Regular Manuscript

Augmented Data Low Confidence (ADLC): A Confidence-Driven Data Augmentation Framework with Ensemble Optimization for Enhanced Machine Learning Performance

Submission ID 8e8d7940-e0e3-433c-ba30-7f9d5f55b95d

Submission Version Initial Submission

PDF Generation 19 Sep 2025 10:17:50 EST by Atypon ReX

Authors

Dr. Dwi Sunaryono
Corresponding Author

Affiliations
- Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia

Prof. Riyanarto Sarno

Affiliations
- Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia

Mrs. Shoffi Izza Sabilla
Submitting Author

ORCiD
https://orcid.org/0000-0003-0060-2143

Affiliations
- Department of Medical Technology, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia

Miss Rizqy Ahsana Putri

Affiliations
- Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia

Mr. Taufiq Choirul Amri

Affiliations
- Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia

Mr. Irfan Mirda

ORCiD
https://orcid.org/0009-0003-1499-1992

Affiliations

- Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia

Dr. Joko Siswantoro

Affiliations

- Department of Informatics Engineering, University of Surabaya, Surabaya, 60293, Indonesia

Additional Information

Subject Category

Communications technology

Computational and artificial intelligence

Computers and information processing

Keywords

Data-driven modeling

Machine learning

Machine learning algorithms

Performance analysis

Select a Manuscript Type

Research Article

Files for peer review

All files submitted by the author for peer review are listed below. Files that could not be converted to PDF are indicated; reviewers are able to access them online.

Name	Type of File	Size	Page
Clean FinalManuscript_adlc ieee paper .pdf	Main Document - PDF	1.0 MB	Page 4
FinalManuscript_adlc ieee paper 2.docx	Author's Tracked Changes (Highlighted Manuscript of Changes)	815.4 KB	Page 26
0. Authors Response Files.pdf	Author Response	320.5 KB	Page 48

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

"Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000."

"Digital Object Identifier 10.1109/ACCESS.2024.Doi Number"

Augmented Data Low Confidence (ADLC): A Confidence-Driven Data Augmentation Framework with Ensemble Optimization for Enhanced Machine Learning Performance

Dwi Sunaryono[1], Riyanarto Sarno[1], Shoffi Izza Sabilla[2], Rizqy Ahsana Putri[1], Taufiq Choirul Amri[1], Irfan Mirda[1], Joko Siswantoro[3]

[1]Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia
[2]Department of Medical Technology, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia
[3]Department of Informatics Engineering, University of Surabaya, Surabaya, 60293, Indonesia

Corresponding author: Dwi Sunaryono (dwi@if.its.ac.id)

ABSTRACT The application of machine learning in data-driven solutions has matured, yet efforts continue to improve predictive accuracy. This study presents a comprehensive approach that begins with data preprocessing, including the removal of invalid values, duplicate entries, feature selection, and dimensionality reduction, followed by model optimization through hyperparameter tuning. A novel method, Augmented-Data Low Confidence, is introduced to enhance model performance by augmenting samples with low prediction confidence. The k-nearest neighbors method is used to estimate prediction probabilities. Samples falling below a defined confidence threshold are selected for augmentation by generating new data points. These points are created by randomly sampling feature values within the upper and lower bounds of the low-confidence instances. The augmented dataset is then optimized using the Gray Wolf Optimization algorithm, which adjusts model parameters based on an accuracy-driven fitness function. Experiments on ten public datasets and two proprietary datasets show that this feedback-based augmentation consistently improves the accuracy of various machine learning models. The results demonstrate the effectiveness of incorporating uncertain predictions into the learning process, leading to improved generalization and classification performance.

INDEX TERMS Augmentation, Machine Learning, Performance Improvement

I. INTRODUCTION

Machine learning (ML) has become an essential tool in many fields including healthcare, finance, and transportation because of its ability to analyze large volumes of data and derive meaningful insights [1]. Classification and regression tasks frequently employ ML models such as k-nearest neighbors (kNN), support vector machines (SVM), and decision trees, with their performance heavily influenced by data preprocessing, feature selection, and optimization strategies [2]. Despite continuous advancements in ML algorithms, persistent challenges, such as class imbalance, overfitting, and suboptimal hyperparameter tuning, continue to hinder model performance [3].

Class imbalance is a well-documented issue in classification tasks, where certain classes contain significantly fewer instances than others, often resulting in biased predictions toward the majority class [4]. To address this challenge, various oversampling methods have been proposed, including the Synthetic Minority Oversampling Technique (SMOTE), Improved Majority Weighted Minority Oversampling Technique (IMWMOTE), and Adaptive Clustering Weighted Oversampling (ACWOS) [5]. These approaches enhance classification performance

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

by generating synthetic samples for the minority class. However, they can introduce substantial computational overhead and may produce unrealistic data points, increasing the risk of overfitting. IMWMOTE has demonstrated greater effectiveness than traditional SMOTE by incorporating adaptive noise handling and clustering techniques, making it more suitable for heterogeneous and imbalanced datasets. Nonetheless, its sensitivity to noise and high computational complexity remain significant limitations [4].

Overfitting arises when an ML model captures excessively complex patterns specific to the training data, thereby impairing its ability to generalize to unseen data [6]. To address this challenge, particularly in scenarios involving limited or imbalanced datasets, data augmentation is widely utilized. Techniques such as Gaussian noise injections, SMOTE-ENN (a combination of the Synthetic Minority Oversampling Technique and Edited Nearest Neighbor), and generative models like generative adversarial networks (GANs) are employed to increase the diversity of training samples [7]. However, these methods require careful parameter optimization, as poorly generated synthetic data may be redundant or introduce noise, ultimately compromising model performance.

Hyperparameter tuning is crucial in deciding the performance of ML models. Conventional optimization techniques, such as grid search and random search, are generally inefficient and have a large computing overhead, especially when dealing with high-dimensional parameter fields. To address these issues, researchers have increasingly relied on metaheuristic optimization methods such as Particle Swarm Optimization (PSO), Genetic methods (GA), and Grey Wolf Optimization (GWO) for hyperparameter selection [8]. Among these approaches, Enhanced Grey Wolf Optimization (E-GWO) has shown significant gains in optimizing deep learning architectures, particularly convolutional neural networks (CNNs), delivering higher classification accuracy while requiring less computational resources than PSO and GA [6].

GWO is based on the social structure and cooperative hunting behaviour of grey wolves in the wild. It successfully balances exploration and exploitation in the optimization process, allowing for a more thorough search of the solution space. Unlike traditional methods, GWO dynamically updates the positions of candidate solutions by referencing the most promising individuals, thereby enhancing its ability to escape local optima [9]. This property is particularly beneficial for tuning ML models that involve extensive and complex hyperparameter configurations. In CNN-based image classification tasks, GWO has been shown to reduce classification error rates by up to 39.2 percent compared to PSO and by 15 percent relative to GA, highlighting its efficacy in hyperparameter optimization for deep learning applications [8].

Table I summarizes several previous studies that have applied various approaches to optimize machine learning models, most of which have achieved notable advancements. Despite these advancements, existing methods still face limitations in adaptively managing data augmentation and hyperparameter optimization. To address these challenges, this study proposes Augmented-Data Low Confidence (ADLC), a novel approach that integrates probability-based data augmentation with GWO-based hyperparameter tuning to dynamically improve ML model accuracy. Unlike conventional augmentation techniques that generate synthetic samples indiscriminately, ADLC selectively identifies low-confidence predictions and augments only the most informative instances. This targeted strategy ensures that augmented data contributes meaningfully to the learning process rather than introducing redundancy. Furthermore, the augmentation ratio is adjusted in real-time based on model performance, preventing unnecessary oversampling and reducing the risk of overfitting.

The integration of GWO within the ADLC framework further enhances its effectiveness by providing an adaptive and efficient search mechanism for hyperparameter tuning. This approach reduces computational costs while improving model generalization. Compared to traditional optimization techniques, GWO offers more stable convergence and a superior balance between exploration and exploitation, enabling efficient identification of optimal hyperparameter configurations. Prior studies have shown that GWO can outperform grid search, random search, and even other evolutionary algorithms such as PSO in optimizing ML models [10].

In this study, the ADLC framework is evaluated across multiple datasets, including both publicly available benchmarks and domain-specific research datasets, to assess its effectiveness in addressing class imbalance, overfitting, and inefficient hyperparameter tuning. The experimental results demonstrate that ADLC consistently improves model accuracy and generalization performance when compared to conventional data augmentation and optimization techniques.

Section II presents the materials and methods used in this study, including dataset descriptions, preprocessing steps, and a detailed explanation of the ADLC framework, which integrates probability-based augmentation with GWO. The implementation of dynamic augmentation and GWO-based hyperparameter tuning is also discussed. Section III presents the experimental results and comparative analysis between ADLC and existing approaches, highlighting its impact on model performance. Finally, Section IV summarizes the key findings, emphasizes ADLC's contributions to enhancing ML efficiency, and outlines potential directions for future research. By systematically integrating data augmentation

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

and optimization, ADLC offers a robust, scalable, and effective solution for addressing key challenges in ML.

II. MATERIALS AND METHODS

A. MATERIALS

This study employs a total of twelve datasets, consisting of ten numerical datasets, one image dataset, and one electroencephalogram (EEG) signal dataset. The numerical datasets include Iris, Wine, Glass, Car Evaluation, Balance Scale, Heart Disease, Diabetes, Breast Cancer, Pancreatic Ductal Adenocarcinoma (PDAC), and chronic kidney disease (CKD). The image dataset utilized is Digits, while the signal dataset is EEG Bonn.

Most of the numerical datasets were obtained from the UCI Machine Learning Repository. The Iris dataset comprises 150 samples characterized by four attributes, each representing a specific iris flower species. The Wine dataset includes 178 samples with thirteen chemical attributes corresponding to different wine types. The Glass dataset consists of 214 instances with nine features used to classify glass types. The Car Evaluation dataset includes 1,728 entries and six attributes associated with car acceptability. The Balance Scale dataset contains 625 samples with four features related to the physical balance of a scale. The Heart Disease dataset comprises 303 records and thirteen attributes designed for predicting the risk of heart disease.

The Diabetes dataset, sourced from Mendeley Data, includes 1,000 instances with fourteen attributes, reflecting patient data collected from Medical City Hospital and the Specialized Center for Endocrinology and Diabetes at Al-Kindy Teaching Hospital. The Breast Cancer dataset, retrieved from the Kaggle platform, contains data from 569 patients with thirty-two features, including thirty-one numerical attributes and one classification label [11]. The PDAC dataset, titled "Urinary Biomarkers for Pancreatic Cancer" and available on Kaggle through a contribution by John Davis, includes 590 entries with thirteen input features and one target label [12]. The CKD data set, also sourced from Kaggle, consists of 400 records characterized by twenty-four input attributes and one classification label [13].

The Digits dataset comprises 1,797 grayscale images of handwritten digits, classified into ten distinct categories. The EEG Bonn dataset includes 500 instances with eight features used for EEG signal analysis. A summary of all datasets employed in this study is provided in Table II. In the table, numerical datasets are marked in yellow, image datasets in blue, and signal or EEG datasets in green. The inclusion of heterogeneous datasets aims to comprehensively assess the proposed model's performance across a wide range of real-world scenarios. This strategy enhances the experimental robustness and affirms the model's adaptability across different data modalities. In addition to ensuring reproducibility, the complete source code used for data processing, model training, and metaheuristic algorithm implementation is made publicly available in a GitHub repository (https://github.com/Irfanmrd12/ADLC).

B. AUGMENTATION

Data augmentation is a critical technique in ML, aimed at increasing the volume, quality, and diversity of training data. By generating new data points through various transformations or by incorporating information from multiple sources, data augmentation enriches the original dataset [27]. This process not only expands the dataset size, an essential factor for training robust models, but also introduces variability that enhances model generalization, particularly in cases where the original data is limited or imbalanced [28], [29]. Furthermore, data augmentation improves the informativeness of training samples, making them more valuable for the learning process [27].

TABLE I
PREVIOUS RELATED STUDIES

Author	Title	ML/Processing Method	Optimization Method	Evaluation Results
Jiaxin Wang et al. [4]	"IMWMOTE: A novel oversampling technique for fault diagnosis in heterogeneous imbalanced data"	AHC Clustering, k-NN denoising, ICEEMDAN	IMWMOTE, adaptive noise handling	Sensitivity 100%, G-mean 94.33%, F-measure 94.66%, AUC 95.83%
Sai Li et al. [5]	"Rolling Bearing Fault Diagnosis Under Data Imbalance and Variable Speed Based on Adaptive Clustering Weighted Oversampling"	NAP, Density Peak Clustering, ACWOS	Soft thresholding, weighted oversampling	Accuracy up to 99%
Priyanka et al. [6]	"Enhanced Grey Wolf Optimization based Hyper-parameter optimized Convolution Neural Network for Kidney Image Classification"	CNN	Enhanced Grey Wolf Optimization (E-GWO)	Accuracy 97.01%
Rasmiranjan Mohakud & Rajashree Dash [8]	"Designing a Grey Wolf Optimization-based Hyper-parameter Optimized CNN Classifier for Skin Cancer Detection"	CNN, ISIC dataset	Grey Wolf Optimization (GWO)	Accuracy 98.33%, Error reduction 39.2% (PSO), 15% (GA)

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Author	Title	ML/Processing Method	Optimization Method	Evaluation Results
Yanlu Gong et al. [14]	"A Diversity and Reliability-Enhanced Synthetic Minority Oversampling Technique for Multi-Label Learning"	MLL (BR, CC, MLKNN, ML-RBF)	DR-SMOTE	Performance improvement over MLSMOTE, MLROS, etc.
Junnan Li et al. [15]	"A novel oversampling technique for class-imbalanced learning based on SMOTE and natural neighbours"	3NN, SVM, SMOTE	NaNSMOTE	Improved accuracy and balance over classic SMOTE
Efstathios D. Gennatas et al. [16]	"Expert-Augmented Machine Learning"	RuleFit, Random Forest	Expert-augmented ML (EADLC)	Balanced accuracy increased from 67.3% to 74.4%
Emre Avuçlu [17]	"A new data augmentation method to use in machine learning algorithms using statistical measurements"	k-NN, Decision Tree, Naïve Bayes, Random Forest, SVM	Statistical-based data augmentation	Accuracy improved up to 87.76% (DT, RF, NB)
Valdemar Švábenský et al. [7]	"Evaluating the Impact of Data Augmentation on Predictive Model Performance"	Logistic Regression, SVM, Random Forest, MLP	SMOTE-ENN, Noise Addition	AUC increased by 0.01 over baseline
Muhammad Mujahid et al. [18]	"Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering"	RF, SVM, KNN, AdaBoost, CNN, LSTM, BiLSTM	SMOTE, ADASYN, K-Means SMOTE	SVM + ADASYN accuracy 99.67%, recall 100%
Chinmay Chakraborty et al. [9]	"Novel Enhanced-Grey Wolf Optimization hybrid machine learning technique for biomedical data computation"	Hybrid ML (Bagging & Boosting)	Enhanced Grey Wolf Optimization (E-GWO)	Accuracy 99.26%, Accuracy improvement 11.90%
Suraj Kumar Parhi, Sanjaya Kumar Patro [10]	"Prediction of Compressive Strength of Geopolymer Concrete Using a Hybrid Ensemble of Grey Wolf Optimized Machine Learning Estimators"	Hybrid Ensemble ML (HEML)	Grey Wolf Optimization (GWO)	R^2 Training: 0.97, Testing: 0.94, MAE: 1.34, RMSE: 2.5
Feng Shen et al. [19]	"A New Deep Learning Ensemble Credit Risk Evaluation Model with an Improved Synthetic Minority Oversampling Technique"	LSTM, AdaBoost	Improved SMOTE, Mahalanobis Distance	AUC: 80.32% (German), 74.90% (Taiwan)
Azal Ahmad Khan et al. [20]	"A Review of Ensemble Learning and Data Augmentation Models for Class Imbalanced Problems"	SMOTE, ADASYN, GANs, LightGBM, XGBoost	SMOTE + LightGBM	SMOTE + LightGBM showed best results
Teppei Suzuki [21]	"TeachAugment: Data Augmentation Optimization Using Teacher Knowledge"	Neural Network	Adversarial Data Augmentation, SGD	CIFAR-10: 2.5% error rate, CIFAR-100: 16.8%, ImageNet: 22.2%
Francisco J. Moreno-Barea et al. [22]	"Improving Classification Accuracy Using Data Augmentation on Small Data Sets"	VAE, GANs	Balanced Multiclass, WGAN-GP	Accuracy increased up to 24% on small datasets
Manoharan Premkumar et al. [23]	"Augmented weighted K-means grey wolf optimizer: An enhanced metaheuristic algorithm for data clustering problems"	K-means clustering	Grey Wolf Optimization (GWO)	Better performance than standard GWO
Min Zhang et al. [24]	"A New Approach for Evaluating Node Importance in Complex Networks via Deep Learning Methods"	Graph Neural Network (GNN), CNN	Adam Optimizer	Kendall's τ improved over baseline
Misgana Negassi et al. [25]	"Smart (Sampling)Augment: Optimal and Efficient Data Augmentation for Semantic Segmentation"	Bayesian Optimization, Smart Sampling Augment	SmartAugment	SmartAugment outperformed all previous methods
Hartono, Erianto Ongko [26]	"Avoiding Overfitting and Overlapping in Handling Class Imbalanced Using Hybrid Approach"	Smoothed Bootstrap Resampling, SMOTE	Feature Selection (FAST)	Lower Balanced Error Rate (BER), better than Wrapper Approach-SMOTE

C. K-NEAREST NEIGHBORS (K-NN)

The k-nearest neighbors (kNN) algorithm is a widely used supervised ML technique, known for its simplicity and effectiveness. Unlike traditional models that require training to build predictive functions, kNN adopts an instance-based or lazy learning approach, where the entire training dataset is stored and directly referenced during classification. Each data instance is treated as a point in an n-dimensional feature space. To classify a new sample, the algorithm identifies its k nearest neighbors using a selected distance metric, such as Euclidean or Manhattan distance, and assigns the most frequent class label among them. The choice of k is crucial: a small k may lead to overfitting due to sensitivity to noise, whereas a large k may cause underfitting by smoothing out meaningful distinctions [30], [31].

The Euclidean distance, which measures the straight-line distance between two points in n-dimensional space, is defined in Equation (1).

$$d(y, x_i^j) = \sqrt{(y - x_i^j)^T (y - x_j^i)} \quad (1)$$

where x_i^j represents the i-th training sample from class j.

Alternatively, the Manhattan distance, defined in Equation (2), calculates the sum of absolute differences between corresponding features.

$$d(y, x_i^j) = \sum_{k=1}^{n} |y_k - x_{ik}^j| \quad (2)$$

where x_i^j is the i-th training sample from class j while y_k and x_{ik}^j are their respective feature values.

Recent advancements have focused on improving kNN's robustness, particularly in reducing its sensitivity to the choice of k [32]. For example, Gou et al. proposed weighted representation-based and weighted local mean representation-based kNN variants, which adjust the influence of neighbouring points in classification, thereby enhancing reliability. Additionally, they introduced a generalized mean distance-based method that further refines the selection of neighbourhood size [33].

In this study, kNN is employed not only for classification but also for filtering samples based on confidence thresholds. Only instances that fall below a specified confidence level are selected for augmentation. By clearly separating test and training data, this process ensures a structured and leakage-free augmentation strategy. The use of kNN for confidence-based filtering highlights its utility beyond classification, contributing to both data preprocessing and model optimization.

D. PREDICTION PROBABILISTIC

In ML, probabilistic predictions refer to outputs that assign a probability distribution over potential outcomes, providing not only a classification but also a measure of the model's certainty. The confidence level is a central concept within this framework, indicating how strongly the model believes in each prediction. A higher confidence value reflects greater certainty, while a lower value suggests uncertainty. The threshold defines the minimum acceptable confidence level for a prediction to be considered valid. By adjusting this threshold, practitioners can control the model's behavior, whether prioritizing precision by accepting only high-confidence predictions or favoring broader coverage by including lower-confidence ones, depending on the specific task requirements [34], [35].

This study explores how varying the threshold influences the selection of low-confidence samples for augmentation. Specifically, the threshold determines which predictions are deemed uncertain and subsequently selected for synthetic expansion. Adjusting this parameter enables a flexible trade-off between different performance metrics and affects the flow and quality of augmented data. The relationship between predicted probabilities and empirical accuracy is formalized in Equation (3), which provides the basis for determining confidence levels in a quantifiable manner.

$$p(c_j \mid p^{c_j}) = p^{c_j} \quad (3)$$

where p^{c_j} is the probability estimate for class j. The (average) probability estimate for the predicted label should therefore match the empirical accuracy if it is examined empirically [36].

TABLE II
SUMMARY OF DATASETS USED

Datasets	Number of Instance	Number of Features	Imbalance Ratio
Iris	150	4	0.00
Wine	178	13	1.48
Glass	214	9	8.44
Car Evaluation	1728	6	20.61
Balance Scale	625	4	5.88
Heart Disease	303	13	1.18
Diabetes	1000	14	0.06
Breast Cancer	569	32	0.59
Pancreatic Ductal Adenocarcinoma (PDAC)	590	14	0.88
Chronic Kidney Disease (CKD)	400	25	0.60
Digits	1797	64	1.05
EEG Bonn	500	8	1.00

E. OPTIMIZATION ALGORITHM PSO_GWO_DE_AP_WOA_ENSEMBLE

The PSO_GWO_DE_AP_WOA Ensemble is a hybrid optimization framework that integrates four distinct metaheuristic algorithms: Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO), Whale Optimization Algorithm (WOA), and Differential Evolution (DE). This integrated approach is designed to improve model accuracy and stability by harnessing the complementary strengths of each algorithm in both the exploration and exploitation phases of the search process. In the context of this study, the ensemble is employed to optimize data augmentation strategies, allowing for the dynamic selection and adjustment of augmentation techniques based on the outcomes of the optimization process. This adaptive

mechanism contributes to the generation of a more diverse and representative training dataset, thereby enhancing the overall performance of the model.

1) PARTICLE SWARM OPTIMIZATION (PSO)

The classical Particle Swarm Optimization (PSO) algorithm is inspired by the collective behavior observed in bird flocks during the search for food. In this algorithm, each particle represents a potential solution and adjusts its position within the solution space by referencing three key elements: its current position, its historically best-known position, and the globally best position identified by the swarm [37], [38]. In the present study, PSO is employed to identify the optimal combination of data augmentation techniques that maximize model accuracy. The algorithm is integrated into the ensemble framework and is specifically applied to candidate samples that exhibit predicted probabilities below the threshold of 0.33. The mechanism for updating a particle's position in PSO is described by Equation (4), which models both the velocity and positional changes of each particle:

$$\alpha = w\ x\ \alpha + d_1 x\ e_1\ x\ (\mu - p) + d_2 x\ e_2\ x\ (\beta - p) \quad (4)$$

In this formulation, w denotes the inertia weight, which regulates the influence of the particle's previous velocity on its current motion. The parameter α represents the particle's velocity, determining the rate at which the particle navigates the solution space. The coefficient d_1 reflects the cognitive component, directing the particle toward its personal best position denoted as μ. This component encourages individualized learning based on past success. Conversely, d_2 serves as the social component, guiding the particle toward the global best position β, thereby promoting collective intelligence. The variables e_1 and e_2 are uniformly distributed random numbers between 0 and 1, incorporated to introduce stochastic variability and reduce the likelihood of convergence to local optima. Finally, p denotes the current position of the particle within the solution space. This formulation enables the algorithm to effectively balance exploration and exploitation in the search for optimal solutions.

