	Recorded start time
Step ٤b	Equation(X)	Computes the output y for each X	y	Computed outputs of the equation
Step ٤c	time.time()	Captures the end time	end_time	Recorded end time
Step ٤d	end_time - start_time	Computes runtime	runtime	Execution time without ML
Step ٤e	runtime	Printed runtime without ML	Console Output	Displays runtime without ML
Step ٥a	X.reshape(-١,١)	Reshapes X into a column vector	X_reshaped	Reshaped input data
Step ٥b	time.time() - start_time for each X	Generates y values based on time differences	y_time_diff	Execution time differences
Step ٥c	X_reshaped, y_reshaped	Reshapes inputs for regression	X_reshaped, y_reshaped	Reshaped data
Step ٥d	LinearRegression().fit(X_reshaped, y_reshaped)	Fits regression model	reg	Trained linear regression model
Step ٦a	X_new = [[١]]	Defines a new input for prediction	X_new	New input for ML prediction
Step ٦b	reg.predict(X_new)	Predicts runtime for X_new	predicted_runtime	Predicted runtime with ML
Step ٧	predicted_runtime	Predicted runtime for input size of ١,٠٠٠,٠٠٠	Console Output	Displays predicted runtime with ML
Step ٨a	DataFrame creation	Stores results in a DataFrame	results	DataFrame with equation, runtime, and ML-predicted runtime

Step ٨b	results.to_excel('runtimes.xlsx')	Saves results in an Excel file	runtimes.xlsx	Output file containing execution times

٤. Results and analysis

This Section compares runtime measurements made without machine learning to runtime predictions made using a trained regression model. Regression analysis is a fundamental concept in the field of machine learning. It falls under supervised learning, wherein the algorithm is trained with both input features and output labels. It helps establish a relationship among the variables by estimating how one affects the other. Training a Regression Model involves finding the best possible values of the gradient (m) and y-intercept (c) to model a line for a given data set. This allows the model to predict the runtime of equation productivity (y). The predictive model's efficacy is evaluated by contrasting the anticipated runtimes with the actual runtimes achieved without machine learning. Statistical analyses and visuals are offered to show how well the models work.

١- One input size equation: Table ١ shows the runtime results **(with and without ML)** for different equations with one input size.

Table ٢ shows the results of runtime (with and without ML) for different equations with one input size

Equation	Runtime (without ML)	Predicted Runtime (with ML)
٢ * x^٣ + ٤ * x^٢ + ٦ * x + ٨	٢,٠٦٣٥٦٠٩٦٣	٢,١٩٣٣٢٠٧٠٢
lambda x: ٣ * x ** ٢ + ٥ * x + ٢	١,٦٥٦١٢٧٢١٤	١,٧٩٢١٧٠٩٦٩
lambda x: ٢ * x ** ٣ + ٤ * x ** ٢ + ٦ * x + ٨,	١٣,١٥٧٧٣٦٣	١٣,٢٩١٦٩٢٦٧
lambda x: np.sin(x) + np.cos(x)	٣,٥٤٠٩٦٦٢٧٢	٣,٦٨٩٢٥١٨٢٤
lambda x: np.exp(x) + np.log(x)	٣,٣٥٩١٢٣٤٦٨	٣,٤٩٢٧٨١٩٠٦
lambda x: ٤ * x + np.sqrt(x) - np.sinh(x)	٦,١٤٠٥٠٣٦٤٥	٦,٢٨٠٨٧٣٦٣٧

While figure ١ illustrates these results.