2) GREY WOLF OPTIMIZATION (GWO)

Grey Wolf Optimization (GWO) is a population-based metaheuristic algorithm that emulates the leadership structure and collaborative hunting strategies observed in grey wolf packs. Within the GWO framework, candidate solutions are ranked according to their fitness values. The most optimal solution is designated as alpha (α), followed by beta (β) for the second-best, delta (δ) for the third best, and the remaining solutions are referred to as omega (ω). The positions of α, β, and δ serve as reference points that guide the movement of the ω wolves, facilitating convergence toward superior solutions by mimicking the wolves' natural encircling and hunting behaviors [39], [40].

In this study, GWO is utilized to optimize the selection of data augmentation techniques, particularly for samples whose predicted probabilities fall within the range of 0.33 to 0.66. This targeted application allows for adaptive augmentation that improves model generalization. The ensemble framework incorporates GWO to strengthen the robustness and consistency of the decision-making process. The mathematical formulation governing the update of candidate positions in GWO is presented in Equations (5) and (6), which define the algorithm's dynamic adjustment process in navigating the search space.

$$y_i = \frac{Y_a + Y_b + Y_x}{3} \quad (5)$$

$$D_a = |C_1.Y_a - y_i|, D_b = |C_1.Y_b - y_i|, D_x = |C_1.Y_x - y_i| \quad (6)$$

where y_i represents the current position of the wolf (candidate solution) in the solution space. Y_a is the position of the alpha wolf, which corresponds to the best solution found so far, while Y_b dan Y_x represent the positions of the beta and delta wolves, which correspond to the second and third best solutions, respectively. C_1, C_2, dan C_3 are random coefficients (between 0 and 1) that control the influence of alpha, beta, and delta wolves on the movement of the current solution. D_a, D_b, dan D_d are the distances between the current wolf position and the alpha, beta, and delta wolf positions, each scaled by the corresponding random coefficients (C_1, C_2, dan C_3) which guide the current solution to a better position in the search space.

3) DIFFERENTIAL EVOLUTION (DE)

Differential Evolution (DE) is an evolutionary optimization algorithm that enhances candidate solutions through the recombination of information derived from multiple individuals within a population. The algorithm operates through four fundamental stages: initialization, mutation, crossover, and selection. In the mutation phase, DE generates mutant vectors by adding the scaled difference between two randomly selected individuals to a third individual. These mutant vectors are subsequently combined with existing solutions through the crossover process to produce trial vectors [41], [42]. The trial vectors are then evaluated based on their fitness, and the superior candidates are retained for the next generation.

In the present study, DE is employed to identify the most effective data augmentation strategies for samples with prediction probabilities ranging from 0.66 to 0.80. This approach enables fine-grained control over augmentation in moderately uncertain regions of the classification space. The mathematical procedures underlying the DE operations are presented in Equations (7) through (9), which formally define the mutation, crossover, and selection mechanisms used in the optimization process.

$$v = k_1 + F.(k_2 - k_3) \quad (7)$$

$$u_j = \{v_j, if\ r_j < CR\ and\ k_{ij}, otherwise\} \quad (8)$$

$$k_i = u \quad (9)$$

where v represents the mutated vector, which is generated by adding a scaled difference between two randomly selected vectors (k_2 dan k_3 ke vektor target (k_1), where F is the scaling factor that controls the magnitude of the difference. u_j denotes

the value of the trial individual in the z-th dimension, which is selected from either the mutated vector (v_j) or the target vector (k_{ij}), depending on the ratio between a random number (between 0 and 1) and the cross rate (CR). The algorithm iteratively updates the population by selecting trial individuals based on the mutation, crossover, and selection steps.

4) WHALE OPTIMIZATION ALGORITHM (WOA)

The Whale Optimization Algorithm (WOA) is a bio-inspired algorithm modeled on the spiral feeding behavior of humpback whales. It mimics the way whales encircle prey, perform bubble-net attacks, and explore the search space using a spiral-shaped trajectory. This behavior enables WOA to efficiently explore and exploit the solution space [43], [44].

In this study, WOA is integrated into the ensemble and operates on samples with selection probabilities greater than 0.80. The algorithm is described by the following equations:

$$y_i = g - O.Q \tag{10}$$

$$Q = |P.g - y_i|, O = 2a.r_1 - a, P = 2r_2 \tag{11}$$

where y_i represents the current position of the whale (search agent) in the solution space. g is the position of the best solution found so far (global best solution). Q is the distance between the current position of the whale (y_i) and the global best solution, which P is calculated as an absolute difference, where is a random coefficient between 0 and 2. O is a coefficient that controls the shape of the spiral motion, where a is a linear decrease factor over time and r_1 is a random value between 0 and 1. These variables guide the movement of the search agent in the solution space, mimicking the feeding behavior of a humpback whale's bubble net in hunting prey.

5) ADAPTIVE PARAMETES (AP)

Adaptive Parameters (AP) are mechanisms used within optimization algorithms to dynamically adjust control variables during the search process. The primary purpose of AP is to maintain a balanced trade-off between exploration (global search) and exploitation (local refinement), ultimately improving convergence and solution quality.

In this study, AP is utilized to enhance the robustness of the optimization process for data augmentation. By allowing parameters to adjust automatically based on the fitness landscape, the algorithm can reduce manual tuning requirements, respond to complex problem spaces more effectively, and improve both convergence rate and final solution quality [45].

6) ENSEMBLE (COMBINED ALGORITHM)

Ensemble learning is a technique that combines multiple base learners—in this case, optimization algorithms—to improve overall model performance in terms of accuracy and stability. Each individual strategy PSO, GWO, WOA, and DE—has its strengths and weaknesses depending on the nature of the optimization problem. By integrating these methods into an ensemble, the hybrid approach leverages the advantages of each algorithm to achieve better solution quality [46].

This study adopts an ensemble-based metaheuristic optimization [47] approach by integrating Bayesian optimization (via Optuna) with a set of metaheuristic algorithms—PSO, GWO, WOA, and DE. The ensemble employs predetermined selection probabilities to guide the use of each algorithm, enabling layered optimization that adapts effectively to varying dataset complexities. The proposed PSO_GWO_DE_AP_WOA Ensemble thus combines four powerful metaheuristic methods and adaptive parameter tuning to improve convergence, solution quality, and computational efficiency.

The ensemble mechanism randomly selects one of the four optimization strategies for each agent based on a predefined selection probability *r*. This mechanism allows each agent to update its position using a different algorithm, enhancing the diversity and adaptability of the optimization process. The choice of probability *r* is justified by its role as an adaptive selection mechanism that balances exploration and exploitation across different phases of the search. The range of *r* is divided into four intervals so that each algorithm has a fair yet adaptive allocation.

PSO is assigned to lower *r* values because it is effective in early global exploration and demonstrates fast convergence in initial iterations [48]. GWO occupies the lower-middle range, stabilizing the process through adaptive social hierarchy and encircling mechanisms that balance exploration and exploitation [49]. WOA is positioned in the upper-middle interval, where it exploits promising regions while maintaining population diversity as its exploration–exploitation ratio naturally shifts during iterations [50]. Finally, DE is assigned to higher *r* values due to its strength in intensive exploitation and fine-tuning, which refines solutions in the final stages of optimization [51]. This balanced allocation ensures efficient convergence, reduces the risk of overfitting, and improves the stability of solutions across diverse datasets.

The selection of the interval for the *r* value in each optimization method is based on a series of comprehensive experiments with systematically tested parameter configurations. Five different interval configurations are evaluated using four machine learning models, namely SVC, KNN, Decision Tree, and Random Forest, as follows:

1. PSO [0,0.25), GWO[0.25,0.5), WOA [0.5,0.75), DE [0.75,1]
2. PSO [0,0.1), GWO [0.1,0.5), WOA [0.5,0.6), DE [0.6,1]
3. PSO [0,0.33), GWO [0.33,0.66), WOA [0.66,0.8), DE [0.88,1]
4. PSO [0,0.2), GWO [0.2,0.55), WOA [0.55,0.8), DE [0.8,1]
5. PSO [0,0.4), GWO[0.4,0.5), WOA [0.5,0.9), DE [0.9,1]

Based on the evaluation of all configurations, the configuration that yields the best performance is PSO [0,0.33), GWO [0.33,0.66), WOA [0.66,0.8), and DE [0.88,1]. This configuration achieves an average accuracy of 92.47%, precision of 92.65%, recall of 92.47%, F1-score of 92.33%, and convergence at iteration 26.45, outperforming the other configurations.

Therefore, the selection of the optimization method in the ensemble metaheuristic optimization is as follows:

- If r <0.33, the PSO strategy is used.

- If $0.33 \leq r < 0.660$, the GWO strategy is selected.
- If $0.66 \leq r < 0.80$, the WOA strategy is applied.
- If $r \geq 0.80$, the DE strategy is used.

F. OVERFITTING AND UNDERFITTING

This study adopts the concepts of overfitting and underfitting to assess a model's ability to generalize while accurately learning patterns from the input data. Overfitting arises when a model becomes excessively complex, often due to an inflated number of parameters or estimators [52]. In such cases, the model achieves high accuracy on the training data but demonstrates poor generalization to unseen data. In contrast, underfitting occurs when the model lacks sufficient complexity to capture the underlying structure of the data, resulting in subpar performance on both training and testing datasets.

To address both overfitting and underfitting, this study employs early stopping as the primary control mechanism, including the use of Optuna's pruning mechanism as an early stopping strategy during hyperparameter search. Early stopping halts the training or optimization process once the performance metric reaches a predefined threshold, preventing the model from continuing iterations that would either lead to memorization of the training data (overfitting) or waste computation on redundant updates. By stopping at the point where the model has already captured the essential patterns, early stopping ensures that the solution remains both accurate and efficient.

Early stopping is implemented in multiple stages of the proposed system to prevent unnecessary computation and avoid overfitting [53]. Within the tune_hyperparameters_with_optuna function, early stopping is triggered using the stop_on_accuracy_100 callback, which halts the hyperparameter search when the fitness score (trial.value) reaches 1.0. A similar mechanism is used in the knn_accuracy function to terminate parameter tuning for the kNN algorithm once perfect accuracy is achieved. Additionally, early stopping is incorporated into the optimize method of the OptimizationBase class and its subclasses (e.g., GWO, PSO, ACO), where the process is automatically stopped when the global best fitness value (self.best_score_global) reaches 1.0, even if the maximum iteration limit has not been reached. This approach ensures computational efficiency while maintaining model performance.

To ensure comprehensive model evaluation and enhance generalizability, K-Fold Cross-Validation is utilized. This technique allows for performance assessment across multiple data partitions, reducing bias assoc iated with a single train-test split. Successful mitigation of overfitting typically results in improved accuracy on the test dataset and reduced sensitivity to noise present in the training data. In contrast, resolving underfitting leads to performance gains on both training and testing datasets by enabling the model to more effectively learn relevant data patterns. Collectively, these strategies promote a balanced model capable of both accurate learning and robust generalization.

G. DYNAMIC AUGMENTATION

Dynamic augmentation is a mechanism designed to adaptively adjust the data augmentation ratio based on the model's real-time performance during training [54]. This method is implemented through the (calculate_dynamic_augmentation) function, which monitors the model's accuracy history.

If accuracy improves, the augmentation ratio is reduced to preserve model stability. Conversely, when accuracy declines, the augmentation ratio is increased to introduce additional training samples, thereby reinforcing the model's ability to learn complex or underrepresented classes.

In this study, the augmentation ratio is controlled using a Dynamic Augmentation Adjustment strategy, which operates within a predefined range bounded by min_augment_ratio and max_augment_ratio, as in Algorithm 1 and Algorithm 2. These boundaries serve as safeguards to prevent under- or over-augmentation. Based on empirical tuning, the values are set by default to 0.1 and 0.5, respectively. A minimum value of 0.1 ensures that even well-performing models still receive a sufficient degree of augmentation to enhance data diversity without significantly altering the original distribution. Meanwhile, the maximum value of 0.5 is chosen to avoid excessive augmentation that could introduce noise and degrade the quality of the original data.

Algorithm 1. Dynamic Augmentation Adjustment

Input: fitness_history, max_augment_ratio (default: 0.5), min_augment_ratio (default: 0.1)
Output: augment_ratio

```
If length(fitness_history) < 2 then
  Return max_augment_ratio
Else
  If fitness_history[-1] ≥ fitness_history[-2] then
    augment_ratio = max(min_augment_ratio, max_augment_ratio - 0.05)
  Else
    augment_ratio = max_augment_ratio + 0.05
  End If
End If

Return augment_ratio
```

This mechanism is embedded in the fitness function, where low-confidence predictions (those with probabilities below a defined threshold) are selectively chosen for augmentation. This targeted augmentation improves the model's ability to generalize while preventing overfitting.

Dynamic augmentation has been shown to provide notable benefits, including improved model accuracy, enhanced generalization, and optimized training efficiency. By incorporating augmentation only, when necessary, this approach reduces training time and computational resource usage without compromising performance.

Algorithm 2. Dynamic Data Augmentation Algorithm

Input: training data X_train and y_train, a trained model modelsKNN[idx], and fitness_history.
Output: augmented training data augmented_train_data and labels augmented_train_labels.

```
Compute augmentation ratio:
```

```
    augment_ratio = calculate_dynamic_augmentation(
      fitness_history,
max_augment_ratio=augment_ratio
    )

if augment_ratio > 0:
    Generate augmented data:
      augmented_train_data, augmented_train_labels =
augment_data_based_on_error(
        seed, X_train, y_train, parameters,
modelsKNN[idx], augment_ratio,
num_augmented_samples, threshold
      )
else:
    No augmentation needed:
      augmented_train_data, augmented_train_labels =
X_train, y_train

Return augmented_train_data and
augmented_train_labels.
```

H. HYPERPARAMETER TRAINING

Hyperparameter tuning is a crucial step in ML model development to ensure optimal prediction performance. The selection of appropriate hyperparameter tuning can enhance accuracy, accelerate convergence, and mitigate the risks of overfitting or underfitting. One of the widely adopted hypertuning optimization frameworks is Optuna, which offers efficient automated hyperparameter search using advanced algorithm.

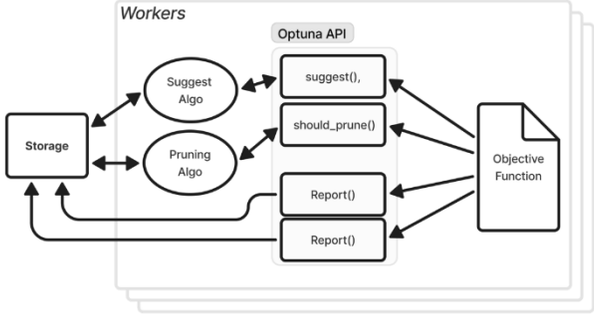


FIGURE 1. Overview of Optuna'0s system design.

Optuna is an open-source framework designed to facilitate automatic, flexible, and efficient hypertuning. Through its optimization-based approach, Optuna intelligently searches for optimal hypertuning combinations to enhance model performance [55], [56]. Optuna employs a define-by-run approach, enabling flexibility in defining hypertuning search spaces during runtime [57]. The primary process in Optuna consists of several stages in Figure 1.

In this study, the hyperparameter search space is defined specifically for each model. For the KNeighborsClassifier, the tuned parameters include the number of neighbors ranging from 1 to 30, the distance metric parameter p with values of either 1 or 2, and a fixed weight configuration set to "uniform." For the SVC model, the parameters optimized are the regularization parameter C ranging from 1e-3 to 1e3 and the kernel coefficient *gamma* ranging from 1e-4 to 1e1, both on a logarithmic scale, while the cache size remains fixed. For the Decision Tree model, the tuning includes the splitting criterion ("gini" or "entropy"), splitter type ("best" or "random"), maximum depth from 1 to 50, minimum samples required to split a node from 2 to 20, and minimum samples required at a leaf node from 1 to 20. Across all models, the number of trials is set to 50, and the tuning process is terminated early if the model reaches 100% accuracy as the stopping criterion.

I. PREPROCESSING DATA

Data preprocessing is a crucial component of the strategy to ensure data quality before use in the modelling process [58], [59]. This step aims to eliminate unreliable data, reduce the risk of delays in the modelling process, address data inaccuracies, and minimize suboptimal model outcomes [60]. Preprocessing encompasses various techniques such as data cleaning, segmentation, and scale adjustment and normalization.

In the data cleaning phase, data with missing values, duplicates, or values considered as outliers are removed or corrected. Data segmentation is performed to divide the dataset into more specific subsets based on certain features to facilitate analysis. Scale adjustment and normalization aim to align the range of values between features so that the model can learn data more effectively, particularly when using algorithms that are sensitive to scale differences, such as distance-based models or gradient descent.

This preprocessing process is conducted using the Pandas library to facilitate data manipulation and processing. The data is stored in Data Frame format throughout this process to maintain its initial structure while simplifying operations such as filtering, aggregation, and transformation. With appropriate preprocessing, the dataset becomes cleaner, more structured, and ready for use in ML modelling [61].

J. PERFORMANCE MEASUREMENT

In this paper, the performance measures used are accuracy, precision, recall, and f1-score. The accuracy is calculated by dividing the number of correct predictions by the total number of samples. However, accuracy can be misleading if the data is between different classes, specifically if there is an imbalance in the number of samples between the different classes. Accuracy tends to give high estimates when the majority class is correctly classified but can neglect performance on the minority class, the equation can be seen in the Equation (12). The calculation of f1 scores is influenced by precision and recall because the f1 score notices false positive and false negative and combines two measures determined based on the total number of properly recognized samples, The equation can be seen in the Equation (13) to Equation (15). Here is the equation of several performance sizes:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

where $accuracy$ is the ratio between the number of properly classified samples and the total number of samples. TP is true

positive, TN is true negative, FP is false positive, and FN is false negative.

$$precision = \frac{TP}{TP + FP} \tag{13}$$

where $precision$ is defined as the correctly recognized positive sample of the total number of samples known as positive.

$$recall = \frac{TP}{TP + FN} \tag{14}$$

where $recall$ is defined as a recognised positive sample correctly calculated out of the total positive samples.

$$f1 - score = \frac{2\ x\ precision\ x\ recall}{precision + recall} \tag{15}$$

where the $f1 - score$ is defined as a comprehensive estimate of the model's accuracy and can be calculated as the harmonious averages of precision and recall.

K. PROCESSED METHOD ADLC

ADLC (Augmented Data Low Confidence) is an algorithm updating method where this research combines various optimization techniques in data augmentation based on low confidence level data to improve accuracy in ML models. Figure 2 shows the ADLC (Augmented Data Low Confidence) working system. The dataset is imported into the system and goes through a preprocessing stage that includes data cleaning and normalization using methods such as Min-Max Scaling or Standard Scaling, depending on the data distribution. Afterwards, the data is split using a k-fold cross-validation technique with $k = 5$ to ensure a representative distribution of training and testing data. The subset data is then trained using the kNN algorithm for probabilistic predicting, where class probabilities are calculated based on the nearest neighbor distribution. For multiclass classification, the algorithm calculates the proportion of neighbors that belong to each class [62]. The highest proportion becomes the predicted class, while the value of that proportion is interpreted as the confidence score. For example, if among 10 nearest neighbors, 6 belong to class A, 3 to class B, and 1 to class C, the model assigns class A with a confidence of 0.6.

K-Nearest Neighbors (kNN) algorithm is used—not for generating new data, but to identify which data points are classified with low confidence based on class probability outputs. In addition to kNN, confidence level can also be calculated using other algorithms such as SVC and Decision Tree. SVC determines prediction confidence based on the distance to the hyperplane, which can be calibrated into probabilities using Platt scaling. Decision Tree, on the other hand, determines it from leaf purity or the proportion of samples belonging to each class in the leaf where the instance falls. The k-NN algorithm is selected in this research due to its simplicity and computational efficiency, making it a practical and widely applicable approach across diverse applications [63].

Once the low-confidence samples are identified, the data augmentation process begins. This is carried out by introducing Gaussian noise to those specific samples, enhancing their diversity while preserving core features. The application of Gaussian noise depends on the type of data. In numerical datasets, noise is added to each feature value by sampling from a Gaussian distribution. In the case of image data, the Digits dataset is used, where each image is originally in an 8×8 format and is flattened into a 64-element feature vector. Gaussian noise is then applied to each pixel value within this flattened representation. For EEG signal data, the process begins with feature extraction using the Discrete Wavelet Transform (DWT). This involves decomposing the raw EEG signal into several wavelet coefficients using the 'sym2' wavelet at level 3. From each set of coefficients, the mean and standard deviation are calculated to form the final feature vector. After the DWT-based features are extracted, Gaussian noise is added to each feature value. The augmentation is further refined using the PGDAWE optimization technique, which controls the level of noise added to maintain data quality and avoid unnecessary distortion. The newly augmented samples are then combined with the original dataset, resulting in a richer and more informative training set that aims to improve model accuracy.

The next stage is training using ML algorithms such as Support Vector Classifier (SVC), Decision Tree (DT), and KNN. These three models are selected because they represent three fundamental approaches in machine learning, each with a distinct learning mechanism. SVC classifies data by finding the optimal hyperplane that separates classes with the maximum margin. DT constructs a decision structure in the form of a tree, where each node represents a condition leading to a classification outcome. Meanwhile, KNN classifies samples based on the majority class of their nearest neighbors [64]. These differences provide a broad view of how various learning strategies respond to the optimization process and dataset characteristics.

The selected models will then be optimized using Optuna to determine the best hyperparameters for each algorithm

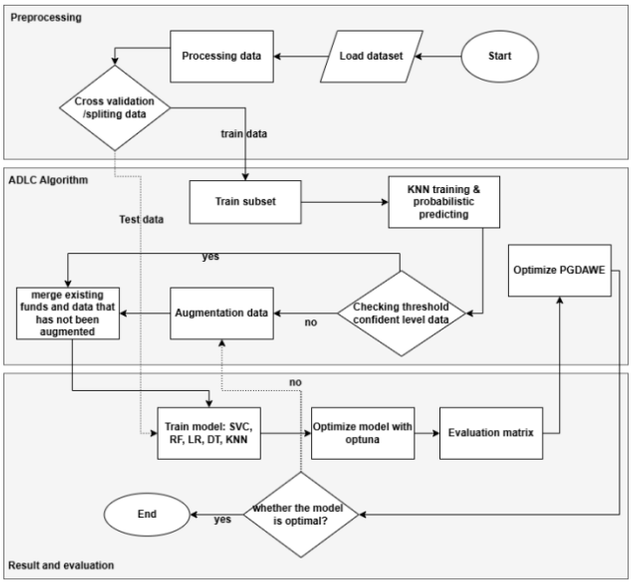


FIGURE 2. ADLS working system.

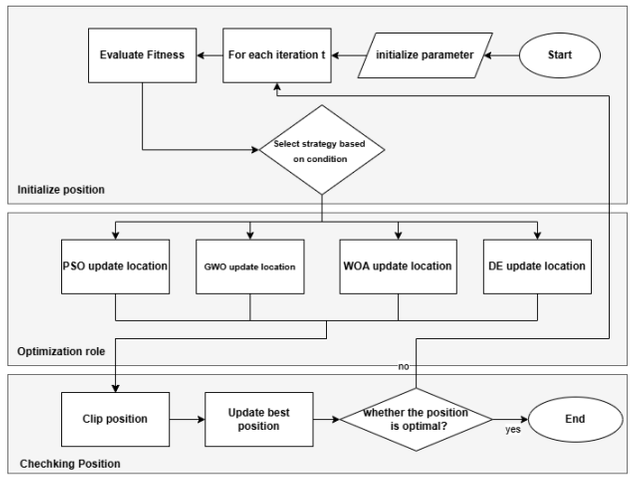


FIGURE 3. The augmented optimizer role.

If the evaluation results show that the model has not achieved optimal performance, the optimization process is repeated using meta-heuristic optimization algorithms such as PSO, GWO, DE, and WOA which are ensembled with a predetermined selection probability. During this phase, the training data may be re-segmented to further improve learning outcomes.

To determine whether a model is considered optimal, accuracy is used as the primary evaluation metric. The optimization process concludes when the model reaches an accuracy threshold of 1.0 (100%), which serves as the main stopping criterion indicating that the model is fully optimal. Once the optimal model is established, a final test is conducted using the test dataset to validate its generalization ability and robustness on new data.

L. AUGMENTED OPTIMIZER ROLE (PGDAWE)

Figure 3 shows the augmented optimizer role where this ensemble optimization algorithm implements a unique approach by combining four popular optimization algorithms such as PSO, GWO, WOA, and DE. The process starts with the initialization of key parameters such as inertia weight of 0.5 which controls the momentum of the particles, cognitive coefficient of 1.5 which controls the influence of personal experience, and social coefficient of 1.5 which determines the influence of group experience. The wolfTrack tracking system is also initialized to monitor the movement of wolves in specified dimensions. In each iteration, the algorithm performs fitness evaluation for each wolf and updates wolfTrack based on the fitness value and position of the wolf. The adaptive parameter is calculated using a specific formula where this value will decrease linearly from 2 to 0 throughout the iteration process. This value decrease plays an important role in balancing the exploration phase at the beginning of optimization and the exploitation phase at the end of optimization.

The selection of the optimization strategy is done by probabilistic selection based on the random value of r. For r values less than 0.33, the algorithm uses PSO which utilizes inertia weight, cognitive coefficient, and social coefficient to update the particle position. If r it is between 0.33 and 0.66, the GWO strategy is applied using the wolf hierarchy (alpha, beta, delta, and omega) and the adaptive parameter. When r between 0.66 and 0.8, the algorithm uses WOA which simulates the spiral movement of killer whales with parameters O (*coefficient vector*), P (*search vector*), and Q (*distance vector*). For r values greater than or equal to 0.8, a DE strategy is used with a differential factor of 0.5 and a crossover rate of 0.9 to perform mutation and crossover.

Each agent position update is limited by bounds [0] as a lower bound and bounds [1] as an upper bound to ensure the solution remains valid. The optimization process will stop if it reaches the maximum iteration or when the best_score_global reaches or exceeds 1. The final results obtained include best_position_global, best_score_global, best_time, iteration, and data_to_save_best. This adaptive approach and combination of various optimization strategies allows the algorithm to utilize the strengths of each method in finding the optimal solution in optimization to get the best, optimal, and consistent accuracy.

III. RESULTS AND DISCUSSION

A. AUGMENTED OPTIMIZER ROLE (PGDAWE)

The experiments in this study were conducted on a computer with an Intel® Core™ i9-13000KS processor, NVIDIA RTX 4080 Super GPU, and 64 GB of RAM. All processes were executed using Python 3.12.0, with support from libraries such as Pandas, NumPy, and Scikit-learn. This research consists of four main experimental scenarios designed to evaluate the effectiveness of the ADLC method in improving ML model performance.

The first scenario aims to analyze how the confidence threshold (T) affects the number of low-confidence samples (Na) and its impact on model accuracy. By varying Na and T, the study examines how these parameters influence the final performance. The second scenario evaluates the performance of ADLC-GWO on numerical datasets to assess the

improvements it brings compared to models without optimization. In the third scenario, the method is applied to image and signal-based datasets to determine its effectiveness in handling different types of data. The final scenario focuses on testing the ADLC method on 12 different datasets, integrating it with various optimization algorithms, including PSO, GWO, AP, Ant Colony Optimization (ACO), DE, and WOA. Additionally, this experiment explores the ensemble optimization approach to assess how combining optimization methods can enhance model generalization.

B. FIRST EXPERIMENTAL SCENARIO

In this section, all experiments were conducted using the Iris dataset, which consists of 150 samples. A total of 120 samples were used for training and 30 samples for testing. The analysis focuses on evaluating the impact of the confidence level threshold on (1) the number of augmented samples and (2) the effect of both the augmentation volume and the confidence threshold on model accuracy.

The confidence level is a probabilistic metric indicating how certain the model is about its prediction [65]. It ranges from 0 to 1, with higher values reflecting greater certainty. In this study, samples with confidence levels below a specified threshold were selected for augmentation. To identify such samples, a k-nearest neighbors (kNN) model was trained independently on each fold using cross-validation. Test data were excluded from the training set to prevent data leakage. After training, the model's output probabilities were filtered using predetermined confidence thresholds.

Four confidence thresholds were defined: $U_1 = 0.6$, $U_2 = 0.7, U_3 = 0.8, U_4 = 0.9$. These values were chosen based on empirical experiments that examined how the effectiveness of data augmentation varies depending on the model's confidence levels.

The number of samples falling below each threshold for every fold is summarized in Table III.

TABLE III
THE NUMBER OF SAMPLES WITH CONFIDENCE LEVELS BELOW THE THRESHOLD

Fold	Threshold			
	U_1	U_2	U_3	U_4
1	2	7	22	38
2	6	7	20	23
3	3	6	6	18
4	7	11	19	36
5	4	9	19	31

As shown, increasing the threshold results in a larger number of samples being classified as low confidence, which are subsequently augmented to enhance training diversity and improve the model's ability to generalize to more complex patterns.

For instance, in Fold 1, with $U_1 = 0.6$, only 2 training samples exhibit low confidence. As the threshold increases, more samples qualify: 7 at U_2, 22 at U_3, and 38 at U_4. Similar trends are observed in the remaining folds, affirming that higher thresholds capture a broader set of uncertain predictions suitable for augmentation.

The augmentation process is governed by key parameters as described in Equation (1). For example, with a threshold $T = 0.6$, a selected number of samples for augmentation $N_a = 10$, and an augmentation ratio $R_a = 0.5$, the resulting number of augmented samples per fold is 10, 30, 15, 35, and 20, as shown in Table III.

Subsequently, the impact of varying N_a on model accuracy was evaluated using values of N_a = 2, 5, 10, and 20. These values were tested across three machine learning models: Decision Tree (DT), k-nearest neighbors (kNN), and Support Vector Classifier (SVC). The augmentation ratio R_a was fixed at 0.5, and the confidence threshold was fixed at $T = 0.8$. The results of this analysis are presented in Table IV. It is evident that $N_a = 10$ produces the highest accuracy across all three models.

In addition to examining the effect of N_a, this study also investigates the impact of varying the threshold T on model performance. The thresholds considered were 0.6, 0.7, 0.8, and 0.9, while $R_a = 0.5$ and $N_a = 10$ were kept constant. Again, DT, kNN, and SVC were used as the base classifiers. The results are shown in Table V, demonstrates that a threshold of $T = 0.8$ yields the highest accuracy across all machine learning models, including 100% accuracy for both DT and SVC.

Based on the results presented in Tables IV and V, the first experimental scenario using the Iris dataset indicates that the optimal parameters are $N_a = 10$ and $T = 0.8$. These parameter values will be consistently applied in subsequent experiments on other datasets to ensure comparability and maintain consistency in the evaluation process.

C. SECOND EXPERIMENTAL SCENARIO

In this section, various performance metrics are used to evaluate the effectiveness of the proposed method compared to the original approach. These metrics include accuracy, precision, recall, and F1 score. The experiments were conducted using three distinct machine learning algorithms, namely Decision Tree (DT), k-nearest neighbors (kNN), and Support Vector Classifier (SVC). These models were selected due to their varied characteristics and their representation of different learning paradigms, which provides a comprehensive evaluation of the proposed method.

To ensure robustness and generalizability, the experiments employed a diverse set of numerical datasets. These include widely used benchmarks such as Iris, Wine, Diabetes, Glass, Car Evaluation, Adult Income, Heart Disease, Breast Cancer, and Pancreatic Ductal Adenocarcinoma (PDAC). The datasets range from small and

TABLE IV
EFFECT OF DIFFERENT N_a MODEL ACCURACY

Model	*accuracy* (%)			
	$N_a = 2$	$N_a = 5$	$N_a = 10$	$N_a = 20$
DT	99.33	99.33	100.00	100.00
kNN	98.00	98.67	99.33	99.33
SVC	99.33	99.33	100.00	100.00

TABLE V
EFFECT OF DIFFERENT T VALLUES ON MODEL ACCURACY

Model	*accuracy* (%)			
	$T = 0.6$	$T = 0.7$	$T = 0.8$	$T = 0.9$
DT	100.00	100.00	100.00	99.33
kNN	98.00	98.00	99.33	98.67
SVC	99.33	99.33	100.00	99.33

structured to large and complex, allowing an assessment of the proposed method across multiple real-world scenarios. All datasets consist primarily of numerical features, making them suitable for the chosen classification tasks.

The experimental results are presented in Table VI, which compares the performance of the baseline methods with the proposed approach, referred to as ADLC with Grey Wolf Optimization (GWO). The proposed method achieves an average accuracy improvement of 6.25 percent across all datasets and models. This improvement demonstrates the ability of the proposed method to enhance predictive performance regardless of dataset structure or complexity.

When analyzed by model type, the accuracy improvements obtained using ADLC are as follows. For the Decision Tree model, ADLC increases the average accuracy by 7.26 percent, representing the most significant improvement among the three models. This result highlights the benefit of adaptive enhancements, especially in datasets characterized by rule-based or hierarchical patterns. For the kNN model, ADLC improves the average accuracy by 5.83 percent. Lastly, the Support Vector Classifier model shows an average accuracy improvement of 5.66 percent. Although this represents the smallest gain among the models, it remains notable and indicates the robustness of the proposed method in improving performance in margin-based classification.

These results confirm that the proposed ADLC method with GWO offers consistent and meaningful performance enhancements across various classification models and datasets.

D. THIRD EXPERIMENTAL SCENARIO

In this section, several performance metrics, including accuracy, precision, recall, and F1 score, are used to evaluate the effectiveness of the proposed method compared to the original approach. The experiments were conducted using three machine learning algorithms, namely Decision Tree (DT), k nearest neighbors (KNN), and Support Vector Classifier (SVC).

Unlike the previous section, which focused on the application of the proposed model to numerical datasets, this section examines its performance in classifying image and signal datasets. The objective is to demonstrate that the proposed method performs effectively across different data types. For image classification, the DIGITS dataset is used. It consists of grayscale images of handwritten digits from zero to nine. For signal classification, the EEG Bonn dataset is employed, which contains electroencephalogram recordings for identifying neural activity patterns.

The experimental results are summarized in Table VII, which compares the performance of the baseline methods with the proposed Adaptive Learning with Data Confidence (ADLC) approach. Across both image and signal datasets, ADLC shows consistent improvements in classification performance.

For the DIGITS dataset, ADLC achieves an average accuracy increase of 1.06 percent across all models. Among individual classifiers, the Decision Tree model demonstrates the highest gain, with an average accuracy increase of 2.06 percent. The KNN and SVC models also show improvements of 0.72 percent and 0.39 percent, respectively.

TABLE VI
IMPROVEMENT ACCURACY USING ADLC-GWO IN NUMERICAL DATASETS

No.	Dataset	model	Original				ADLC-GWO				Improvment ADLC-GWO (%)
			accuracy (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	*accuracy* (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	
1	Iris	DT	93.33	94.32	93.33	93.21	100.00	100.00	100.00	100.00	6.67
		KNN	95.33	95.62	95.33	95.32	99.33	99.39	99.33	99.33	4.00
		SVC	96.00	96.46	96.00	95.98	100.00	100.00	100.00	100.00	4.00
2	Wine	DT	92.73	93.14	92.73	92.75	99.44	99.49	9944	99.45	6.71
		KNN	96.06	96.51	96.06	96.06	99.44	99.49	99.44	99.45	3.38
		SVC	98.30	98.43	98.30	98.30	100.00	100.00	100.00	100.00	1.70
3	Diabetes	DT	98.43	98.49	98.43	98.42	99.64	99.65	99.64	99.63	1.21
		KNN	95.16	95.26	95.16	95.18	97.21	97.37	97.21	97.25	2.06
		SVC	94.91	94.82	94.91	94.76	97.58	97.62	97.58	97.59	2.66
4	Glass	DT	69.15	69.08	69.15	68.08	81.31	80.48	81.31	79.34	12.16
		KNN	65.87	62.21	65.87	62.53	78.50	76.40	78.50	76.06	12.64
		SVC	67.23	61.40	67.23	62.87	79.91	81.15	79.91	78.82	12.68
5	Car Evaluation	DT	97.57	97.66	97.57	97.57	98.32	98.38	98.32	98.32	0.75
		KNN	87.10	88.93	87.10	87.29	96.87	97.08	96.87	96.83	9.78

No.	Dataset	model	Original				ADLC-GWO				Improvment ADLC-GWO (%)
			accuracy (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	*accuracy* (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	
6	Balance Scale	SVC	89.70	88.89	89.70	88.53	99.83	99.83	99.83	99.82	10.12
		DT	76.96	80.78	76.96	78.75	87.04	81.52	87.04	83.94	10.08
		KNN	80.64	80.64	80.64	80.49	91.20	85.60	91.20	87.71	10.56
7	Heart Disease	SVC	90.40	83.48	90.40	86.74	99.84	99.84	99.84	99.84	9.44
		DT	50.15	50.93	50.15	50.15	66.99	65.15	66.99	63.51	16.84
		KNN	58.10	51.76	58.10	53.86	66.01	63.73	66.01	63.10	7.91
8	Breast Cancer	SVC	58.42	50.09	58.42	52.32	66.33	62.31	66.33	63.12	7.91
		DT	92.80	92.89	92.80	92.79	98.24	98.27	98.24	98.24	5.45
		KNN	97.01	97.09	97.01	96.99	98.42	98.46	98.42	98.41	1.40
9	PDAC	SVC	97.89	97.93	97.89	97.88	99.12	99.14	99.12	99.12	1.23
		DT	86.40	86.57	86.40	86.38	95.03	95.10	95.03	95.03	8.63
		KNN	84.30	84.90	84.30	84.28	88.76	89.54	88.76	88.73	4.45
		SVC	83.26	84.01	83.26	83.23	91.89	92.11	91.89	91.89	8.64

TABLE VII.
IMPROVEMENT ACCURACY USING ADLC-GWO IN IMAGE AND SIGNAL DATASETS

No.	Dataset	Model	Original				ADLC-GWO				Improvment ADLC-GWO (%)
			accuracy (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	*accuracy* (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	
1	Digits	DT	84.92	85.20	84.92	84.90	86.98	87.53	86.98	87.05	2.06
		KNN	98.33	98.39	98.33	98.33	99.05	99.08	99.05	99.05	0.72
		SVC	99.00	99.03	99.00	99.00	99.39	99.40	99.39	99.39	0.39
2	EEG Bonn	DT	71.60	71.26	71.6	71.21	80.40	81.27	80.40	80.39	8.80
		KNN	67.80	67.20	67.8	66.79	76.60	76.82	76.60	76.12	8.80
		SVC	53.00	54.59	53	50.01	83.40	83.99	83.40	83.41	30.40

A more pronounced improvement in performance is observed for the EEG Bonn dataset, where the proposed Adaptive Learning with Data Confidence (ADLC) method consistently outperforms the original models across all evaluated metrics. On average, ADLC increases accuracy by 16.00 percent across all classifiers. Both the Decision Tree and k nearest neighbors models achieve an accuracy improvement of 8.80 percent, while the Support Vector Classifier demonstrates the most significant gain with an increase of 30.40 percent. Similar patterns are observed in precision, recall, and F1 score, further confirming the effectiveness of the proposed method.

These results indicate that ADLC not only enhances model performance on numerical datasets but also performs effectively on more complex and noisy data types, such as images and physiological signals. The substantial improvements observed for more challenging datasets suggest that ADLC is a robust and adaptable approach, capable of maintaining high performance across varying data characteristics.

E. FOURTH EXPERIMENTAL SCENARIO

This experiment aims to evaluate the performance of the proposed method while providing insights into its effectiveness by testing it on 12 different datasets. To gain a detailed understanding, ADLC is presented as the proposed method with a focus on improving the performance of ML models. In this study, the proposed method will be combined with various optimization algorithms, including PSO, GWO, AP, ACO, DE, and WOA. The optimization algorithms employed will also be integrated into an ensemble optimization approach.

This experiment aims to evaluate the performance of the proposed method while providing insights into its effectiveness by testing it on 12 different datasets. To gain a

detailed understanding, ADLC is presented as the proposed method with a focus on improving the performance of ML models. In this study, the proposed method will be combined with various optimization algorithms, including PSO, GWO, AP, ACO, DE, and WOA. The optimization algorithms employed will also be integrated into an ensemble optimization approach.

Figure 4 presents a comparison of the performance of various optimization algorithms, including PGDAWE, an ensemble method that leverages the strengths of multiple optimization techniques on diabetes dataset. The dataset originally consisted of fourteen features. To simplify interpretation and enable two-dimensional visualization, Principal Component Analysis (PCA) is applied to reduce the feature space to two principal components. The X-axis represents PCA Component 1 and the Y-axis represents PCA Component 2. During this experiment, the iterations and corresponding accuracies of each algorithm are recorded to evaluate the learning behavior throughout the optimization process. The figure visualizes the progression of each algorithm within the PCA space using four data points. Each point represents the iteration and the accuracy achieved by the algorithm; for instance, (1:99.39) indicates that an accuracy of 99.39% was achieved in the first iteration. PGDAWE exhibits outstanding performance, achieving an initial accuracy of 99.39% and further improving across iterations, ultimately reaching 100% accuracy by iteration 47. Other algorithms generally demonstrate lower accuracy levels. For example, the GA achieves 99.39% accuracy in the first iteration and improves to 99.64% by iteration 47. Similarly, the ACO algorithm starts with an accuracy of 99.39% in the first iteration and reaches 99.54% accuracy by iteration 34. This indicates that PGDAWE not only reaches high accuracy faster but also maintains optimal performance more consistently compared to other optimization methods.

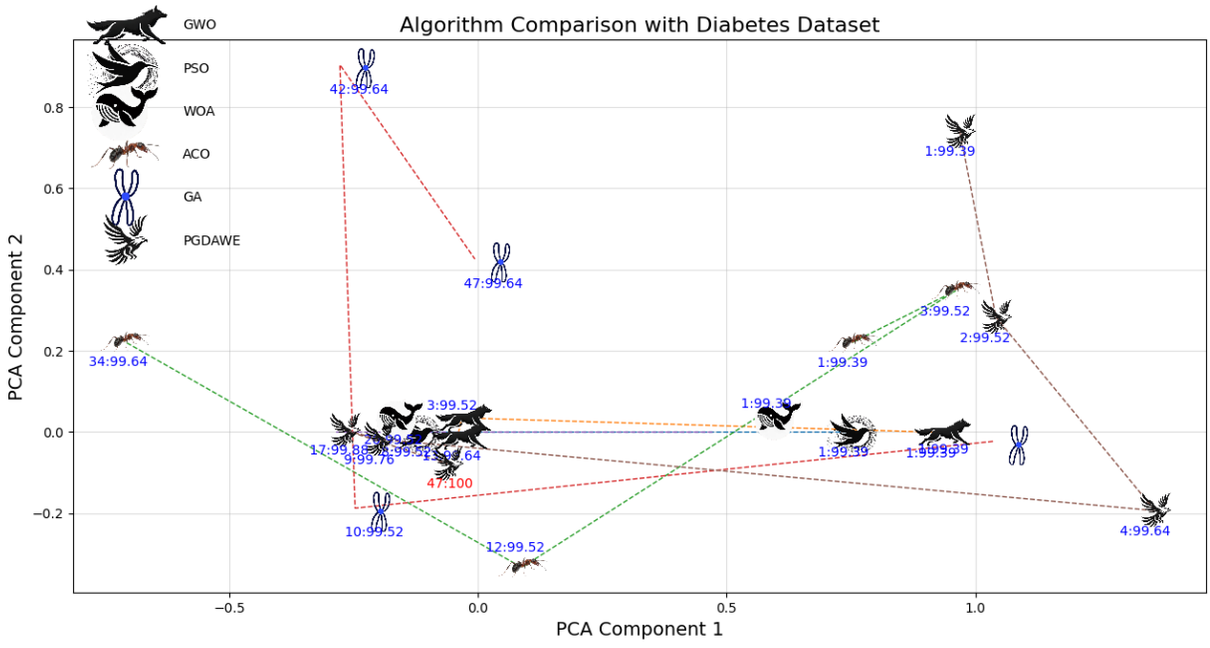


FIGURE 4. Algorithm Comparison on Diabetes Dataset.