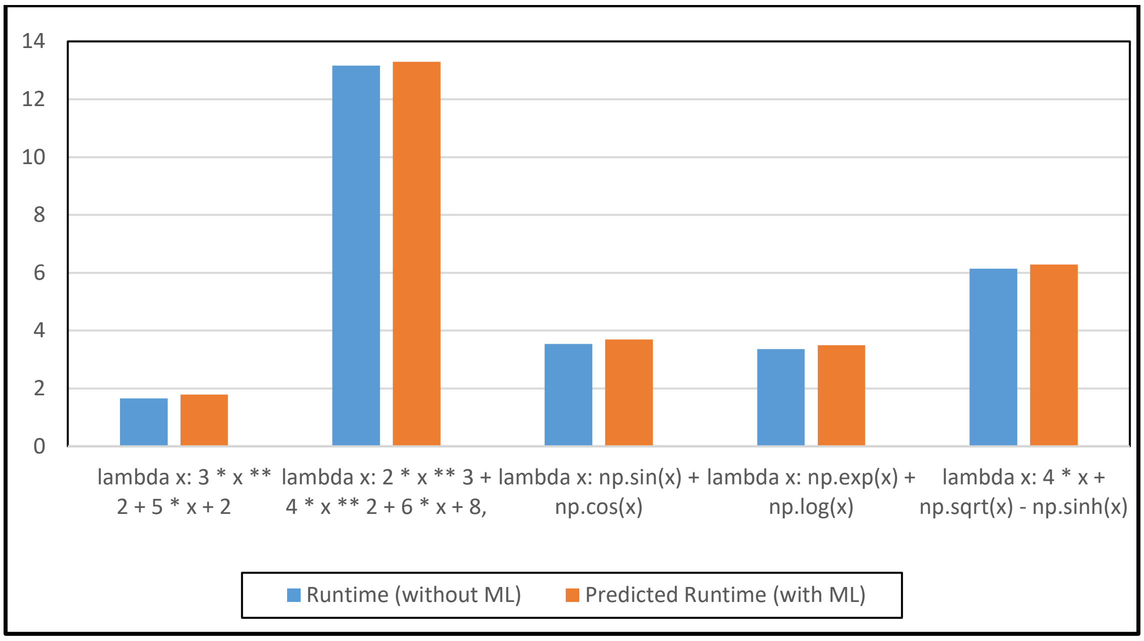

Figure ٢: One input size equation

٢- Multiple inputs size equation: Table ٢ shows the results of **runtime (with and without ML)** for different equations with considerable input size.

Table ٢ shows the runtime results (with and without ML) for different equations with multiple input sizes.

Equation	Runtime (without ML)	Predicted Runtime (with ML)
lambda x, y: ٢ * x ** ٢ + ٣ * y - ٤ * x * y + np.sin(x * y)	٥,٥٢٧٤٤٩	٥,٦٤٦١٩٨
lambda x, y, z: (x + y) * z ** ٢ + np.sqrt(x * y * z) - np.exp(x - y + z)	٦,٣٧٤٥٢	٦,٤٩٢٦١٧
lambda x, y, z: np.cos(x) * np.sinh(y) + np.log(z)	٦,٤٠٥٧٥٣	٦,٥٣١٨٤٥

While figure ٣ illustrates these results.

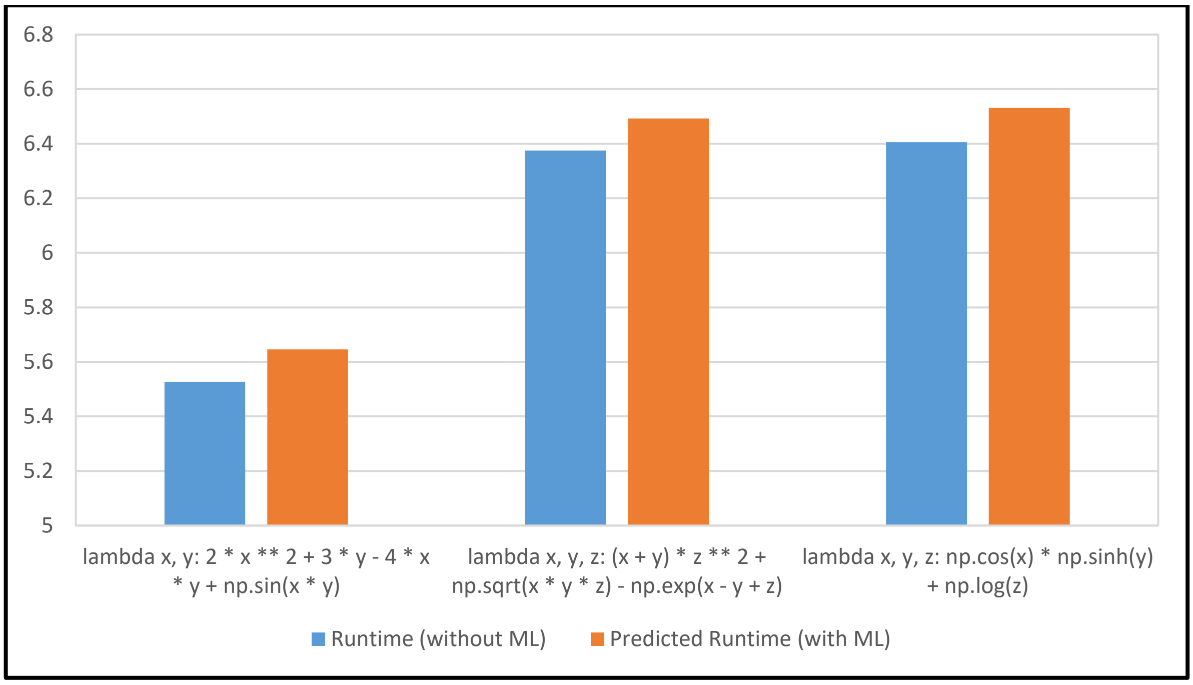

Figure ٣. Multiple inputs size equation

As well as table ٣ shows the runtime results **(with and without ML)** for differences in other equations and figure ٣ illustrates these results.