TABLE VIII
COMPARISON OPTIMIZATION OF SVC

Dataset	Optimization							
	Original No Optimization	Original Hyper + No Optimization	Proposed Method + GWO	Proposed Method + WOA	Proposed Method + ACO	Proposed Method + PSO	Proposed Method + GA	Proposed Method + Ensemble
Iris	96.00	98.67	100.00	100.00	100.00	100.00	100.00	100.00
Glass	67.23	75.70	79.91	80.35	78.50	79.90	79.42	80.84
Balance Scale	90.40	99.52	99.84	99.84	100.00	96.48	96.32	100.00
Digits	99.00	99.28	99.39	99.39	99.44	99.39	99.44	99.44
Car Evaluation	89.70	99.65	99.83	99.77	99.42	99.71	99.83	99.94
Diabetes	94.91	97.21	97.58	97.58	97.70	97.58	97.82	98.06
EEG Bonn	53.00	81.00	81.60	82.60	80.60	80.60	80.80	84.60
Heart Disease	58.42	62.70	66.66	66.00	66.33	66.32	67.34	68.64

Dataset	Optimization							
	Original No Optimization	Original Hyper + No Optimization	Proposed Method + GWO	Proposed Method + WOA	Proposed Method + ACO	Proposed Method + PSO	Proposed Method + GA	Proposed Method + Ensemble
Wine	98.30	99.43	100.00	100.00	100.00	100.00	100.00	100.00
Breast Cancer Wisconsin	97.89	98.95	99.12	99.12	99.12	99.12	99.12	99.12
Pancreatic Ductal Adenocarcinoma (PDAC)	83.26	89.28	91.89	91.63	91.10	91.64	91.37	92.16
Chronic Kidney Disease (CKD)	98.00	99.25	99.50	99.50	99.50	99.50	99.50	99.50

TABLE IX.
COMPARISON OPTIMIZATION OF KNN

Dataset	Optimization							
	Original No Optimization	Original Hyper + No Optimization	Proposed Method + GWO	Proposed Method + WOA	Proposed Method + ACO	Proposed Method + PSO	Proposed Method + GA	Proposed Method + Ensemble
Iris	95.33	97.33	99.33	98.67	98.67	99.33	98.67	99.33
Glass	65.87	76.14	78.50	78.03	78.50	79.90	77.08	78.98
Balance Scale	80.64	90.40	91.20	91.20	90.88	91.04	91.04	91.20
Digits	98.33	99.00	99.11	99.00	99.05	99.05	99.05	99.00
Car Evaluation	87.10	93.98	96.87	95.26	96.24	95.37	95.37	97.45
Diabetes	95.16	96.00	97.21	97.46	97.46	96.00	96.00	97.46
EEG Bonn	67.80	72.60	77.20	73.80	73.60	72.20	72.00	78.60
Heart Disease	58.10	58.10	66.01	66.34	66.02	64.37	65.02	67.33
Wine	96.06	98.32	99.44	99.44	99.44	98.87	98.87	99.44
Breast Cancer Wisconsin	97.01	97.89	98.42	98.24	98.24	98.24	98.24	98.42
Pancreatic Ductal Adenocarcinoma (PDAC)	84.30	87.18	88.76	88.49	88.75	88.23	89.28	89.29
Chronic Kidney Disease (CKD)	97.00	98.50	99.00	99.00	99.00	99.00	99.00	99.00

TABLE X.
COMPARISON OPTIMIZATION OF DT

Dataset	Optimization							
	Original No Optimization	Original Hyper + No Optimization	Proposed Method + GWO	Proposed Method + WOA	Proposed Method + ACO	Proposed Method + PSO	Proposed Method + GA	Proposed Method + Ensemble
Iris	93.33	98.67	100.00	100.00	100.00	100.00	100.00	100.00
Glass	69.15	78.52	81.31	81.77	81.30	81.32	81.79	84.11
Balance Scale	76.96	84.16	87.04	87.84	86.24	87.84	86.72	87.20
Digits	84.92	86.09	86.59	86.81	86.98	87.15	87.04	86.93
Car Evaluation	97.57	96.64	98.32	98.61	98.44	97.92	97.92	98.73
Diabetes	98.43	98.43	99.64	99.52	99.76	99.76	99.64	100.00
EEG Bonn	71.60	78.20	80.40	82.80	81.80	83.80	81.40	82.80
Heart Disease	50.15	50.15	67.33	66.00	65.35	67.67	67.64	67.32
Wine	92.73	96.63	99.44	99.44	100.00	99.44	99.44	100.00
Breast Cancer Wisconsin	92.80	96.49	98.24	98.24	97.89	97.89	98.42	98.59
Pancreatic Ductal Adenocarcinoma (PDAC)	86.40	90.57	95.03	94.50	93.99	94.24	95.03	96.07
Chronic Kidney Disease (CKD)	95.75	98.75	99.5	99.25	99.50	99.50	99.25	99.50

Tables VIII, IX, and X present a comparison of the accuracy achieved by the ML models using different optimization algorithms. Table VIII shows the optimization results for the Support Vector Classifier (SVC), where the ensemble method consistently provides the best accuracy on most datasets, such as Iris, Balance Scale and Wine achieving 100% accuracy, as well as very good results on more complex datasets such as EEG Bonn (84.60%) and PDAC (92.16%). Table IX presents the optimization results for kNN, which also shows the superiority of the ensemble method with the highest accuracy on datasets such as Iris (99.33%), Balance Scale (91.20%), Digits (99.00%), and chronic kidney disease (99.00%), including very good results on complex datasets such as PDAC (89.29%). Table X shows the optimization results for Decision Tree, where the ensemble method performs best, achieving the highest accuracy on simple datasets such as Iris, Diabetes, and Wine (100.00%), along with significant improvements on more complex datasets such as Glass and Balance Scale. Overall, these experimental results emphasize that the proposed method, combined with the ensemble optimization algorithm, is the most effective approach to improve the accuracy of SVC, KNN, and Decision Tree models across a wide range of datasets.

It should be noted that several datasets achieved 100% accuracy after optimization (e.g., Iris, Wine, and Diabetes). This result does not necessarily indicate overfitting but rather reflects the relatively small size and clear separability of these datasets. For instance, in the Iris dataset, accuracy improved from 98.67% to 100%, which corresponds to correcting only two previously misclassified samples. Likewise, the Wine dataset increased from 96.63% to 100% (around six samples corrected), and Diabetes improved from 98.43% to 100%. Across all datasets, the average accuracy gain from hyperparameter tuning to the proposed ensemble method was +3.99% (σ = 4.48%), with improvements ranging from +0.75% to +17%. Importantly, the datasets that achieved perfect accuracy were still within the normal range of variation, falling inside the 95% confidence interval of improvement (−4.78% to +12.78%). In other words, these perfect accuracies are statistically consistent with the overall performance distribution and should be regarded as a natural outcome of the dataset characteristics rather than evidence of overfitting.

ADLC optimized with an ensemble not only excels in achieving the highest accuracy on simple datasets but also demonstrates significant performance consistency on complex datasets. This advantage indicates that the ensemble approach can handle data with different characteristics. Therefore, this approach can be considered a primary recommendation for implementing ML model optimization that requires high performance and stability across various application scenarios.

Table XI presents the performance of ADLC with ensemble optimization, evaluated using three metrics: accuracy, delivery time, and number of iterations. SVC achieves 100% accuracy on simpler datasets but requires significantly longer training times on more complex datasets. On the Balance Scale dataset, the total training duration reached 83,989.49 seconds, consisting of an initialization time of 82,833.38 seconds and an augmentation time of 0.01 seconds. Similarly, on the EEG Bonn dataset, the total duration was 265,409.70 seconds, with an initialization time of 264,237.27 seconds and an augmentation time of 0.01 seconds. These results highlight the substantial computational cost of SVC on complex data. By contrast, KNN provides a balance between accuracy and computational efficiency, performing well on the Wine dataset but less effectively on more complex datasets. Decision Tree remains the most time-efficient, although its accuracy varies depending on the dataset. Overall, while SVC consistently delivers the highest accuracy, its performance comes at the expense of significantly longer training times on complex datasets.

TABLE XI.
RESULT OF PROPOSED METHOD WITH ENSEMBLE OPTIMIZATION

Classifier	SVC			kNN			DT		
Dataset	*accuracy* (%)	Time	Iteration	*accuracy* (%)	Time	Iteration	*accuracy* (%)	Time	Iteration
Iris	100.00	2.79	1	99.33	48.53	5	100.00	46.62	4
Glass	80.84	1933.12	45	78.98	248.21	21	84.11	1116.56	60
Balance Scale	100.00	83989.49	94	91.20	627.77	37	87.20	1463.25	96
Digits	99.44	151.73	2	99.00	0.44	1	86.93	4806.38	65
Car Evaluation	99.94	2047.86	3	97.45	3370.07	90	98.73	53.94	3
Diabetes	98.06	3476.17	82	97.46	430.37	21	100.00	207.06	13
EEG Bonn	84.60	265409.70	49	78.60	1284.40	82	82.80	1709.12	65
Heart Disease	68.64	2553.16	66	67.33	799.81	97	67.32	840.32	78
Wine	100.00	2.44	1	99.44	2.52	1	100.00	744.67	53
Breast Cancer Wisconsin	99.12	7.00	1	98.42	216.62	17	98.59	1077.80	48
Pancreatic Ductal Adenocarcinoma (PDAC)	96.07	1264.50	75	89.29	531.23	38	96.07	1264.50	75
Chronic Kidney Disease (CKD)	99.50	3.08	1	99.00	0.69	1	99.50	156.32	11

In addition, a comparison with previous studies employing widely recognized augmentation techniques demonstrates the superior performance of the proposed ADLC method. For the CKD dataset, prior work applying SMOTENC achieved 99.4% accuracy, while SMOTE-based augmentation reached 92%; in contrast, ADLC attained 99.5% with SVC and DT, and consistently outperformed the SMOTE baseline across all classifiers. On the Diabetes dataset, SMOTE augmentation reported 99.7% accuracy, whereas ADLC achieved 99.88% with DT. Similarly, for the Wine dataset, SMOTEHashBoost obtained 99.44%, while ADLC reached 100% with both SVC and DT. These results confirm that ADLC delivers higher accuracy than established augmentation methods across multiple datasets, thereby highlighting its distinct advantage.

CONCLUSION

This study presents a novel data augmentation and optimization framework called Augmented Data Low Confidence (ADLC), designed to improve the accuracy and generalization of machine learning models. The ADLC method identifies data samples with low prediction confidence using the k-nearest neighbors (kNN) algorithm and augments these specific instances to enhance the training dataset. This targeted approach ensures that only the most uncertain and informative samples are enriched, thereby reducing the risk of overfitting and redundancy.

The ADLC framework is further enhanced through integration with various metaheuristic optimization algorithms, including Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO), Whale Optimization Algorithm (WOA), Differential Evolution (DE), and Adaptive Parameters (AP). These optimizers are combined into a single ensemble strategy known as PGDAWE, which dynamically selects the most suitable algorithm based on the prediction confidence level of each data point. This adaptive ensemble optimization mechanism allows for improved convergence speed, higher model stability, and better exploitation of the feature space.

Experiments were conducted across 12 datasets, including numerical, image, and signal data, to evaluate the performance of ADLC in diverse scenarios. The results demonstrate consistent accuracy improvements across three machine learning models: Decision Tree (DT), Support Vector Classifier (SVC), and k-nearest neighbors (kNN). Notably, ADLC with PGDAWE achieved up to 100 percent accuracy on simpler datasets such as Iris, Wine, Balance Scale, and chronic kidney disease. In more complex datasets like EEG Bonn and PDAC, the method significantly outperformed baseline and traditionally tuned models, demonstrating its robustness and adaptability. Among the tested classifiers, the Support Vector Classifier exhibited the highest average accuracy across datasets, particularly when optimized using the ADLC framework with ensemble tuning. Furthermore, the proposed approach showed favorable training efficiency and reduced iteration counts for simpler datasets, while maintaining strong accuracy on complex cases despite longer optimization times.

In conclusion, the ADLC framework, especially when integrated with PGDAWE ensemble optimization, provides a scalable, accurate, and efficient solution for improving machine learning model performance across various data types. Its selective augmentation strategy and adaptive optimization capability make it a promising tool for addressing challenges such as class imbalance, overfitting, and suboptimal hyperparameter tuning in real-world applications.

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support from the Institut Teknologi Sepuluh Nopember for this work, under project scheme of the Publication Writing and IPR Incentive Program (PPHKI) in 2025. Also gratefully to the Faculty of Intelligent Electrical and Informatics Technology for providing the research infrastructure and technical support essential for the completion of this study.

REFERENCES

[1] N. L. Rane, S. K. Mallick, Ö. Kaya, and J. Rane, "Applications of machine learning in healthcare, finance, agriculture, retail, manufacturing, energy, and transportation: A review," in *Applied Machine Learning and Deep Learning: Architectures and Techniques*, Deep Science Publishing, 2024. doi: 10.70593/978-81-981271-4-3_6.

[2] K. Goyle, Q. Xie, and V. Goyle, "DataAssist: A Machine Learning Approach to Data Cleaning and Preparation," in *Intelligent Systems Conference*, Springer, Jul. 2023, pp. 476–486. doi: https://doi.org/10.48550/arXiv.2307.07119.

[3] P. Li, Z. Chen, X. Chu, and K. Rong, "DiffPrep: Differentiable Data Preprocessing Pipeline Search for Learning over Tabular Data," *Proceedings of the ACM on Management of Data*, vol. 1, no. 2, pp. 1–26, Jun. 2023, doi: 10.1145/3589328.

[4] J. Wang *et al.*, "IMWMOTE: A novel oversampling technique for fault diagnosis in heterogeneous imbalanced data," *Expert Syst Appl*, vol. 251, p. 123987, Oct. 2024, doi: 10.1016/j.eswa.2024.123987.

[5] S. Li *et al.*, "Rolling Bearing Fault Diagnosis Under Data Imbalance and Variable Speed Based on Adaptive Clustering Weighted Oversampling," *Reliab Eng Syst Saf*, vol. 244, p. 109938, Apr. 2024, doi: 10.1016/j.ress.2024.109938.

[6] P. Priyanka, N. Kumar, and D. Kumar, "Enhanced Grey Wolf Optimization based Hyper-parameter optimized Convolution Neural Network for Kidney Image Classification," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 11, no. 5, pp. 363–374, May 2023, doi: 10.17762/ijritcc.v11i5.6624.

[7] V. Švábenský, C. Borchers, E. B. Cloude, and A. Shimada, "Evaluating the Impact of Data Augmentation on Predictive Model Performance,"

in *Proceedings of the 15th International Learning Analytics and Knowledge Conference*, New York, NY, USA: ACM, Mar. 2025, pp. 126–136. doi: 10.1145/3706468.3706485.

[8] R. Mohakud and R. Dash, "Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6280–6291, Sep. 2022, doi: 10.1016/j.jksuci.2021.05.012.

[9] C. Chakraborty, A. Kishor, and J. J. P. C. Rodrigues, "Novel Enhanced-Grey Wolf Optimization hybrid machine learning technique for biomedical data computation," *Computers and Electrical Engineering*, vol. 99, p. 107778, Apr. 2022, doi: 10.1016/j.compeleceng.2022.107778.

[10] S. K. Parhi and S. K. Patro, "Prediction of compressive strength of geopolymer concrete using a hybrid ensemble of grey wolf optimized machine learning estimators," *Journal of Building Engineering*, vol. 71, p. 106521, Jul. 2023, doi: 10.1016/j.jobe.2023.106521.

[11] D. Yulvida and A. Saikhu, "Optimizing Adaptive Boosting Model for Breast Cancer Prediction Using Principal Component Analysis and Random Oversampling Techniques," in *2024 8th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, IEEE, Aug. 2024, pp. 173–178. doi: 10.1109/ICITISEE63424.2024.10730573.

[12] S. Quinevera and A. Saikhu, "Optimization of Classification Model for Early Detection of Pancreatic Cancer Using GridSearchCV and Autoencoder," in *2024 8th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, IEEE, Aug. 2024, pp. 133–137. doi: 10.1109/ICITISEE63424.2024.10730676.

[13] R. Mardianto and A. Saikhu, "Development of Pre-Processing for Chronic Kidney Disease Prediction Using K-Nearest Neighbors Imputer and Chi-Square," in *2024 8th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, IEEE, Aug. 2024, pp. 179–184. doi: 10.1109/ICITISEE63424.2024.10730259.

[14] Y. Gong, Q. Wu, M. Zhou, and C. Chen, "A diversity and reliability-enhanced synthetic minority oversampling technique for multi-label learning," *Inf Sci (N Y)*, vol. 690, p. 121579, Feb. 2025, doi: 10.1016/j.ins.2024.121579.

[15] J. Li, Q. Zhu, Q. Wu, and Z. Fan, "A novel oversampling technique for class-imbalanced learning based on SMOTE and natural neighbors," *Inf Sci (N Y)*, vol. 565, pp. 438–455, Jul. 2021, doi: 10.1016/j.ins.2021.03.041.

[16] E. D. Gennatas *et al.*, "Expert-augmented machine learning," *Proceedings of the National Academy of Sciences*, vol. 117, no. 9, pp. 4571–4577, Mar. 2020, doi: 10.1073/pnas.1906831117.

[17] E. Avuçlu, "A new data augmentation method to use in machine learning algorithms using statistical measurements," *Measurement*, vol. 180, p. 109577, Aug. 2021, doi: 10.1016/j.measurement.2021.109577.

[18] M. Mujahid *et al.*, "Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering," *J Big Data*, vol. 11, no. 1, p. 87, Jun. 2024, doi: 10.1186/s40537-024-00943-4.

[19] F. Shen, X. Zhao, G. Kou, and F. E. Alsaadi, "A new deep learning ensemble credit risk evaluation model with an improved synthetic minority oversampling technique," *Appl Soft Comput*, vol. 98, p. 106852, Jan. 2021, doi: 10.1016/j.asoc.2020.106852.

[20] A. A. Khan, O. Chaudhari, and R. Chandra, "A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation," *Expert Syst Appl*, vol. 244, p. 122778, Jun. 2024, doi: 10.1016/j.eswa.2023.122778.

[21] T. Suzuki, "TeachAugment: Data Augmentation Optimization Using Teacher Knowledge," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2022, pp. 10894–10904. doi: 10.1109/CVPR52688.2022.01063.

[22] F. J. Moreno-Barea, J. M. Jerez, and L. Franco, "Improving classification accuracy using data augmentation on small data sets," *Expert Syst Appl*, vol. 161, p. 113696, Dec. 2020, doi: 10.1016/j.eswa.2020.113696.

[23] M. Premkumar *et al.*, "Augmented weighted K-means grey wolf optimizer: An enhanced metaheuristic algorithm for data clustering problems," *Sci Rep*, vol. 14, no. 1, p. 5434, Mar. 2024, doi: 10.1038/s41598-024-55619-z.

[24] M. Zhang, X. Wang, L. Jin, M. Song, and Z. Li, "A new approach for evaluating node importance in complex networks via deep learning methods," *Neurocomputing*, vol. 497, pp. 13–27, Aug. 2022, doi: 10.1016/j.neucom.2022.05.010.

[25] M. Negassi, D. Wagner, and A. Reiterer, "Smart(Sampling)Augment: Optimal and Efficient Data Augmentation for Semantic Segmentation," *Algorithms*, vol. 15, no. 5, p. 165, May 2022, doi: 10.3390/a15050165.

[26] H. Hartono and E. Ongko, "Avoiding Overfitting dan Overlapping in Handling Class Imbalanced Using Hybrid Approach with Smoothed Bootstrap Resampling and Feature Selection," *JOIV : International Journal on Informatics Visualization*,

vol. 6, no. 2, p. 343, Jun. 2022, doi: 10.30630/joiv.6.2.985.

[27] A. Mumuni and F. Mumuni, "Data augmentation: A comprehensive survey of modern approaches," *Array*, vol. 16, p. 100258, Dec. 2022, doi: 10.1016/j.array.2022.100258.

[28] B. Xiong, X. Zhao, Y. Hu, H. Huang, Y. Liu, and Y. Su, "Machine learning assisted empirical formula augmentation," *Mater Des*, vol. 210, p. 110037, Nov. 2021, doi: 10.1016/j.matdes.2021.110037.

[29] P. Mathur, H. Shaikh, F. Sheth, D. Kumar, and A. K. Gupta, "A Computational Intelligence Framework Integrating Data Augmentation and Meta-Heuristic Optimization Algorithms for Enhanced Hybrid Nanofluid Density Prediction Through Machine and Deep Learning Paradigms," *IEEE Access*, vol. 13, pp. 35750–35779, 2025, doi: 10.1109/ACCESS.2025.3543475.

[30] B. Hettwer, "Deep learning-enhanced side-channel analysis of cryptographic implementations," Dissertation, Ruhr-Universität Bochum, Bochum, 2020.

[31] H. Arslan and H. Arslan, "A new COVID-19 detection method from human genome sequences using CpG island features and KNN classifier," *Engineering Science and Technology, an International Journal*, vol. 24, no. 4, pp. 839–847, Aug. 2021, doi: 10.1016/j.jestch.2020.12.026.

[32] J. Gou, W. Qiu, Z. Yi, X. Shen, Y. Zhan, and W. Ou, "Locality constrained representation-based K-nearest neighbor classification," *Knowl Based Syst*, vol. 167, pp. 38–52, Mar. 2019, doi: 10.1016/j.knosys.2019.01.016.

[33] J. Gou, H. Ma, W. Ou, S. Zeng, Y. Rao, and H. Yang, "A generalized mean distance-based k-nearest neighbor classifier," *Expert Syst Appl*, vol. 115, pp. 356–372, Jan. 2019, doi: 10.1016/j.eswa.2018.08.021.

[34] H. Tyralis and G. Papacharalampous, "A review of predictive uncertainty estimation with machine learning," *Artif Intell Rev*, vol. 57, no. 4, p. 94, Mar. 2024, doi: 10.1007/s10462-023-10698-8.

[35] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., in Proceedings of Machine Learning Research, vol. 70. PMLR, May 2017, pp. 1321–1330. [Online]. Available: https://proceedings.mlr.press/v70/guo17a.html

[36] U. Johansson, C. Sonstrod, T. Lofstrom, and H. Bostrom, "Confidence Classifiers with Guaranteed Accuracy or Precision," in *Proceedings of the Twelfth Symposium on Conformal and Probabilistic Prediction with Applications*, H. Papadopoulos, K. A. Nguyen, H. Boström, and L. Carlsson, Eds., in Proceedings of Machine Learning Research, vol. 204. PMLR, May 2023, pp. 513–533. [Online]. Available: https://proceedings.mlr.press/v204/johansson23a.html

[37] G. F. Shidik *et al.*, "Bird Species Classification Enhancement via Adaptive Inertia Weight Particle Swarm Optimization-Based Image Augmentation Selection," *IEEE Access*, vol. 12, pp. 197048–197060, 2024, doi: 10.1109/ACCESS.2024.3521455.

[38] F. Zhao, F. Ji, T. Xu, N. Zhu, and Jonrinaldi, "Hierarchical parallel search with automatic parameter configuration for particle swarm optimization," *Appl Soft Comput*, vol. 151, p. 111126, Jan. 2024, doi: 10.1016/j.asoc.2023.111126.

[39] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[40] S. N. Makhadmeh *et al.*, "Recent Advances in Grey Wolf Optimizer, its Versions and Applications: Review," *IEEE Access*, vol. 12, pp. 22991–23028, 2024, doi: 10.1109/ACCESS.2023.3304889.

[41] Y. Shen *et al.*, "Improved differential evolution algorithm based on cooperative multi-population," *Eng Appl Artif Intell*, vol. 133, p. 108149, Jul. 2024, doi: 10.1016/j.engappai.2024.108149.

[42] W. Deng, S. Shang, X. Cai, H. Zhao, Y. Song, and J. Xu, "An improved differential evolution algorithm and its application in optimization problem," *Soft comput*, vol. 25, no. 7, pp. 5277–5298, Apr. 2021, doi: 10.1007/s00500-020-05527-x.

[43] K. A. Alnowibet, S. Shekhawat, A. Saxena, K. M. Sallam, and A. W. Mohamed, "Development and Applications of Augmented Whale Optimization Algorithm," *Mathematics*, vol. 10, no. 12, p. 2076, Jun. 2022, doi: 10.3390/math10122076.

[44] M. Braik, "Hybrid enhanced whale optimization algorithm for contrast and detail enhancement of color images," *Cluster Comput*, vol. 27, no. 1, pp. 231–267, Feb. 2024, doi: 10.1007/s10586-022-03920-9.

[45] K. Meidani, A. Hemmasian, S. Mirjalili, and A. Barati Farimani, "Adaptive grey wolf optimizer," *Neural Comput Appl*, vol. 34, no. 10, pp. 7711–7731, May 2022, doi: 10.1007/s00521-021-06885-9.

[46] J. Yin and N. Li, "Ensemble learning models with a Bayesian optimization algorithm for mineral prospectivity mapping," *Ore Geol Rev*, vol. 145, p. 104916, Jun. 2022, doi: 10.1016/j.oregeorev.2022.104916.

[47] J. L. Potharlanka and N. B. M, "Feature importance feedback with Deep Q process in ensemble-based metaheuristic feature selection algorithms," *Sci Rep*, vol. 14, no. 1, p. 2923, Feb. 2024, doi: 10.1038/s41598-024-53141-w.

[48] T. Shaqarin and B. R. Noack, "A Fast-Converging Particle Swarm Optimization through Targeted, Position-Mutated, Elitism (PSO-TPME)," *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, p. 6, Jan. 2023, doi: 10.1007/s44196-023-00183-z.

[49] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[50] M. H. Nadimi-Shahraki, H. Zamani, Z. Asghari Varzaneh, and S. Mirjalili, "A systematic review of the whale optimization algorithm: theoretical foundation, improvements, and hybridizations," *Archives of Computational Methods in Engineering*, vol. 30, no. 7, pp. 4113–4159, 2023.

[51] M. Chen, C. Feng, and R. Cheng, "Metade: Evolving differential evolution by differential evolution," *IEEE Transactions on Evolutionary Computation*, 2025.

[52] J. A. Cook and J. Ranstam, "Overfitting," *British Journal of Surgery*, vol. 103, no. 13, pp. 1814–1814, Nov. 2016, doi: 10.1002/bjs.10244.

[53] H. Li, G. K. Rajbahadur, D. Lin, C.-P. Bezemer, and Z. M. Jiang, "Keeping Deep Learning Models in Check: A History-Based Approach to Mitigate Overfitting," *IEEE Access*, vol. 12, pp. 70676–70689, 2024, doi: 10.1109/ACCESS.2024.3402543.

[54] C. Aliferis and G. Simon, "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI," 2024, pp. 477–524. doi: 10.1007/978-3-031-39355-6_10.

[55] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA: ACM, Jul. 2019, pp. 2623–2631. doi: 10.1145/3292500.3330701.

[56] S. Dhanka and S. Maini, "A hybridization of XGBoost machine learning model by Optuna hyperparameter tuning suite for cardiovascular disease classification with significant effect of outliers and heterogeneous training datasets," *Int J Cardiol*, vol. 420, p. 132757, Feb. 2025, doi: 10.1016/j.ijcard.2024.132757.

[57] P. Srinivas and R. Katarya, "hyOPTXg: OPTUNA hyper-parameter optimization framework for predicting cardiovascular disease using XGBoost," *Biomed Signal Process Control*, vol. 73, p. 103456, Mar. 2022, doi: 10.1016/j.bspc.2021.103456.

[58] A. K. Gupta, P. Mathur, F. Sheth, C. M. Travieso-Gonzalez, and S. Chaurasia, "Advancing geological image segmentation: Deep learning approaches for rock type identification and classification," *Applied Computing and Geosciences*, vol. 23, p. 100192, Sep. 2024, doi: 10.1016/j.acags.2024.100192.

[59] F. Sheth, P. Mathur, A. K. Gupta, and S. Chaurasia, "An advanced artificial intelligence framework integrating ensembled convolutional neural networks and Vision Transformers for precise soil classification with adaptive fuzzy logic-based crop recommendations," *Eng Appl Artif Intell*, vol. 158, p. 111425, Oct. 2025, doi: 10.1016/j.engappai.2025.111425.

[60] R. Sarno, K. Triyana, S. I. Sabilla, D. R. Wijaya, D. Sunaryono, and C. Fatichah, "Detecting Pork Adulteration in Beef for Halal Authentication Using an Optimized Electronic Nose System," *IEEE Access*, vol. 8, pp. 221700–221711, 2020, doi: 10.1109/ACCESS.2020.3043394.

[61] D. R. Wijaya, R. Sarno, and E. Zulaika, "DWTLSTM for electronic nose signal processing in beef quality monitoring," *Sens Actuators B Chem*, vol. 326, p. 128931, Jan. 2021, doi: 10.1016/j.snb.2020.128931.

[62] Scikit-learn, "KNeighborsClassifier." Accessed: Aug. 08, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

[63] A. Naimi, J. Deng, S. R. Shimjith, and A. J. Arul, "Fault Detection and Isolation of a Pressurized Water Reactor Based on Neural Network and K-Nearest Neighbor," *IEEE Access*, vol. 10, pp. 17113–17121, 2022, doi: 10.1109/ACCESS.2022.3149772.

[64] G. Muhammad *et al.*, "Enhancing Prognosis Accuracy for Ischemic Cardiovascular Disease Using K Nearest Neighbor Algorithm: A Robust Approach," *IEEE Access*, vol. 11, pp. 97879–97895, 2023, doi: 10.1109/ACCESS.2023.3312046.

[65] X. Han, H. Zheng, and M. Zhou, "CARD: Classification and Regression Diffusion Models," Jun. 2022, [Online]. Available: http://arxiv.org/abs/2206.07275

DWI SUNARYONO (Member, IEEE) was born in Surabaya, in May 1972. He received the bachelor's degree in computer engineering, the master's and doctoral degree in informatics engineering from the Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember (ITS), in 1995, 2009, 2024, respectively. He is currently a Lecturer with the Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, ITS. His research interests include face recognition, signal processing, the Internet of Things, intelligent systems, and management information.

RIYANARTO SARNO (Member, IEEE) received the Doctor (Ph.D.) degree, in 1992. He is currently a Professor with the Informatics Department, Institut Teknologi Sepuluh Nopember (ITS). He is interested in research projects about machine learning, the Internet of Things, and knowledge engineering. He has researched E-nose for a period of five years. Being an author of more than five books and over 300 scientific articles led him incorporated in the top 2% world ranking scientist, in 2020, by Standford University.

SHOFFI IZZA SABILLA (Member, IEEE) received the Doctoral. degree in informatics engineering from the Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember (ITS) in 2021. Currently, she is a lecturer in department of medical technology at the Institut Teknologi Sepuluh Nopember (ITS). Her academic and research interests are primarily focused on the intersection of machine learning, optimization, and medical technology. Over the past three years, she has conducted extensive research on E-nose technology, exploring its applications in various fields such as healthcare, diagnostics, and environmental monitoring. Her expertise in machine learning and optimization has significantly contributed to the advancement of these areas, particularly in the development of intelligent systems and sensors for medical and environmental purposes.

RIZQY AHSANA PUTRI was born in Gresik, Jawa Timur, Indonesia, in 2001. She received the master's degree in informatics from Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia, in 2025. She is currently pursuing a doctoral degree in the informatics department, Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. She received a scholarship from the Ministry of Research, Technology, and Higher Education for batch 7 in the PMDSU program. Her research interests include artificial intelligence, machine learning, and signal processing.

TAUFIQ CHOIRUL AMRI was born in Banyuwangi, Jawa Timur, Indonesia. He received a master's degree in Informatics from Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. He is currently pursuing the doctoral degree in Computer Science at the Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. His research interests include Internet of Things, machine learning, artificial intelligence, and signal processing.

IRFAN MIRDA was born in Mataram Baru Lampung Timur, Lampung, in 2002. He received his bachelor's degree in electrical engineering from Universitas Lampung, Indonesia, in 2024. He is currently pursuing a master's degree in the informatics department at Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. He received a scholarship from the Ministry of Research, Technology, and Higher Education for batch 8 in the PMDSU program. His research interests include Internet of Things, data science, artificial intelligence, and robotics.

JOKO SISWANTORO received the B.Sc. degree in Mathematics from Airlangga University, Surabaya, Indonesia, in 1997, the M.Sc. degree in Applied Mathematics from Institut Teknologi Bandung, Bandung, Indonesia, in 2002, and the Ph.D. degree in Computer Science from Universiti Kebangsaan Malaysia, Bangi, Malaysia, in 2016. He is currently an Associate Professor and Head of the Department of Informatics Engineering at the University of Surabaya, Surabaya, Indonesia. His research interests include data science, machine learning, digital image processing, computer vision, and optimization.

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

"Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000."

"Digital Object Identifier 10.1109/ACCESS.2024.Doi Number"

Augmented Data Low Confidence (ADLC): A Confidence-Driven Data Augmentation Framework with Ensemble Optimization for Enhanced Machine Learning Performance

Dwi Sunaryono[1], Riyanarto Sarno[1], Shoffi Izza Sabilla[2], Rizqy Ahsana Putri[1], Taufiq Choirul Amri[1], Irfan Mirda[1], Joko Siswantoro[3]

[1]Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia
[2]Department of Medical Technology, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia
[3]Department of Informatics Engineering, University of Surabaya, Surabaya, 60293, Indonesia

Corresponding author: Dwi Sunaryono (dwi@if.its.ac.id)

ABSTRACT The application of machine learning in data-driven solutions has matured, yet efforts continue to improve predictive accuracy. This study presents a comprehensive approach that begins with data preprocessing, including the removal of invalid values, duplicate entries, feature selection, and dimensionality reduction, followed by model optimization through hyperparameter tuning. A novel method, Augmented-Data Low Confidence, is introduced to enhance model performance by augmenting samples with low prediction confidence. The k-nearest neighbors method is used to estimate prediction probabilities. Samples falling below a defined confidence threshold are selected for augmentation by generating new data points. These points are created by randomly sampling feature values within the upper and lower bounds of the low-confidence instances. The augmented dataset is then optimized using the Gray Wolf Optimization algorithm, which adjusts model parameters based on an accuracy-driven fitness function. Experiments on ten public datasets and two proprietary datasets show that this feedback-based augmentation consistently improves the accuracy of various machine learning models. The results demonstrate the effectiveness of incorporating uncertain predictions into the learning process, leading to improved generalization and classification performance.

INDEX TERMS Augmentation, Machine Learning, Performance Improvement

I. INTRODUCTION

Machine learning (ML) has become an essential tool in many fields including healthcare, finance, and transportation because of its ability to analyze large volumes of data and derive meaningful insights [1]. Classification and regression tasks frequently employ ML models such as k-nearest neighbors (kNN), support vector machines (SVM), and decision trees, with their performance heavily influenced by data preprocessing, feature selection, and optimization strategies [2]. Despite continuous advancements in ML algorithms, persistent challenges, such as class imbalance, overfitting, and suboptimal hyperparameter tuning, continue to hinder model performance [3].

Class imbalance is a well-documented issue in classification tasks, where certain classes contain significantly fewer instances than others, often resulting in biased predictions toward the majority class [4]. To address this challenge, various oversampling methods have been proposed, including the Synthetic Minority Oversampling

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Technique (SMOTE), Improved Majority Weighted Minority Oversampling Technique (IMWMOTE), and Adaptive Clustering Weighted Oversampling (ACWOS) [5]. These approaches enhance classification performance by generating synthetic samples for the minority class. However, they can introduce substantial computational overhead and may produce unrealistic data points, increasing the risk of overfitting. IMWMOTE has demonstrated greater effectiveness than traditional SMOTE by incorporating adaptive noise handling and clustering techniques, making it more suitable for heterogeneous and imbalanced datasets. Nonetheless, its sensitivity to noise and high computational complexity remain significant limitations [4].

Overfitting arises when an ML model captures excessively complex patterns specific to the training data, thereby impairing its ability to generalize to unseen data [6]. To address this challenge, particularly in scenarios involving limited or imbalanced datasets, data augmentation is widely utilized. Techniques such as Gaussian noise injections, SMOTE-ENN (a combination of the Synthetic Minority Oversampling Technique and Edited Nearest Neighbor), and generative models like generative adversarial networks (GANs) are employed to increase the diversity of training samples [7]. However, these methods require careful parameter optimization, as poorly generated synthetic data may be redundant or introduce noise, ultimately compromising model performance.

Hyperparameter tuning is crucial in deciding the performance of ML models. Conventional optimization techniques, such as grid search and random search, are generally inefficient and have a large computing overhead, especially when dealing with high-dimensional parameter fields. To address these issues, researchers have increasingly relied on metaheuristic optimization methods such as Particle Swarm Optimization (PSO), Genetic methods (GA), and Grey Wolf Optimization (GWO) for hyperparameter selection [8]. Among these approaches, Enhanced Grey Wolf Optimization (E-GWO) has shown significant gains in optimizing deep learning architectures, particularly convolutional neural networks (CNNs), delivering higher classification accuracy while requiring less computational resources than PSO and GA [6].

GWO is based on the social structure and cooperative hunting behaviour of grey wolves in the wild. It successfully balances exploration and exploitation in the optimization process, allowing for a more thorough search of the solution space. Unlike traditional methods, GWO dynamically updates the positions of candidate solutions by referencing the most promising individuals, thereby enhancing its ability to escape local optima [9]. This property is particularly beneficial for tuning ML models that involve extensive and complex hyperparameter configurations. In CNN-based image classification tasks, GWO has been shown to reduce classification error rates by up to 39.2 percent compared to PSO and by 15 percent relative to GA, highlighting its efficacy in hyperparameter optimization for deep learning applications [8].

Table I summarizes several previous studies that have applied various approaches to optimize machine learning models, most of which have achieved notable advancements. Despite these advancements, existing methods still face limitations in adaptively managing data augmentation and hyperparameter optimization. To address these challenges, this study proposes Augmented-Data Low Confidence (ADLC), a novel approach that integrates probability-based data augmentation with GWO-based hyperparameter tuning to dynamically improve ML model accuracy. Unlike conventional augmentation techniques that generate synthetic samples indiscriminately, ADLC selectively identifies low-confidence predictions and augments only the most informative instances. This targeted strategy ensures that augmented data contributes meaningfully to the learning process rather than introducing redundancy. Furthermore, the augmentation ratio is adjusted in real-time based on model performance, preventing unnecessary oversampling and reducing the risk of overfitting.

The integration of GWO within the ADLC framework further enhances its effectiveness by providing an adaptive and efficient search mechanism for hyperparameter tuning. This approach reduces computational costs while improving model generalization. Compared to traditional optimization techniques, GWO offers more stable convergence and a superior balance between exploration and exploitation, enabling efficient identification of optimal hyperparameter configurations. Prior studies have shown that GWO can outperform grid search, random search, and even other evolutionary algorithms such as PSO in optimizing ML models [10].

In this study, the ADLC framework is evaluated across multiple datasets, including both publicly available benchmarks and domain-specific research datasets, to assess its effectiveness in addressing class imbalance, overfitting, and inefficient hyperparameter tuning. The experimental results demonstrate that ADLC consistently improves model accuracy and generalization performance when compared to conventional data augmentation and optimization techniques.

Section II presents the materials and methods used in this study, including dataset descriptions, preprocessing steps, and a detailed explanation of the ADLC framework, which integrates probability-based augmentation with GWO. The implementation of dynamic augmentation and GWO-based hyperparameter tuning is also discussed. Section III presents the experimental results and comparative analysis between ADLC and existing approaches, highlighting its impact on model performance. Finally, Section IV summarizes the key findings,

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

emphasizes ADLC's contributions to enhancing ML efficiency, and outlines potential directions for future research. By systematically integrating data augmentation and optimization, ADLC offers a robust, scalable, and effective solution for addressing key challenges in ML.

II. MATERIALS AND METHODS

A. MATERIALS

This study employs a total of twelve datasets, consisting of ten numerical datasets, one image dataset, and one electroencephalogram (EEG) signal dataset. The numerical datasets include Iris, Wine, Glass, Car Evaluation, Balance Scale, Heart Disease, Diabetes, Breast Cancer, Pancreatic Ductal Adenocarcinoma (PDAC), and chronic kidney disease (CKD). The image dataset utilized is Digits, while the signal dataset is EEG Bonn.

Most of the numerical datasets were obtained from the UCI Machine Learning Repository. The Iris dataset comprises 150 samples characterized by four attributes, each representing a specific iris flower species. The Wine dataset includes 178 samples with thirteen chemical attributes corresponding to different wine types. The Glass dataset consists of 214 instances with nine features used to classify glass types. The Car Evaluation dataset includes 1,728 entries and six attributes associated with car acceptability. The Balance Scale dataset contains 625 samples with four features related to the physical balance of a scale. The Heart Disease dataset comprises 303 records and thirteen attributes designed for predicting the risk of heart disease.

The Diabetes dataset, sourced from Mendeley Data, includes 1,000 instances with fourteen attributes, reflecting patient data collected from Medical City Hospital and the Specialized Center for Endocrinology and Diabetes at Al-Kindy Teaching Hospital. The Breast Cancer dataset, retrieved from the Kaggle platform, contains data from 569 patients with thirty-two features, including thirty-one numerical attributes and one classification label [11]. The PDAC dataset, titled "Urinary Biomarkers for Pancreatic Cancer" and available on Kaggle through a contribution by John Davis, includes 590 entries with thirteen input features and one target label [12]. The CKD data set, also sourced from Kaggle, consists of 400 records characterized by twenty-four input attributes and one classification label [13].

The Digits dataset comprises 1,797 grayscale images of handwritten digits, classified into ten distinct categories. The EEG Bonn dataset includes 500 instances with eight features used for EEG signal analysis. A summary of all datasets employed in this study is provided in Table II. In the table, numerical datasets are marked in yellow, image datasets in blue, and signal or EEG datasets in green. The inclusion of heterogeneous datasets aims to comprehensively assess the proposed model's performance across a wide range of real-world scenarios. This strategy enhances the experimental robustness and affirms the model's adaptability across different data modalities. In addition to ensuring reproducibility, the complete source code used for data processing, model training, and metaheuristic algorithm implementation is made publicly available in a GitHub repository (https://github.com/Irfanmrd12/ADLC).

B. AUGMENTATION

Data augmentation is a critical technique in ML, aimed at increasing the volume, quality, and diversity of training data. By generating new data points through various transformations or by incorporating information from multiple sources, data augmentation enriches the original dataset [27]. This process not only expands the dataset size, an essential factor for training robust models, but also introduces variability that enhances model generalization, particularly in cases where the original data is limited or imbalanced [28], [29]. Furthermore, data augmentation improves the informativeness of training samples, making them more valuable for the learning process [27].

TABLE I
PREVIOUS RELATED STUDIES

Author	Title	ML/Processing Method	Optimization Method	Evaluation Results
Jiaxin Wang et al. [4]	"IMWMOTE: A novel oversampling technique for fault diagnosis in heterogeneous imbalanced data"	AHC Clustering, k-NN denoising, ICEEMDAN	IMWMOTE, adaptive noise handling	Sensitivity 100%, G-mean 94.33%, F-measure 94.66%, AUC 95.83%
Sai Li et al. [5]	"Rolling Bearing Fault Diagnosis Under Data Imbalance and Variable Speed Based on Adaptive Clustering Weighted Oversampling"	NAP, Density Peak Clustering, ACWOS	Soft thresholding, weighted oversampling	Accuracy up to 99%
Priyanka et al. [6]	"Enhanced Grey Wolf Optimization based Hyper-parameter optimized Convolution Neural Network for Kidney Image Classification"	CNN	Enhanced Grey Wolf Optimization (E-GWO)	Accuracy 97.01%

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Author	Title	ML/Processing Method	Optimization Method	Evaluation Results
Rasmiranjan Mohakud & Rajashree Dash [8]	"Designing a Grey Wolf Optimization-based Hyper-parameter Optimized CNN Classifier for Skin Cancer Detection"	CNN, ISIC dataset	Grey Wolf Optimization (GWO)	Accuracy 98.33%, Error reduction 39.2% (PSO), 15% (GA)
Yanlu Gong et al. [14]	"A Diversity and Reliability-Enhanced Synthetic Minority Oversampling Technique for Multi-Label Learning"	MLL (BR, CC, MLKNN, ML-RBF)	DR-SMOTE	Performance improvement over MLSMOTE, MLROS, etc.
Junnan Li et al. [15]	"A novel oversampling technique for class-imbalanced learning based on SMOTE and natural neighbours"	3NN, SVM, SMOTE	NaNSMOTE	Improved accuracy and balance over classic SMOTE
Efstathios D. Gennatas et al. [16]	"Expert-Augmented Machine Learning"	RuleFit, Random Forest	Expert-augmented ML (EADLC)	Balanced accuracy increased from 67.3% to 74.4%
Emre Avuçlu [17]	"A new data augmentation method to use in machine learning algorithms using statistical measurements"	k-NN, Decision Tree, Naïve Bayes, Random Forest, SVM	Statistical-based data augmentation	Accuracy improved up to 87.76% (DT, RF, NB)
Valdemar Švábenský et al. [7]	"Evaluating the Impact of Data Augmentation on Predictive Model Performance"	Logistic Regression, SVM, Random Forest, MLP	SMOTE-ENN, Noise Addition	AUC increased by 0.01 over baseline
Muhammad Mujahid et al. [18]	"Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering"	RF, SVM, KNN, AdaBoost, CNN, LSTM, BiLSTM	SMOTE, ADASYN, K-Means SMOTE	SVM + ADASYN accuracy 99.67%, recall 100%
Chinmay Chakraborty et al. [9]	"Novel Enhanced-Grey Wolf Optimization hybrid machine learning technique for biomedical data computation"	Hybrid ML (Bagging & Boosting)	Enhanced Grey Wolf Optimization (E-GWO)	Accuracy 99.26%, Accuracy improvement 11.90%
Suraj Kumar Parhi, Sanjaya Kumar Patro [10]	"Prediction of Compressive Strength of Geopolymer Concrete Using a Hybrid Ensemble of Grey Wolf Optimized Machine Learning Estimators"	Hybrid Ensemble ML (HEML)	Grey Wolf Optimization (GWO)	R^2 Training: 0.97, Testing: 0.94, MAE: 1.34, RMSE: 2.5
Feng Shen et al. [19]	"A New Deep Learning Ensemble Credit Risk Evaluation Model with an Improved Synthetic Minority Oversampling Technique"	LSTM, AdaBoost	Improved SMOTE, Mahalanobis Distance	AUC: 80.32% (German), 74.90% (Taiwan)
Azal Ahmad Khan et al. [20]	"A Review of Ensemble Learning and Data Augmentation Models for Class Imbalanced Problems"	SMOTE, ADASYN, GANs, LightGBM, XGBoost	SMOTE + LightGBM	SMOTE + LightGBM showed best results
Teppei Suzuki [21]	"TeachAugment: Data Augmentation Optimization Using Teacher Knowledge"	Neural Network	Adversarial Data Augmentation, SGD	CIFAR-10: 2.5% error rate, CIFAR-100: 16.8%, ImageNet: 22.2%
Francisco J. Moreno-Barea et al. [22]	"Improving Classification Accuracy Using Data Augmentation on Small Data Sets"	VAE, GANs	Balanced Multiclass, WGAN-GP	Accuracy increased up to 24% on small datasets
Manoharan Premkumar et al. [23]	"Augmented weighted K-means grey wolf optimizer: An enhanced metaheuristic algorithm for data clustering problems"	K-means clustering	Grey Wolf Optimization (GWO)	Better performance than standard GWO
Min Zhang et al. [24]	"A New Approach for Evaluating Node Importance in Complex Networks via Deep Learning Methods"	Graph Neural Network (GNN), CNN	Adam Optimizer	Kendall's τ improved over baseline
Misgana Negassi et al. [25]	"Smart (Sampling)Augment: Optimal and Efficient Data Augmentation for Semantic Segmentation"	Bayesian Optimization, Smart Sampling Augment	SmartAugment	SmartAugment outperformed all previous methods
Hartono, Erianto Ongko [26]	"Avoiding Overfitting and Overlapping in Handling Class Imbalanced Using Hybrid Approach"	Smoothed Bootstrap Resampling, SMOTE	Feature Selection (FAST)	Lower Balanced Error Rate (BER), better than Wrapper Approach-SMOTE

C. K-NEAREST NEIGHBORS (K-NN)

The k-nearest neighbors (kNN) algorithm is a widely used supervised ML technique, known for its simplicity and effectiveness. Unlike traditional models that require training to build predictive functions, kNN adopts an instance-based or lazy learning approach, where the entire training dataset is stored and directly referenced during classification. Each data instance is treated as a point in an n-dimensional feature space. To classify a new sample, the algorithm identifies its k nearest neighbors using a selected distance metric, such as Euclidean or Manhattan distance, and assigns the most frequent class label among them. The choice of k is crucial: a small k may lead to overfitting due to sensitivity to noise, whereas a large k may cause underfitting by smoothing out meaningful distinctions [30], [31].