Table ٣. Results of runtime (with and without ML) for other different equations

Equation	**Runtime (without ML)**	**Predicted Runtime (with ML)**
lambda x, y: ٢ * x ** ٢ + ٣ * y - ٤ * x * y + np.sin(x * y)	٦,٥٩٤٧٠٧	٦,٧٣٦٨٨٩
lambda x, y, z: (x + y) * z ** ٢ + np.sqrt(x * y * z) - np.exp(x - y + z)	٦,٠٩٥٥١٧	٦,٢٣٣٨٣٧
lambda x, y, z: np.cos(x) * np.sinh(y) + np.log(z)	٦,١١٨٠٢١	٦,٢٦١٠٩٥

Figure ٣ shows the runtime results (with and without ML) for the equations in Table ٣. As we can see, the model can predict the runtime with the Equation compared to the runtime without ML.

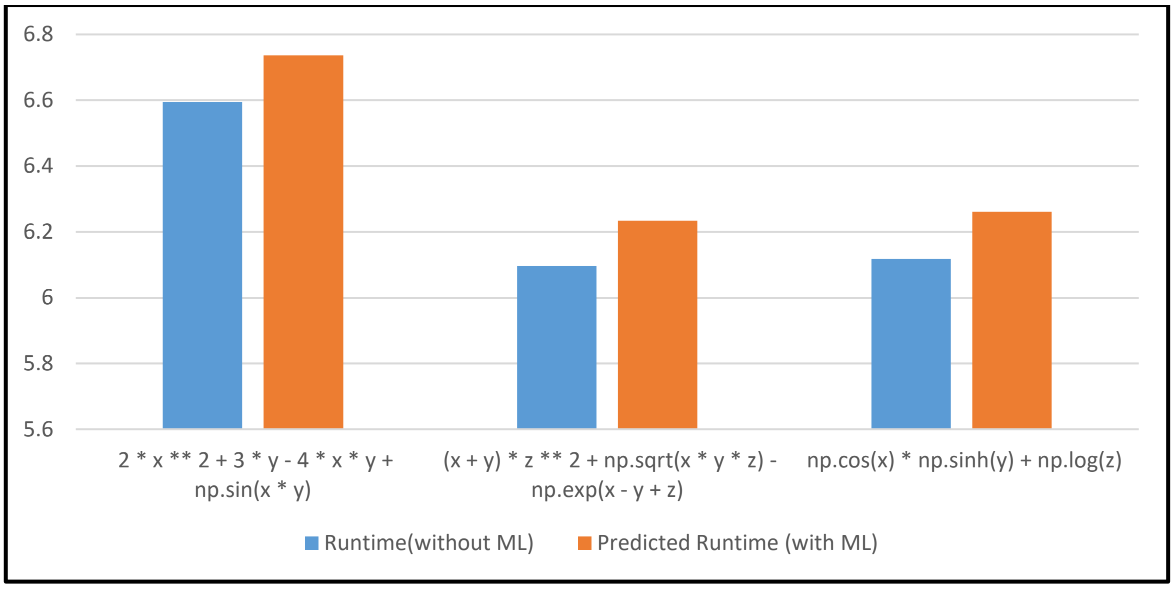

Figure ٣. The runtime results (with and without ML) for differences in other equations.

٥. Conclusion and Future Work

The study shows the necessity to enhance mathematical equation runtime estimation accuracy through machine learning techniques development. Research into previous studies showed two main issues with traditional methods and small-scale dataset testing limitations. Building an improved efficient model represents an essential need to enhance runtime estimation while improving machine learning integration with traditional computational analysis.

Future research should investigate high-level artificial intelligence approaches including deep neural networks and reinforcement learning models to achieve more precise estimates. The proposed model requires testing on multiple datasets to prove its capability of working across various usage scenarios. Real-time prediction processes become faster through combined use of cloud computing and parallel computing with performance optimization techniques.

Acknowledgment

The authors would like to thank Mustansiriyah University (https://uomustansiriyah.edu.iq) in Baghdad, Iraq, for its support in the present work.

References

[١] She, C., Sun, C., Gu, Z., Li, Y., Yang, C., Poor, H. V., & Vucetic, B. (٢٠٢١). A tutorial on ultrareliable and low-latency communications in ٦G: Integrating domain knowledge into deep learning. *Proceedings of the IEEE*, *109*(٣), ٢٠٤-٢٤٦.

[٢] Thirunavukkarasu, M., Sawle, Y., & Lala, H. (٢٠٢٣). A comprehensive review of optimization of hybrid renewable energy systems using various optimization techniques. *Renewable and Sustainable Energy Reviews*, *176*, ١١٣١٩٢

[٣] S.. Ayad and I. T. Abbas, "Using Evolving Algorithm with Distance Indicator for Solving Different Non-linear Optimization Problems," *Al-Mustansiriyah Journal of Science*, vol. ٣٣, no. ٣, pp. ٦٦–٧٣, Sep. ٢٠٢٢.