The Euclidean distance, which measures the straight-line distance between two points in n-dimensional space, is defined in Equation (1).

$$d(y, x_i^j) = \sqrt{(y - x_i^j)^T (y - x_j^i)} \quad (1)$$

where x_i^j represents the i-th training sample from class j.

Alternatively, the Manhattan distance, defined in Equation (2), calculates the sum of absolute differences between corresponding features.

$$d(y, x_i^j) = \sum_{k=1}^{n} |y_k - x_{ik}^j| \quad (2)$$

where x_i^j is the i-th training sample from class j while y_k and x_{ik}^j are their respective feature values.

Recent advancements have focused on improving kNN's robustness, particularly in reducing its sensitivity to the choice of k [32]. For example, Gou et al. proposed weighted representation-based and weighted local mean representation-based kNN variants, which adjust the influence of neighbouring points in classification, thereby enhancing reliability. Additionally, they introduced a generalized mean distance-based method that further refines the selection of neighbourhood size [33].

In this study, kNN is employed not only for classification but also for filtering samples based on confidence thresholds. Only instances that fall below a specified confidence level are selected for augmentation. By clearly separating test and training data, this process ensures a structured and leakage-free augmentation strategy. The use of kNN for confidence-based filtering highlights its utility beyond classification, contributing to both data preprocessing and model optimization.

D. PREDICTION PROBABILISTIC

In ML, probabilistic predictions refer to outputs that assign a probability distribution over potential outcomes, providing not only a classification but also a measure of the model's certainty. The confidence level is a central concept within this framework, indicating how strongly the model believes in each prediction. A higher confidence value reflects greater certainty, while a lower value suggests uncertainty. The threshold defines the minimum acceptable confidence level for a prediction to be considered valid. By adjusting this threshold, practitioners can control the model's behavior, whether prioritizing precision by accepting only high-confidence predictions or favoring broader coverage by including lower-confidence ones, depending on the specific task requirements [34], [35].

TABLE II
SUMMARY OF DATASETS USED

Datasets	Number of Instance	Number of Features	Imbalance Ratio
Iris	150	4	0.00
Wine	178	13	1.48
Glass	214	9	8.44
Car Evaluation	1728	6	20.61
Balance Scale	625	4	5.88
Heart Disease	303	13	1.18
Diabetes	1000	14	0.06
Breast Cancer	569	32	0.59
Pancreatic Ductal Adenocarcinoma (PDAC)	590	14	0.88
Chronic Kidney Disease (CKD)	400	25	0.60
Digits	1797	64	1.05
EEG Bonn	500	8	1.00

This study explores how varying the threshold influences the selection of low-confidence samples for augmentation. Specifically, the threshold determines which predictions are deemed uncertain and subsequently selected for synthetic expansion. Adjusting this parameter enables a flexible trade-off between different performance metrics and affects the flow and quality of augmented data. The relationship between predicted probabilities and empirical accuracy is formalized in Equation (3), which provides the basis for determining confidence levels in a quantifiable manner.

$$p(c_j \mid p^{c_j}) = p^{c_j} \quad (3)$$

where p^{c_j} is the probability estimate for class j. The (average) probability estimate for the predicted label should therefore match the empirical accuracy if it is examined empirically [36].

E. OPTIMIZATION ALGORITHM PSO_GWO_DE_AP_WOA_ENSEMBLE

The PSO_GWO_DE_AP_WOA Ensemble is a hybrid optimization framework that integrates four distinct metaheuristic algorithms: Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO), Whale Optimization Algorithm (WOA), and Differential Evolution (DE). This integrated approach is designed to improve model accuracy and stability by harnessing the complementary strengths of each algorithm in both the exploration and exploitation phases of the search process. In the context of this study, the ensemble is employed to optimize data

augmentation strategies, allowing for the dynamic selection and adjustment of augmentation techniques based on the outcomes of the optimization process. This adaptive mechanism contributes to the generation of a more diverse and representative training dataset, thereby enhancing the overall performance of the model.

1) PARTICLE SWARM OPTIMIZATION (PSO)

The classical Particle Swarm Optimization (PSO) algorithm is inspired by the collective behavior observed in bird flocks during the search for food. In this algorithm, each particle represents a potential solution and adjusts its position within the solution space by referencing three key elements: its current position, its historically best-known position, and the globally best position identified by the swarm [37], [38]. In the present study, PSO is employed to identify the optimal combination of data augmentation techniques that maximize model accuracy. The algorithm is integrated into the ensemble framework and is specifically applied to candidate samples that exhibit predicted probabilities below the threshold of 0.33. The mechanism for updating a particle's position in PSO is described by Equation (4), which models both the velocity and positional changes of each particle:

$$\alpha = w\,x\,\alpha + d_1 x\, e_1\, x\,(\mu - p) + \; d_2 x\, e_2 \;\; x\,(\beta - p) \qquad (4)$$

In this formulation, w denotes the inertia weight, which regulates the influence of the particle's previous velocity on its current motion. The parameter α represents the particle's velocity, determining the rate at which the particle navigates the solution space. The coefficient d_1 reflects the cognitive component, directing the particle toward its personal best position denoted as μ. This component encourages individualized learning based on past success. Conversely, d_2 serves as the social component, guiding the particle toward the global best position β, thereby promoting collective intelligence. The variables e_1 and e_2 are uniformly distributed random numbers between 0 and 1, incorporated to introduce stochastic variability and reduce the likelihood of convergence to local optima. Finally, p denotes the current position of the particle within the solution space. This formulation enables the algorithm to effectively balance exploration and exploitation in the search for optimal solutions.

2) GREY WOLF OPTIMIZATION (GWO)

Grey Wolf Optimization (GWO) is a population-based metaheuristic algorithm that emulates the leadership structure and collaborative hunting strategies observed in grey wolf packs. Within the GWO framework, candidate solutions are ranked according to their fitness values. The most optimal solution is designated as alpha (α), followed by beta (β) for the second-best, delta (δ) for the third best, and the remaining solutions are referred to as omega (ω). The positions of α, β, and δ serve as reference points that guide the movement of the ω wolves, facilitating convergence toward superior solutions by mimicking the wolves' natural encircling and hunting behaviors [39], [40].

In this study, GWO is utilized to optimize the selection of data augmentation techniques, particularly for samples whose predicted probabilities fall within the range of 0.33 to 0.66. This targeted application allows for adaptive augmentation that improves model generalization. The ensemble framework incorporates GWO to strengthen the robustness and consistency of the decision-making process. The mathematical formulation governing the update of candidate positions in GWO is presented in Equations (5) and (6), which define the algorithm's dynamic adjustment process in navigating the search space.

$$y_i = \frac{Y_a + Y_b + Y_x}{3} \qquad (5)$$

$$D_a = |C_1.Y_a - y_i|, D_b = |C_1.Y_b - y_i|, D_x = |C_1.Y_x - y_i| \qquad (6)$$

where y_i represents the current position of the wolf (candidate solution) in the solution space. Y_a is the position of the alpha wolf, which corresponds to the best solution found so far, while Y_b dan Y_x represent the positions of the beta and delta wolves, which correspond to the second and third best solutions, respectively. C_1, C_2, dan C_3 are random coefficients (between 0 and 1) that control the influence of alpha, beta, and delta wolves on the movement of the current solution. D_a, D_b, dan D_d are the distances between the current wolf position and the alpha, beta, and delta wolf positions, each scaled by the corresponding random coefficients (C_1, C_2 , dan C_3) which guide the current solution to a better position in the search space.

3) DIFFERENTIAL EVOLUTION (DE)

Differential Evolution (DE) is an evolutionary optimization algorithm that enhances candidate solutions through the recombination of information derived from multiple individuals within a population. The algorithm operates through four fundamental stages: initialization, mutation, crossover, and selection. In the mutation phase, DE generates mutant vectors by adding the scaled difference between two randomly selected individuals to a third individual. These mutant vectors are subsequently combined with existing solutions through the crossover process to produce trial vectors [41], [42]. The trial vectors are then evaluated based on their fitness, and the superior candidates are retained for the next generation.

In the present study, DE is employed to identify the most effective data augmentation strategies for samples with prediction probabilities ranging from 0.66 to 0.80. This approach enables fine-grained control over augmentation in moderately uncertain regions of the classification space. The mathematical procedures underlying the DE operations are presented in Equations (7) through (9), which formally define the mutation, crossover, and selection mechanisms used in the optimization process.

$$v = \; k_1 + F.(k_2 - k_3) \qquad (7)$$

$$u_j = \{v_j, if\ r_j < CR\ and\ k_{ij}, otherwise\} \qquad (8)$$

$$k_i = u \qquad (9)$$

where v represents the mutated vector, which is generated by adding a scaled difference between two randomly selected vectors (k_2 dan k_3 ke vektor target (k_1), where F is the scaling factor that controls the magnitude of the difference. u_j denotes the value of the trial individual in the z-th dimension, which is

selected from either the mutated vector (v_j) or the target vector (k_{ij}), depending on the ratio between a random number (between 0 and 1) and the cross rate (CR). The algorithm iteratively updates the population by selecting trial individuals based on the mutation, crossover, and selection steps.

4) WHALE OPTIMIZATION ALGORITHM (WOA)

The Whale Optimization Algorithm (WOA) is a bio-inspired algorithm modeled on the spiral feeding behavior of humpback whales. It mimics the way whales encircle prey, perform bubble-net attacks, and explore the search space using a spiral-shaped trajectory. This behavior enables WOA to efficiently explore and exploit the solution space [43], [44].

In this study, WOA is integrated into the ensemble and operates on samples with selection probabilities greater than 0.80. The algorithm is described by the following equations:

$$y_i = g - O.Q \tag{10}$$

$$Q = |P.g - y_i|, O = 2a.r_1 - a, P = 2r_2 \tag{11}$$

where y_i represents the current position of the whale (search agent) in the solution space. g is the position of the best solution found so far (global best solution). Q is the distance between the current position of the whale (y_i) and the global best solution, which P is calculated as an absolute difference, where is a random coefficient between 0 and 2. O is a coefficient that controls the shape of the spiral motion, where a is a linear decrease factor over time and r_1 is a random value between 0 and 1. These variables guide the movement of the search agent in the solution space, mimicking the feeding behavior of a humpback whale's bubble net in hunting prey.

5) ADAPTIVE PARAMETES (AP)

Adaptive Parameters (AP) are mechanisms used within optimization algorithms to dynamically adjust control variables during the search process. The primary purpose of AP is to maintain a balanced trade-off between exploration (global search) and exploitation (local refinement), ultimately improving convergence and solution quality.

In this study, AP is utilized to enhance the robustness of the optimization process for data augmentation. By allowing parameters to adjust automatically based on the fitness landscape, the algorithm can reduce manual tuning requirements, respond to complex problem spaces more effectively, and improve both convergence rate and final solution quality [45].

6) ENSEMBLE (COMBINED ALGORITHM)

Ensemble learning is a technique that combines multiple base learners—in this case, optimization algorithms—to improve overall model performance in terms of accuracy and stability. Each individual strategy PSO, GWO, WOA, and DE—has its strengths and weaknesses depending on the nature of the optimization problem. By integrating these methods into an ensemble, the hybrid approach leverages the advantages of each algorithm to achieve better solution quality [46].

This study adopts an ensemble-based metaheuristic optimization [47] approach by integrating Bayesian optimization (via Optuna) with a set of metaheuristic algorithms—PSO, GWO, WOA, and DE. The ensemble employs predetermined selection probabilities to guide the use of each algorithm, enabling layered optimization that adapts effectively to varying dataset complexities. The proposed PSO_GWO_DE_AP_WOA Ensemble thus combines four powerful metaheuristic methods and adaptive parameter tuning to improve convergence, solution quality, and computational efficiency.

The ensemble mechanism randomly selects one of the four optimization strategies for each agent based on a predefined selection probability r. This mechanism allows each agent to update its position using a different algorithm, enhancing the diversity and adaptability of the optimization process. The choice of probability r is justified by its role as an adaptive selection mechanism that balances exploration and exploitation across different phases of the search. The range of r is divided into four intervals so that each algorithm has a fair yet adaptive allocation.

PSO is assigned to lower r values because it is effective in early global exploration and demonstrates fast convergence in initial iterations [48]. GWO occupies the lower-middle range, stabilizing the process through adaptive social hierarchy and encircling mechanisms that balance exploration and exploitation [49]. WOA is positioned in the upper-middle interval, where it exploits promising regions while maintaining population diversity as its exploration–exploitation ratio naturally shifts during iterations [50]. Finally, DE is assigned to higher r values due to its strength in intensive exploitation and fine-tuning, which refines solutions in the final stages of optimization [51]. This balanced allocation ensures efficient convergence, reduces the risk of overfitting, and improves the stability of solutions across diverse datasets.

The selection of the interval for the r value in each optimization method is based on a series of comprehensive experiments with systematically tested parameter configurations. Five different interval configurations are evaluated using four machine learning models, namely SVC, KNN, Decision Tree, and Random Forest, as follows:

1. PSO [0,0.25), GWO[0.25,0.5), WOA [0.5,0.75), DE [0.75,1]
2. PSO [0,0.1), GWO [0.1,0.5), WOA [0.5,0.6), DE [0.6,1]
3. PSO [0,0.33), GWO [0.33,0.66), WOA [0.66,0.8), DE [0.88,1]
4. PSO [0,0.2), GWO [0.2,0.55), WOA [0.55,0.8), DE [0.8,1]
5. PSO [0,0.4), GWO[0.4,0.5), WOA [0.5,0.9), DE [0.9,1]

Based on the evaluation of all configurations, the configuration that yields the best performance is PSO [0,0.33), GWO [0.33,0.66), WOA [0.66,0.8), and DE [0.88,1]. This configuration achieves an average accuracy of 92.47%, precision of 92.65%, recall of 92.47%, F1-score of 92.33%, and convergence at iteration 26.45, outperforming the other configurations.

Therefore, the selection of the optimization method in the ensemble metaheuristic optimization is as follows:

- If $\boldsymbol{r}$ <0.33, the PSO strategy is used.
- If 0.33≤ $\boldsymbol{r}$ <0.660, the GWO strategy is selected.
- If 0.66≤ $\boldsymbol{r}$ <0.80, the WOA strategy is applied.

- If $r \geq 0.80$, the DE strategy is used.

F. OVERFITTING AND UNDERFITTING

This study adopts the concepts of overfitting and underfitting to assess a model's ability to generalize while accurately learning patterns from the input data. Overfitting arises when a model becomes excessively complex, often due to an inflated number of parameters or estimators [52]. In such cases, the model achieves high accuracy on the training data but demonstrates poor generalization to unseen data. In contrast, underfitting occurs when the model lacks sufficient complexity to capture the underlying structure of the data, resulting in subpar performance on both training and testing datasets.

To address both overfitting and underfitting, this study employs early stopping as the primary control mechanism, including the use of Optuna's pruning mechanism as an early stopping strategy during hyperparameter search. Early stopping halts the training or optimization process once the performance metric reaches a predefined threshold, preventing the model from continuing iterations that would either lead to memorization of the training data (overfitting) or waste computation on redundant updates. By stopping at the point where the model has already captured the essential patterns, early stopping ensures that the solution remains both accurate and efficient.

Early stopping is implemented in multiple stages of the proposed system to prevent unnecessary computation and avoid overfitting [53]. Within the tune_hyperparameters_with_optuna function, early stopping is triggered using the stop_on_accuracy_100 callback, which halts the hyperparameter search when the fitness score (trial.value) reaches 1.0. A similar mechanism is used in the knn_accuracy function to terminate parameter tuning for the kNN algorithm once perfect accuracy is achieved. Additionally, early stopping is incorporated into the optimize method of the OptimizationBase class and its subclasses (e.g., GWO, PSO, ACO), where the process is automatically stopped when the global best fitness value (self.best_score_global) reaches 1.0, even if the maximum iteration limit has not been reached. This approach ensures computational efficiency while maintaining model performance.

To ensure comprehensive model evaluation and enhance generalizability, K-Fold Cross-Validation is utilized. This technique allows for performance assessment across multiple data partitions, reducing bias assoc iated with a single train-test split. Successful mitigation of overfitting typically results in improved accuracy on the test dataset and reduced sensitivity to noise present in the training data. In contrast, resolving underfitting leads to performance gains on both training and testing datasets by enabling the model to more effectively learn relevant data patterns. Collectively, these strategies promote a balanced model capable of both accurate learning and robust generalization.

G. DYNAMIC AUGMENTATION

Dynamic augmentation is a mechanism designed to adaptively adjust the data augmentation ratio based on the model's real-time performance during training [54]. This method is implemented through the (calculate_dynamic_augmentation) function, which monitors the model's accuracy history.

If accuracy improves, the augmentation ratio is reduced to preserve model stability. Conversely, when accuracy declines, the augmentation ratio is increased to introduce additional training samples, thereby reinforcing the model's ability to learn complex or underrepresented classes.

In this study, the augmentation ratio is controlled using a Dynamic Augmentation Adjustment strategy, which operates within a predefined range bounded by min_augment_ratio and max_augment_ratio, as in Algorithm 1 and Algorithm 2. These boundaries serve as safeguards to prevent under- or over-augmentation. Based on empirical tuning, the values are set by default to 0.1 and 0.5, respectively. A minimum value of 0.1 ensures that even well-performing models still receive a sufficient degree of augmentation to enhance data diversity without significantly altering the original distribution. Meanwhile, the maximum value of 0.5 is chosen to avoid excessive augmentation that could introduce noise and degrade the quality of the original data.

Algorithm 1. Dynamic Augmentation Adjustment

```
Input: fitness_history, max_augment_ratio (default:
0.5), min_augment_ratio (default: 0.1)
Output: augment_ratio

If length(fitness_history) < 2 then
  Return max_augment_ratio
Else
  If fitness_history[-1] ≥ fitness_history[-2] then
    augment_ratio = max(min_augment_ratio,
max_augment_ratio - 0.05)
  Else
    augment_ratio = max_augment_ratio + 0.05
  End If
End If

Return augment_ratio
```

This mechanism is embedded in the fitness function, where low-confidence predictions (those with probabilities below a defined threshold) are selectively chosen for augmentation. This targeted augmentation improves the model's ability to generalize while preventing overfitting.

Dynamic augmentation has been shown to provide notable benefits, including improved model accuracy, enhanced generalization, and optimized training efficiency. By incorporating augmentation only, when necessary, this approach reduces training time and computational resource usage without compromising performance.

Algorithm 2. Dynamic Data Augmentation Algorithm

```
Input: training data X_train and y_train, a trained
model modelsKNN[idx], and fitness_history.
Output: augmented training data
augmented_train_data and labels
augmented_train_labels.

Compute augmentation ratio:
  augment_ratio = calculate_dynamic_augmentation(
    fitness_history,
max_augment_ratio=augment_ratio
  )
```

```
if augment_ratio > 0:
   Generate augmented data:
      augmented_train_data, augmented_train_labels =
augment_data_based_on_error(
         seed, X_train, y_train, parameters,
modelsKNN[idx], augment_ratio,
num_augmented_samples, threshold
      )
else:
   No augmentation needed:
      augmented_train_data, augmented_train_labels =
X_train, y_train

Return augmented_train_data and
augmented_train_labels.
```

H. HYPERPARAMETER TRAINING

Hyperparameter tuning is a crucial step in ML model development to ensure optimal prediction performance. The selection of appropriate hyperparameter tuning can enhance accuracy, accelerate convergence, and mitigate the risks of overfitting or underfitting. One of the widely adopted hypertuning optimization frameworks is Optuna, which offers efficient automated hyperparameter search using advanced algorithm.

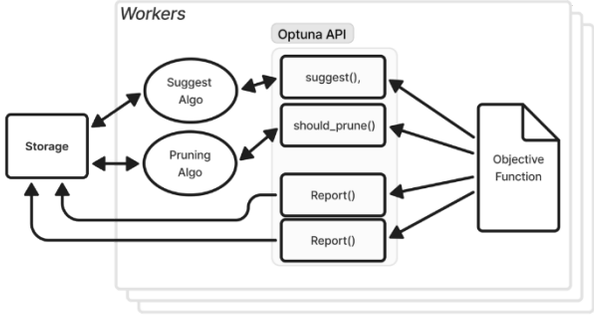


FIGURE 1. Overview of Optuna'0s system design.

Optuna is an open-source framework designed to facilitate automatic, flexible, and efficient hypertuning. Through its optimization-based approach, Optuna intelligently searches for optimal hypertuning combinations to enhance model performance [55], [56]. Optuna employs a define-by-run approach, enabling flexibility in defining hypertuning search spaces during runtime [57]. The primary process in Optuna consists of several stages in Figure 1.

In this study, the hyperparameter search space is defined specifically for each model. For the KNeighborsClassifier, the tuned parameters include the number of neighbors ranging from 1 to 30, the distance metric parameter p with values of either 1 or 2, and a fixed weight configuration set to "uniform." For the SVC model, the parameters optimized are the regularization parameter C ranging from 1e-3 to 1e3 and the kernel coefficient *gamma* ranging from 1e-4 to 1e1, both on a logarithmic scale, while the cache size remains fixed. For the Decision Tree model, the tuning includes the splitting criterion ("gini" or "entropy"), splitter type ("best" or "random"), maximum depth from 1 to 50, minimum samples required to split a node from 2 to 20, and minimum samples required at a leaf node from 1 to 20. Across all models, the number of trials is set to 50, and the tuning process is terminated early if the model reaches 100% accuracy as the stopping criterion.

I. PREPROCESSING DATA

Data preprocessing is a crucial component of the strategy to ensure data quality before use in the modelling process [58], [59]. This step aims to eliminate unreliable data, reduce the risk of delays in the modelling process, address data inaccuracies, and minimize suboptimal model outcomes [60]. Preprocessing encompasses various techniques such as data cleaning, segmentation, and scale adjustment and normalization.

In the data cleaning phase, data with missing values, duplicates, or values considered as outliers are removed or corrected. Data segmentation is performed to divide the dataset into more specific subsets based on certain features to facilitate analysis. Scale adjustment and normalization aim to align the range of values between features so that the model can learn data more effectively, particularly when using algorithms that are sensitive to scale differences, such as distance-based models or gradient descent.

This preprocessing process is conducted using the Pandas library to facilitate data manipulation and processing. The data is stored in Data Frame format throughout this process to maintain its initial structure while simplifying operations such as filtering, aggregation, and transformation. With appropriate preprocessing, the dataset becomes cleaner, more structured, and ready for use in ML modelling [61].

J. PERFORMANCE MEASUREMENT

In this paper, the performance measures used are accuracy, precision, recall, and f1-score. The accuracy is calculated by dividing the number of correct predictions by the total number of samples. However, accuracy can be misleading if the data is between different classes, specifically if there is an imbalance in the number of samples between the different classes. Accuracy tends to give high estimates when the majority class is correctly classified but can neglect performance on the minority class, the equation can be seen in the Equation (12). The calculation of f1 scores is influenced by precision and recall because the f1 score notices false positive and false negative and combines two measures determined based on the total number of properly recognized samples, The equation can be seen in the Equation (13) to Equation (15). Here is the equation of several performance sizes:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

where $accuracy$ is the ratio between the number of properly classified samples and the total number of samples. TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

$$precision = \frac{TP}{TP + FP} \tag{13}$$

where $precision$ is defined as the correctly recognized positive sample of the total number of samples known as positive.

$$recall = \frac{TP}{TP + FN} \tag{14}$$

where $recall$ is defined as a recognised positive sample correctly calculated out of the total positive samples.

$$f1 - score = \frac{2\ x\ precision\ x\ recall}{precision + recall} \tag{15}$$

where the $f1 - score$ is defined as a comprehensive estimate of the model's accuracy and can be calculated as the harmonious averages of precision and recall.

K. PROCESSED METHOD ADLC

ADLC (Augmented Data Low Confidence) is an algorithm updating method where this research combines various optimization techniques in data augmentation based on low confidence level data to improve accuracy in ML models. Figure 2 shows the ADLC (Augmented Data Low Confidence) working system. The dataset is imported into the system and goes through a preprocessing stage that includes data cleaning and normalization using methods such as Min-Max Scaling or Standard Scaling, depending on the data distribution. Afterwards, the data is split using a k-fold cross-validation technique with $k = 5$ to ensure a representative distribution of training and testing data. The subset data is then trained using the kNN algorithm for probabilistic predicting, where class probabilities are calculated based on the nearest neighbor distribution. For multiclass classification, the algorithm calculates the proportion of neighbors that belong to each class [62]. The highest proportion becomes the predicted class, while the value of that proportion is interpreted as the confidence score. For example, if among 10 nearest neighbors, 6 belong to class A, 3 to class B, and 1 to class C, the model assigns class A with a confidence of 0.6.

K-Nearest Neighbors (kNN) algorithm is used—not for generating new data, but to identify which data points are classified with low confidence based on class probability outputs. In addition to kNN, confidence level can also be calculated using other algorithms such as SVC and Decision Tree. SVC determines prediction confidence based on the distance to the hyperplane, which can be calibrated into probabilities using Platt scaling. Decision Tree, on the other hand, determines it from leaf purity or the proportion of samples belonging to each class in the leaf where the instance falls. The k-NN algorithm is selected in this research due to its simplicity and computational efficiency, making it a practical and widely applicable approach across diverse applications [63].

Once the low-confidence samples are identified, the data augmentation process begins. This is carried out by introducing Gaussian noise to those specific samples, enhancing their diversity while preserving core features. The application of Gaussian noise depends on the type of data. In numerical datasets, noise is added to each feature value by sampling from a Gaussian distribution. In the case of image data, the Digits dataset is used, where each image is originally in an 8×8 format and is flattened into a 64-element feature vector. Gaussian noise is then applied to each pixel value within this flattened representation. For EEG signal data, the process begins with feature extraction using the Discrete Wavelet Transform (DWT). This involves decomposing the raw EEG signal into several wavelet coefficients using the 'sym2' wavelet at level 3. From each set of coefficients, the mean and standard deviation are calculated to form the final feature vector. After the DWT-based features are extracted, Gaussian noise is added to each feature value. The augmentation is further refined using the PGDAWE optimization technique, which controls the level of noise added to maintain data quality and avoid unnecessary distortion. The newly augmented samples are then combined with the original dataset, resulting in a richer and more informative training set that aims to improve model accuracy.

The next stage is training using ML algorithms such as Support Vector Classifier (SVC), Decision Tree (DT), and KNN. These three models are selected because they represent three fundamental approaches in machine learning, each with a distinct learning mechanism. SVC classifies data by finding the optimal hyperplane that separates classes with the maximum margin. DT constructs a decision structure in the form of a tree, where each node represents a condition leading to a classification outcome. Meanwhile, KNN classifies samples based on the majority class of their nearest neighbors [64]. These differences provide a broad view of how various learning strategies respond to the optimization process and dataset characteristics.

The selected models will then be optimized using Optuna to determine the best hyperparameters for each algorithm

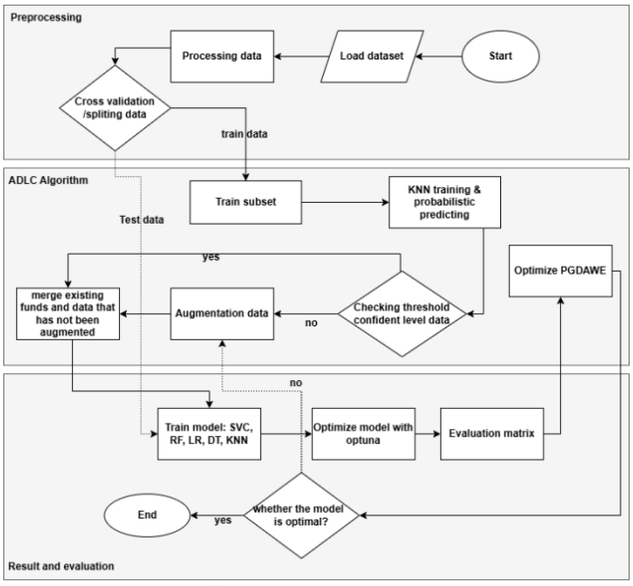


FIGURE 2. ADLS working system.

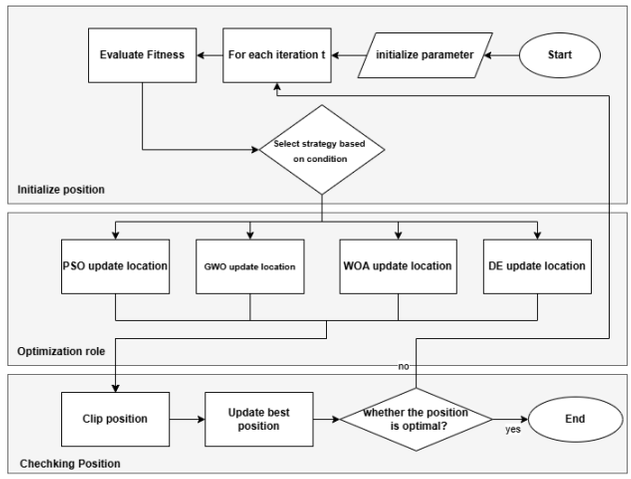


FIGURE 3. The augmented optimizer role.

If the evaluation results show that the model has not achieved optimal performance, the optimization process is repeated using meta-heuristic optimization algorithms such as PSO, GWO, DE, and WOA which are ensembled with a predetermined selection probability. During this phase, the training data may be re-segmented to further improve learning outcomes.

To determine whether a model is considered optimal, accuracy is used as the primary evaluation metric. The optimization process concludes when the model reaches an accuracy threshold of 1.0 (100%), which serves as the main stopping criterion indicating that the model is fully optimal. Once the optimal model is established, a final test is conducted using the test dataset to validate its generalization ability and robustness on new data.

L. AUGMENTED OPTIMIZER ROLE (PGDAWE)

Figure 3 shows the augmented optimizer role where this ensemble optimization algorithm implements a unique approach by combining four popular optimization algorithms such as PSO, GWO, WOA, and DE. The process starts with the initialization of key parameters such as inertia weight of 0.5 which controls the momentum of the particles, cognitive coefficient of 1.5 which controls the influence of personal experience, and social coefficient of 1.5 which determines the influence of group experience. The wolfTrack tracking system is also initialized to monitor the movement of wolves in specified dimensions. In each iteration, the algorithm performs fitness evaluation for each wolf and updates wolfTrack based on the fitness value and position of the wolf. The adaptive parameter is calculated using a specific formula where this value will decrease linearly from 2 to 0 throughout the iteration process. This value decrease plays an important role in balancing the exploration phase at the beginning of optimization and the exploitation phase at the end of optimization.

The selection of the optimization strategy is done by probabilistic selection based on the random value of r. For r values less than 0.33, the algorithm uses PSO which utilizes inertia weight, cognitive coefficient, and social coefficient to update the particle position. If r it is between 0.33 and 0.66, the GWO strategy is applied using the wolf hierarchy (alpha, beta, delta, and omega) and the adaptive parameter. When r between 0.66 and 0.8, the algorithm uses WOA which simulates the spiral movement of killer whales with parameters O (*coefficient vector*), P (*search vector*), and Q (*distance vector*). For r values greater than or equal to 0.8, a DE strategy is used with a differential factor of 0.5 and a crossover rate of 0.9 to perform mutation and crossover.

Each agent position update is limited by bounds [0] as a lower bound and bounds [1] as an upper bound to ensure the solution remains valid. The optimization process will stop if it reaches the maximum iteration or when the best_score_global reaches or exceeds 1. The final results obtained include best_position_global, best_score_global, best_time, iteration, and data_to_save_best. This adaptive approach and combination of various optimization strategies allows the algorithm to utilize the strengths of each method in finding the optimal solution in optimization to get the best, optimal, and consistent accuracy.

III. RESULTS AND DISCUSSION

A. AUGMENTED OPTIMIZER ROLE (PGDAWE)

The experiments in this study were conducted on a computer with an Intel® Core™ i9-13000KS processor, NVIDIA RTX 4080 Super GPU, and 64 GB of RAM. All processes were executed using Python 3.12.0, with support from libraries such as Pandas, NumPy, and Scikit-learn. This research consists of four main experimental scenarios designed to evaluate the effectiveness of the ADLC method in improving ML model performance.

The first scenario aims to analyze how the confidence threshold (T) affects the number of low-confidence samples (Na) and its impact on model accuracy. By varying Na and T, the study examines how these parameters influence the final performance. The second scenario evaluates the performance of ADLC-GWO on numerical datasets to assess the

improvements it brings compared to models without optimization. In the third scenario, the method is applied to image and signal-based datasets to determine its effectiveness in handling different types of data. The final scenario focuses on testing the ADLC method on 12 different datasets, integrating it with various optimization algorithms, including PSO, GWO, AP, Ant Colony Optimization (ACO), DE, and WOA. Additionally, this experiment explores the ensemble optimization approach to assess how combining optimization methods can enhance model generalization.

B. FIRST EXPERIMENTAL SCENARIO

In this section, all experiments were conducted using the Iris dataset, which consists of 150 samples. A total of 120 samples were used for training and 30 samples for testing. The analysis focuses on evaluating the impact of the confidence level threshold on (1) the number of augmented samples and (2) the effect of both the augmentation volume and the confidence threshold on model accuracy.

The confidence level is a probabilistic metric indicating how certain the model is about its prediction [65]. It ranges from 0 to 1, with higher values reflecting greater certainty. In this study, samples with confidence levels below a specified threshold were selected for augmentation. To identify such samples, a k-nearest neighbors (kNN) model was trained independently on each fold using cross-validation. Test data were excluded from the training set to prevent data leakage. After training, the model's output probabilities were filtered using predetermined confidence thresholds.

Four confidence thresholds were defined: $U_1 = 0.6$, $U_2 = 0.7$, $U_3 = 0.8$, $U_4 = 0.9$. These values were chosen based on empirical experiments that examined how the effectiveness of data augmentation varies depending on the model's confidence levels.

The number of samples falling below each threshold for every fold is summarized in Table III.

TABLE III
THE NUMBER OF SAMPLES WITH CONFIDENCE LEVELS BELOW THE THRESHOLD

Fold	Threshold			
	U_1	U_2	U_3	U_4
1	2	7	22	38
2	6	7	20	23
3	3	6	6	18
4	7	11	19	36
5	4	9	19	31

As shown, increasing the threshold results in a larger number of samples being classified as low confidence, which are subsequently augmented to enhance training diversity and improve the model's ability to generalize to more complex patterns.

For instance, in Fold 1, with $U_1 = 0.6$, only 2 training samples exhibit low confidence. As the threshold increases, more samples qualify: 7 at U_2, 22 at U_3, and 38 at U_4. Similar trends are observed in the remaining folds, affirming that higher thresholds capture a broader set of uncertain predictions suitable for augmentation.

The augmentation process is governed by key parameters as described in Equation (1). For example, with a threshold $T = 0.6$, a selected number of samples for augmentation $N_a = 10$, and an augmentation ratio $R_a = 0.5$, the resulting number of augmented samples per fold is 10, 30, 15, 35, and 20, as shown in Table III.

Subsequently, the impact of varying N_a on model accuracy was evaluated using values of N_a = 2, 5, 10, and 20. These values were tested across three machine learning models: Decision Tree (DT), k-nearest neighbors (kNN), and Support Vector Classifier (SVC). The augmentation ratio R_a was fixed at 0.5, and the confidence threshold was fixed at $T = 0.8$. The results of this analysis are presented in Table IV. It is evident that $N_a = 10$ produces the highest accuracy across all three models.

In addition to examining the effect of N_a, this study also investigates the impact of varying the threshold T on model performance. The thresholds considered were 0.6, 0.7, 0.8, and 0.9, while $R_a = 0.5$ and $N_a = 10$ were kept constant. Again, DT, kNN, and SVC were used as the base classifiers. The results are shown in Table V, demonstrates that a threshold of $T = 0.8$ yields the highest accuracy across all machine learning models, including 100% accuracy for both DT and SVC.

Based on the results presented in Tables IV and V, the first experimental scenario using the Iris dataset indicates that the optimal parameters are $N_a = 10$ and $T = 0.8$. These parameter values will be consistently applied in subsequent experiments on other datasets to ensure comparability and maintain consistency in the evaluation process.

C. SECOND EXPERIMENTAL SCENARIO

In this section, various performance metrics are used to evaluate the effectiveness of the proposed method compared

TABLE IV
EFFECT OF DIFFERENT N_a MODEL ACCURACY

Model	*accuracy* (%)			
	$N_a = 2$	$N_a = 5$	$N_a = 10$	$N_a = 20$
DT	99.33	99.33	100.00	100.00
kNN	98.00	98.67	99.33	99.33
SVC	99.33	99.33	100.00	100.00

TABLE V
EFFECT OF DIFFERENT T VALLUES ON MODEL ACCURACY

Model	*accuracy* (%)			
	$T = 0.6$	$T = 0.7$	$T = 0.8$	$T = 0.9$
DT	100.00	100.00	100.00	99.33
kNN	98.00	98.00	99.33	98.67
SVC	99.33	99.33	100.00	99.33

to the original approach. These metrics include accuracy, precision, recall, and F1 score. The experiments were conducted using three distinct machine learning algorithms, namely Decision Tree (DT), k-nearest neighbors (kNN), and Support Vector Classifier (SVC). These models were selected due to their varied characteristics and their representation of different learning paradigms, which provides a comprehensive evaluation of the proposed method.

To ensure robustness and generalizability, the experiments employed a diverse set of numerical datasets. These include widely used benchmarks such as Iris, Wine, Diabetes, Glass, Car Evaluation, Adult Income, Heart Disease, Breast Cancer, and Pancreatic Ductal Adenocarcinoma (PDAC). The datasets range from small and structured to large and complex, allowing an assessment of the proposed method across multiple real-world scenarios.

All datasets consist primarily of numerical features, making them suitable for the chosen classification tasks.

The experimental results are presented in Table VI, which compares the performance of the baseline methods with the proposed approach, referred to as ADLC with Grey Wolf Optimization (GWO). The proposed method achieves an average accuracy improvement of 6.25 percent across all datasets and models. This improvement demonstrates the ability of the proposed method to enhance predictive performance regardless of dataset structure or complexity.

When analyzed by model type, the accuracy improvements obtained using ADLC are as follows. For the Decision Tree model, ADLC increases the average accuracy by 7.26 percent, representing the most significant improvement among the three models. This result highlights the benefit of adaptive enhancements, especially in datasets characterized by rule-based or hierarchical patterns. For the kNN model, ADLC improves the average accuracy by 5.83 percent. Lastly, the Support Vector Classifier model shows an average accuracy improvement of 5.66 percent. Although this represents the smallest gain among the models, it remains notable and indicates the robustness of the proposed method in improving performance in margin-based classification.

These results confirm that the proposed ADLC method with GWO offers consistent and meaningful performance enhancements across various classification models and datasets.

D. THIRD EXPERIMENTAL SCENARIO

In this section, several performance metrics, including accuracy, precision, recall, and F1 score, are used to evaluate the effectiveness of the proposed method compared to the original approach. The experiments were conducted using three machine learning algorithms, namely Decision Tree (DT), k nearest neighbors (KNN), and Support Vector Classifier (SVC).

Unlike the previous section, which focused on the application of the proposed model to numerical datasets, this section examines its performance in classifying image and signal datasets. The objective is to demonstrate that the proposed method performs effectively across different data types. For image classification, the DIGITS dataset is used. It consists of grayscale images of handwritten digits from zero to nine. For signal classification, the EEG Bonn dataset is employed, which contains electroencephalogram recordings for identifying neural activity patterns.

The experimental results are summarized in Table VII, which compares the performance of the baseline methods with the proposed Adaptive Learning with Data Confidence (ADLC) approach. Across both image and signal datasets, ADLC shows consistent improvements in classification performance.

For the DIGITS dataset, ADLC achieves an average accuracy increase of 1.06 percent across all models. Among individual classifiers, the Decision Tree model demonstrates the highest gain, with an average accuracy increase of 2.06 percent. The KNN and SVC models also show improvements of 0.72 percent and 0.39 percent, respectively.

TABLE VI
IMPROVEMENT ACCURACY USING ADLC-GWO IN NUMERICAL DATASETS

No.	Dataset	model	Original				ADLC-GWO				Improvment ADLC-GWO (%)
			$accuracy$ (%)	$precision$ (%)	$recall$ (%)	$f1-score$ (%)	$accuracy$ (%)	$precision$ (%)	$recall$ (%)	$f1-score$ (%)	
1	Iris	DT	93.33	94.32	93.33	93.21	100.00	100.00	100.00	100.00	6.67
		KNN	95.33	95.62	95.33	95.32	99.33	99.39	99.33	99.33	4.00
		SVC	96.00	96.46	96.00	95.98	100.00	100.00	100.00	100.00	4.00
2	Wine	DT	92.73	93.14	92.73	92.75	99.44	99.49	9944	99.45	6.71
		KNN	96.06	96.51	96.06	96.06	99.44	99.49	99.44	99.45	3.38
		SVC	98.30	98.43	98.30	98.30	100.00	100.00	100.00	100.00	1.70
3	Diabetes	DT	98.43	98.49	98.43	98.42	99.64	99.65	99.64	99.63	1.21
		KNN	95.16	95.26	95.16	95.18	97.21	97.37	97.21	97.25	2.06
		SVC	94.91	94.82	94.91	94.76	97.58	97.62	97.58	97.59	2.66
4	Glass	DT	69.15	69.08	69.15	68.08	81.31	80.48	81.31	79.34	12.16
		KNN	65.87	62.21	65.87	62.53	78.50	76.40	78.50	76.06	12.64
		SVC	67.23	61.40	67.23	62.87	79.91	81.15	79.91	78.82	12.68
5	Car Evaluation	DT	97.57	97.66	97.57	97.57	98.32	98.38	98.32	98.32	0.75
		KNN	87.10	88.93	87.10	87.29	96.87	97.08	96.87	96.83	9.78
		SVC	89.70	88.89	89.70	88.53	99.83	99.83	99.83	99.82	10.12

No.	Dataset	model	Original				ADLC-GWO				Improvment ADLC-GWO (%)
			accuracy (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	*accuracy* (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	
6	Balance Scale	DT	76.96	80.78	76.96	78.75	87.04	81.52	87.04	83.94	10.08
		KNN	80.64	80.64	80.64	80.49	91.20	85.60	91.20	87.71	10.56
		SVC	90.40	83.48	90.40	86.74	99.84	99.84	99.84	99.84	9.44
7	Heart Disease	DT	50.15	50.93	50.15	50.15	66.99	65.15	66.99	63.51	16.84
		KNN	58.10	51.76	58.10	53.86	66.01	63.73	66.01	63.10	7.91
		SVC	58.42	50.09	58.42	52.32	66.33	62.31	66.33	63.12	7.91
8	Breast Cancer	DT	92.80	92.89	92.80	92.79	98.24	98.27	98.24	98.24	5.45
		KNN	97.01	97.09	97.01	96.99	98.42	98.46	98.42	98.41	1.40
		SVC	97.89	97.93	97.89	97.88	99.12	99.14	99.12	99.12	1.23
9	PDAC	DT	86.40	86.57	86.40	86.38	95.03	95.10	95.03	95.03	8.63
		KNN	84.30	84.90	84.30	84.28	88.76	89.54	88.76	88.73	4.45
		SVC	83.26	84.01	83.26	83.23	91.89	92.11	91.89	91.89	8.64

TABLE VII.
IMPROVEMENT ACCURACY USING ADLC-GWO IN IMAGE AND SIGNAL DATASETS

No.	Dataset	Model	Original				ADLC-GWO				Improvment ADLC-GWO (%)
			accuracy (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	*accuracy* (%)	*precision* (%)	*recall* (%)	$f1-score$ (%)	
1	Digits	DT	84.92	85.20	84.92	84.90	86.98	87.53	86.98	87.05	2.06
		KNN	98.33	98.39	98.33	98.33	99.05	99.08	99.05	99.05	0.72
		SVC	99.00	99.03	99.00	99.00	99.39	99.40	99.39	99.39	0.39
2	EEG Bonn	DT	71.60	71.26	71.6	71.21	80.40	81.27	80.40	80.39	8.80
		KNN	67.80	67.20	67.8	66.79	76.60	76.82	76.60	76.12	8.80
		SVC	53.00	54.59	53	50.01	83.40	83.99	83.40	83.41	30.40

A more pronounced improvement in performance is observed for the EEG Bonn dataset, where the proposed Adaptive Learning with Data Confidence (ADLC) method consistently outperforms the original models across all evaluated metrics. On average, ADLC increases accuracy by 16.00 percent across all classifiers. Both the Decision Tree and k nearest neighbors models achieve an accuracy improvement of 8.80 percent, while the Support Vector Classifier demonstrates the most significant gain with an increase of 30.40 percent. Similar patterns are observed in precision, recall, and F1 score, further confirming the effectiveness of the proposed method.