[٤] Regazzoni, F., Dede, L., & Quarteroni, A. (٢٠١٩). Machine learning for fast and reliable solutions of time-dependent differential equations. *Journal of Computational Physics*, *397*, ١٠٨٨٥٢.

[٥] Innes, M., Edelman, A., Fischer, K., Rackauckas, C., Saba, E., Shah, V. B., & Tebbutt, W. (٢٠١٩). A differentiable programming system to bridge machine learning and scientific computing. *arXiv preprint arXiv:1907.07587.*

[٦] Saba Abdulbaqi Salman, Sufyan Al-Janabi, and Ali Makki Sagheer, "Security Attacks on E-Voting System Using Blockchain," Iraqi Journal For Computer Science and Mathematics, vol. ٤, no. ٢, pp. ١٧٩–١٨٨, May ٢٠٢٣.

[٧] Heinlein, A., Klawonn, A., Lanser, M., & Weber, J. (٢٠٢١). Combining machine learning and domain decomposition methods to solve partial differential equations—A review. *GAMM-Mitteilungen*, *44*(١), e٢٠٢١٠٠٠٠١.

[٨] Nabian, M. A., & Meidani, H. (٢٠١٩). A deep learning solution approach for high-dimensional random differential equations. *Probabilistic Engineering Mechanics*, *57*, ١٤-٢٥.

[٩]N. N. Hasan and Z. John, "Analytic Approach for Solving System of Fractional Differential Equations," *Al-Mustansiriyah Journal of Science*, vol. ٣٢, no. ١, pp. ١٤–١٧, Feb. ٢٠٢١.

[١٠] S. Ray, "A Quick Review of Machine Learning Algorithms," *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, Faridabad, India, ٢٠١٩, pp. ٣٥-٣٩, doi: ١٠,١١٠٩/COMITCon.٢٠١٩,٨٨٦٢٤٥١.

[١١] M. H. Mahmood, "On α-Fuzzy Soft Irreducible Spaces," *Al-Mustansiriyah Journal of Science*, vol. ٣٤, no. ١, pp. ٦٥–٧٠, Mar. ٢٠٢٣.

[١٢]Westphal, F., Grahn, H., & Lavesson, N. (٢٠١٨). Efficient document image binarization using heterogeneous computing and parameter tuning. *International Journal on Document Analysis and Recognition (IJDAR).* https://doi.org/١٠,١٠٠٧/s١٠٠٣٢-٠١٧-٠٢٩٣-٧.

[١٣] Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., ... & Steinhardt, J. (٢٠٢١). Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

[١٤] Schran, C., Thiemann, F., Rowe, P., Müller, E., Marsalek, O., & Michaelides, A. (٢٠٢١). Machine learning potentials for complex aqueous systems are made simple. Proceedings of the National Academy of Sciences of the United States of America. https://doi.org/١٠,١٠٧٣/pnas.٢١١١٠٠٧٧١١٨.

[١٥] Salman, S. A., Mohialden, Y. M., & Hussien, N. M. (٢٠٢٤). A Generating Distorted CAPTCHA Images Using a Machine Learning Algorithm. Iraqi Journal for Computer Science and Mathematics, ٥(٣), ٢٧.

[١٦] Zhou, Z. H. (٢٠٢١). *Machine learning*. Springer Nature

[١٧] E. A. W. Hachim, M. T. Gaata and T. Abbas. (٢٠٢٢). Iris-based Authentication Model in Cloud Environment (IAMCE). International Conference on Electrical, Computer and Energy Technologies (ICECET), Prague, Czech Republic, pp. ١-٦, doi: ١٠,١١٠٩/ICECET٥٥٥٢٧,٢٠٢٢,٩٨٧٣٤٩٩.

[١٧] Su, X., Yan, X., & Tsai, C. L. (٢٠١٢). Linear regression. *Wiley Interdisciplinary Reviews: Computational Statistics*, *4*(٣), ٢٧٥-٢٩٤.

[١٨] Ethar Abdul Wahhab Hachim, Methaq Talib Gaata a, Thekra Abbas. (٢٠٢٣). Voice-Authentication Model Based on Deep Learning for Cloud Environment. International Journal on Informatics Visualization. https://joiv.org/index.php/joiv/article/view/١٣٠٣.

[١٩] Salman, S. A., Mohialden, Y. M., & Hussien, N. M. (٢٠٢٤). A Generating Distorted CAPTCHA Images Using a Machine Learning Algorithm. Iraqi Journal for Computer Science and Mathematics, ٥(٣), ٢٧.