These results indicate that ADLC not only enhances model performance on numerical datasets but also performs effectively on more complex and noisy data types, such as images and physiological signals. The substantial improvements observed for more challenging datasets suggest that ADLC is a robust and adaptable approach, capable of maintaining high performance across varying data characteristics.

E. FOURTH EXPERIMENTAL SCENARIO

This experiment aims to evaluate the performance of the proposed method while providing insights into its effectiveness by testing it on 12 different datasets. To gain a detailed understanding, ADLC is presented as the proposed method with a focus on improving the performance of ML models. In this study, the proposed method will be combined with various optimization algorithms, including PSO, GWO, AP, ACO, DE, and WOA. The optimization algorithms employed will also be integrated into an ensemble optimization approach.

This experiment aims to evaluate the performance of the proposed method while providing insights into its effectiveness by testing it on 12 different datasets. To gain a detailed understanding, ADLC is presented as the proposed method with a focus on improving the performance of ML models. In this study, the proposed method will be combined with various optimization algorithms, including PSO, GWO,

AP, ACO, DE, and WOA. The optimization algorithms employed will also be integrated into an ensemble optimization approach.

Figure 4 presents a comparison of the performance of various optimization algorithms, including PGDAWE, an ensemble method that leverages the strengths of multiple optimization techniques on diabetes dataset. The dataset originally consisted of fourteen features. To simplify interpretation and enable two-dimensional visualization, Principal Component Analysis (PCA) is applied to reduce the feature space to two principal components. The X-axis represents PCA Component 1 and the Y-axis represents PCA Component 2. During this experiment, the iterations and corresponding accuracies of each algorithm are recorded to evaluate the learning behavior throughout the optimization process. The figure visualizes the progression of each algorithm within the PCA space using four data points. Each point represents the iteration and the accuracy achieved by the algorithm; for instance, (1:99.39) indicates that an accuracy of 99.39% was achieved in the first iteration. PGDAWE exhibits outstanding performance, achieving an initial accuracy of 99.39% and further improving across iterations, ultimately reaching 100% accuracy by iteration 47. Other algorithms generally demonstrate lower accuracy levels. For example, the GA achieves 99.39% accuracy in the first iteration and improves to 99.64% by iteration 47. Similarly, the ACO algorithm starts with an accuracy of 99.39% in the first iteration and reaches 99.54% accuracy by iteration 34. This indicates that PGDAWE not only reaches high accuracy faster but also maintains optimal performance more consistently compared to other optimization methods.

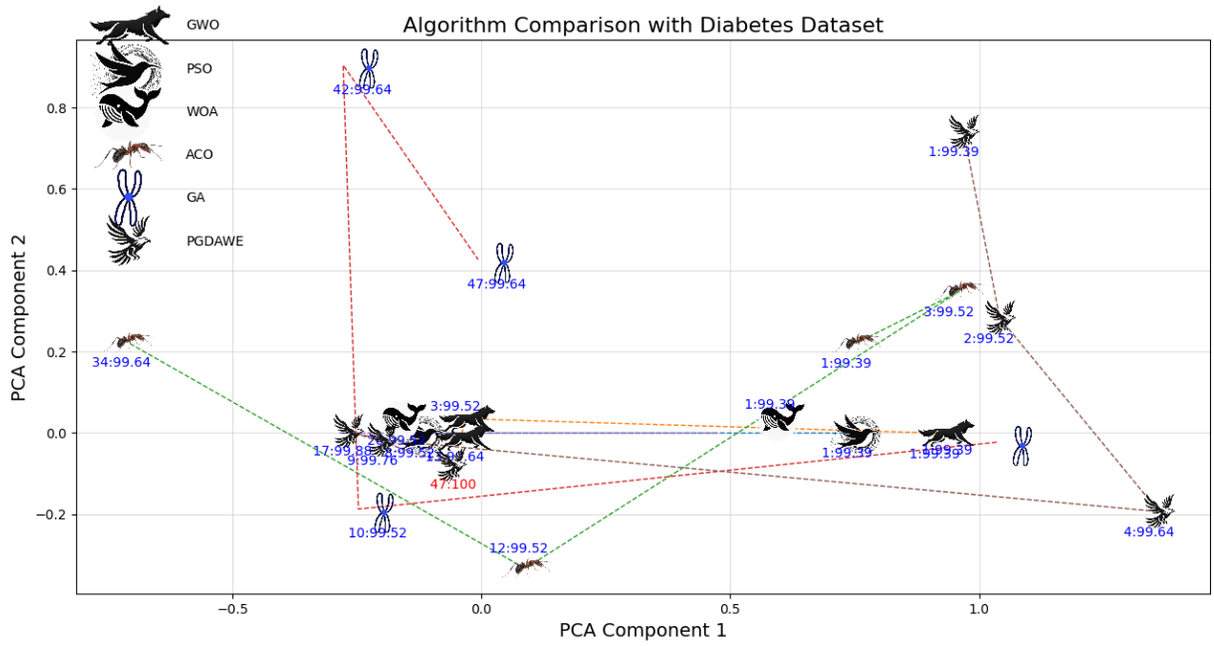


FIGURE 4. Algorithm Comparison on Diabetes Dataset.

TABLE VIII
COMPARISON OPTIMIZATION OF SVC

Dataset	Optimization							
	Original No Optimization	Original Hyper + No Optimization	Proposed Method + GWO	Proposed Method + WOA	Proposed Method + ACO	Proposed Method + PSO	Proposed Method + GA	Proposed Method + Ensemble
Iris	96.00	98.67	100.00	100.00	100.00	100.00	100.00	100.00
Glass	67.23	75.70	79.91	80.35	78.50	79.90	79.42	80.84
Balance Scale	90.40	99.52	99.84	99.84	100.00	96.48	96.32	100.00
Digits	99.00	99.28	99.39	99.39	99.44	99.39	99.44	99.44
Car Evaluation	89.70	99.65	99.83	99.77	99.42	99.71	99.83	99.94
Diabetes	94.91	97.21	97.58	97.58	97.70	97.58	97.82	98.06
EEG Bonn	53.00	81.00	81.60	82.60	80.60	80.60	80.80	84.60
Heart Disease	58.42	62.70	66.66	66.00	66.33	66.32	67.34	68.64
Wine	98.30	99.43	100.00	100.00	100.00	100.00	100.00	100.00
Breast Cancer Wisconsin	97.89	98.95	99.12	99.12	99.12	99.12	99.12	99.12

Dataset	Optimization							
	Original No Optimization	Original Hyper + No Optimization	Proposed Method + GWO	Proposed Method + WOA	Proposed Method + ACO	Proposed Method + PSO	Proposed Method + GA	Proposed Method + Ensemble
Pancreatic Ductal Adenocarcinoma (PDAC)	83.26	89.28	91.89	91.63	91.10	91.64	91.37	92.16
Chronic Kidney Disease (CKD)	98.00	99.25	99.50	99.50	99.50	99.50	99.50	99.50

TABLE IX.
COMPARISON OPTIMIZATION OF KNN

Dataset	Optimization							
	Original No Optimization	Original Hyper + No Optimization	Proposed Method + GWO	Proposed Method + WOA	Proposed Method + ACO	Proposed Method + PSO	Proposed Method + GA	Proposed Method + Ensemble
Iris	95.33	97.33	99.33	98.67	98.67	99.33	98.67	99.33
Glass	65.87	76.14	78.50	78.03	78.50	79.90	77.08	78.98
Balance Scale	80.64	90.40	91.20	91.20	90.88	91.04	91.04	91.20
Digits	98.33	99.00	99.11	99.00	99.05	99.05	99.05	99.00
Car Evaluation	87.10	93.98	96.87	95.26	96.24	95.37	95.37	97.45
Diabetes	95.16	96.00	97.21	97.46	97.46	96.00	96.00	97.46
EEG Bonn	67.80	72.60	77.20	73.80	73.60	72.20	72.00	78.60
Heart Disease	58.10	58.10	66.01	66.34	66.02	64.37	65.02	67.33
Wine	96.06	98.32	99.44	99.44	99.44	98.87	98.87	99.44
Breast Cancer Wisconsin	97.01	97.89	98.42	98.24	98.24	98.24	98.24	98.42
Pancreatic Ductal Adenocarcinoma (PDAC)	84.30	87.18	88.76	88.49	88.75	88.23	89.28	89.29
Chronic Kidney Disease (CKD)	97.00	98.50	99.00	99.00	99.00	99.00	99.00	99.00

TABLE X.
COMPARISON OPTIMIZATION OF DT

Dataset	Optimization							
	Original No Optimization	Original Hyper + No Optimization	Proposed Method + GWO	Proposed Method + WOA	Proposed Method + ACO	Proposed Method + PSO	Proposed Method + GA	Proposed Method + Ensemble
Iris	93.33	98.67	100.00	100.00	100.00	100.00	100.00	100.00
Glass	69.15	78.52	81.31	81.77	81.30	81.32	81.79	84.11
Balance Scale	76.96	84.16	87.04	87.84	86.24	87.84	86.72	87.20
Digits	84.92	86.09	86.59	86.81	86.98	87.15	87.04	86.93
Car Evaluation	97.57	96.64	98.32	98.61	98.44	97.92	97.92	98.73
Diabetes	98.43	98.43	99.64	99.52	99.76	99.76	99.64	100.00
EEG Bonn	71.60	78.20	80.40	82.80	81.80	83.80	81.40	82.80
Heart Disease	50.15	50.15	67.33	66.00	65.35	67.67	67.64	67.32
Wine	92.73	96.63	99.44	99.44	100.00	99.44	99.44	100.00
Breast Cancer Wisconsin	92.80	96.49	98.24	98.24	97.89	97.89	98.42	98.59
Pancreatic Ductal Adenocarcinoma (PDAC)	86.40	90.57	95.03	94.50	93.99	94.24	95.03	96.07
Chronic Kidney Disease (CKD)	95.75	98.75	99.5	99.25	99.50	99.50	99.25	99.50

Tables VIII, IX, and X present a comparison of the accuracy achieved by the ML models using different optimization algorithms. Table VIII shows the optimization results for the Support Vector Classifier (SVC), where the

ensemble method consistently provides the best accuracy on most datasets, such as Iris, Balance Scale and Wine achieving 100% accuracy, as well as very good results on more complex datasets such as EEG Bonn (84.60%) and PDAC (92.16%). Table IX presents the optimization results for kNN, which also shows the superiority of the ensemble method with the highest accuracy on datasets such as Iris (99.33%), Balance Scale (91.20%), Digits (99.00%), and chronic kidney disease (99.00%), including very good results on complex datasets such as PDAC (89.29%). Table X shows the optimization results for Decision Tree, where the ensemble method performs best, achieving the highest accuracy on simple datasets such as Iris, Diabetes, and Wine (100.00%), along with significant improvements on more complex datasets such as Glass and Balance Scale. Overall, these experimental results emphasize that the proposed method, combined with the ensemble optimization algorithm, is the most effective approach to improve the accuracy of SVC, KNN, and Decision Tree models across a wide range of datasets.

It should be noted that several datasets achieved 100% accuracy after optimization (e.g., Iris, Wine, and Diabetes). This result does not necessarily indicate overfitting but rather reflects the relatively small size and clear separability of these datasets. For instance, in the Iris dataset, accuracy improved from 98.67% to 100%, which corresponds to correcting only two previously misclassified samples. Likewise, the Wine dataset increased from 96.63% to 100% (around six samples corrected), and Diabetes improved from 98.43% to 100%. Across all datasets, the average accuracy gain from hyperparameter tuning to the proposed ensemble method was +3.99% (σ = 4.48%), with improvements ranging from +0.75% to +17%. Importantly, the datasets that achieved perfect accuracy were still within the normal range of variation, falling inside the 95% confidence interval of improvement (−4.78% to +12.78%). In other words, these perfect accuracies are statistically consistent with the overall performance distribution and should be regarded as a natural outcome of the dataset characteristics rather than evidence of overfitting.

ADLC optimized with an ensemble not only excels in achieving the highest accuracy on simple datasets but also demonstrates significant performance consistency on complex datasets. This advantage indicates that the ensemble approach can handle data with different characteristics. Therefore, this approach can be considered a primary recommendation for implementing ML model optimization that requires high performance and stability across various application scenarios.

Table XI presents the performance of ADLC with ensemble optimization, evaluated using three metrics: accuracy, delivery time, and number of iterations. SVC achieves 100% accuracy on simpler datasets but requires significantly longer training times on more complex datasets. On the Balance Scale dataset, the total training duration reached 83,989.49 seconds, consisting of an initialization time of 82,833.38 seconds and an augmentation time of 0.01 seconds. Similarly, on the EEG Bonn dataset, the total duration was 265,409.70 seconds, with an initialization time of 264,237.27 seconds and an augmentation time of 0.01 seconds. These results highlight the substantial computational cost of SVC on complex data. By contrast, KNN provides a balance between accuracy and computational efficiency, performing well on the Wine dataset but less effectively on more complex datasets. Decision Tree remains the most time-efficient, although its accuracy varies depending on the dataset. Overall, while SVC consistently delivers the highest accuracy, its performance comes at the expense of significantly longer training times on complex datasets.

TABLE XI.
RESULT OF PROPOSED METHOD WITH ENSEMBLE OPTIMIZATION

Classifier	SVC			kNN			DT		
Dataset	*accuracy* (%)	Time	Iteration	*accuracy* (%)	Time	Iteration	*accuracy* (%)	Time	Iteration
Iris	100.00	2.79	1	99.33	48.53	5	100.00	46.62	4
Glass	80.84	1933.12	45	78.98	248.21	21	84.11	1116.56	60
Balance Scale	100.00	83989.49	94	91.20	627.77	37	87.20	1463.25	96
Digits	99.44	151.73	2	99.00	0.44	1	86.93	4806.38	65
Car Evaluation	99.94	2047.86	3	97.45	3370.07	90	98.73	53.94	3
Diabetes	98.06	3476.17	82	97.46	430.37	21	100.00	207.06	13
EEG Bonn	84.60	265409.70	49	78.60	1284.40	82	82.80	1709.12	65
Heart Disease	68.64	2553.16	66	67.33	799.81	97	67.32	840.32	78
Wine	100.00	2.44	1	99.44	2.52	1	100.00	744.67	53
Breast Cancer Wisconsin	99.12	7.00	1	98.42	216.62	17	98.59	1077.80	48
Pancreatic Ductal Adenocarcinoma (PDAC)	96.07	1264.50	75	89.29	531.23	38	96.07	1264.50	75
Chronic Kidney Disease (CKD)	99.50	3.08	1	99.00	0.69	1	99.50	156.32	11

In addition, a comparison with previous studies employing widely recognized augmentation techniques demonstrates the superior performance of the proposed ADLC method. For the CKD dataset, prior work applying SMOTENC achieved 99.4% accuracy, while SMOTE-based augmentation reached 92%; in contrast, ADLC attained 99.5% with SVC and DT, and consistently outperformed the SMOTE baseline across all classifiers. On the Diabetes dataset, SMOTE augmentation reported 99.7% accuracy, whereas ADLC achieved 99.88% with DT. Similarly, for the Wine dataset, SMOTEHashBoost obtained 99.44%, while ADLC reached 100% with both SVC and DT. These results confirm that ADLC delivers higher accuracy than established augmentation methods across multiple datasets, thereby highlighting its distinct advantage.

CONCLUSION

This study presents a novel data augmentation and optimization framework called Augmented Data Low Confidence (ADLC), designed to improve the accuracy and generalization of machine learning models. The ADLC method identifies data samples with low prediction confidence using the k-nearest neighbors (kNN) algorithm and augments these specific instances to enhance the training dataset. This targeted approach ensures that only the most uncertain and informative samples are enriched, thereby reducing the risk of overfitting and redundancy.

The ADLC framework is further enhanced through integration with various metaheuristic optimization algorithms, including Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO), Whale Optimization Algorithm (WOA), Differential Evolution (DE), and Adaptive Parameters (AP). These optimizers are combined into a single ensemble strategy known as PGDAWE, which dynamically selects the most suitable algorithm based on the prediction confidence level of each data point. This adaptive ensemble optimization mechanism allows for improved convergence speed, higher model stability, and better exploitation of the feature space.

Experiments were conducted across 12 datasets, including numerical, image, and signal data, to evaluate the performance of ADLC in diverse scenarios. The results demonstrate consistent accuracy improvements across three machine learning models: Decision Tree (DT), Support Vector Classifier (SVC), and k-nearest neighbors (kNN). Notably, ADLC with PGDAWE achieved up to 100 percent accuracy on simpler datasets such as Iris, Wine, Balance Scale, and chronic kidney disease. In more complex datasets like EEG Bonn and PDAC, the method significantly outperformed baseline and traditionally tuned models, demonstrating its robustness and adaptability. Among the tested classifiers, the Support Vector Classifier exhibited the highest average accuracy across datasets, particularly when optimized using the ADLC framework with ensemble tuning. Furthermore, the proposed approach showed favorable training efficiency and reduced iteration counts for simpler datasets, while maintaining strong accuracy on complex cases despite longer optimization times.

In conclusion, the ADLC framework, especially when integrated with PGDAWE ensemble optimization, provides a scalable, accurate, and efficient solution for improving machine learning model performance across various data types. Its selective augmentation strategy and adaptive optimization capability make it a promising tool for addressing challenges such as class imbalance, overfitting, and suboptimal hyperparameter tuning in real-world applications.

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support from the Institut Teknologi Sepuluh Nopember for this work, under project scheme of the Publication Writing and IPR Incentive Program (PPHKI) in 2025. Also gratefully to the Faculty of Intelligent Electrical and Informatics Technology for providing the research infrastructure and technical support essential for the completion of this study.

REFERENCES

[1] N. L. Rane, S. K. Mallick, Ö. Kaya, and J. Rane, "Applications of machine learning in healthcare, finance, agriculture, retail, manufacturing, energy, and transportation: A review," in *Applied Machine Learning and Deep Learning: Architectures and Techniques*, Deep Science Publishing, 2024. doi: 10.70593/978-81-981271-4-3_6.

[2] K. Goyle, Q. Xie, and V. Goyle, "DataAssist: A Machine Learning Approach to Data Cleaning and Preparation," in *Intelligent Systems Conference*, Springer, Jul. 2023, pp. 476–486. doi: https://doi.org/10.48550/arXiv.2307.07119.

[3] P. Li, Z. Chen, X. Chu, and K. Rong, "DiffPrep: Differentiable Data Preprocessing Pipeline Search for Learning over Tabular Data," *Proceedings of the ACM on Management of Data*, vol. 1, no. 2, pp. 1–26, Jun. 2023, doi: 10.1145/3589328.

[4] J. Wang *et al.*, "IMWMOTE: A novel oversampling technique for fault diagnosis in heterogeneous imbalanced data," *Expert Syst Appl*, vol. 251, p. 123987, Oct. 2024, doi: 10.1016/j.eswa.2024.123987.

[5] S. Li *et al.*, "Rolling Bearing Fault Diagnosis Under Data Imbalance and Variable Speed Based on Adaptive Clustering Weighted Oversampling," *Reliab Eng Syst Saf*, vol. 244, p. 109938, Apr. 2024, doi: 10.1016/j.ress.2024.109938.

[6] P. Priyanka, N. Kumar, and D. Kumar, "Enhanced Grey Wolf Optimization based Hyper-parameter optimized Convolution Neural Network for Kidney Image Classification," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 11, no. 5, pp. 363–374, May 2023, doi: 10.17762/ijritcc.v11i5.6624.

[7] V. Švábenský, C. Borchers, E. B. Cloude, and A. Shimada, "Evaluating the Impact of Data Augmentation on Predictive Model Performance," in *Proceedings of the 15th International Learning Analytics and Knowledge Conference*, New York, NY, USA: ACM, Mar. 2025, pp. 126–136. doi: 10.1145/3706468.3706485.

[8] R. Mohakud and R. Dash, "Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6280–6291, Sep. 2022, doi: 10.1016/j.jksuci.2021.05.012.

[9] C. Chakraborty, A. Kishor, and J. J. P. C. Rodrigues, "Novel Enhanced-Grey Wolf Optimization hybrid machine learning technique for biomedical data computation," *Computers and Electrical Engineering*, vol. 99, p. 107778, Apr. 2022, doi: 10.1016/j.compeleceng.2022.107778.

[10] S. K. Parhi and S. K. Patro, "Prediction of compressive strength of geopolymer concrete using a hybrid ensemble of grey wolf optimized machine learning estimators," *Journal of Building Engineering*, vol. 71, p. 106521, Jul. 2023, doi: 10.1016/j.jobe.2023.106521.

[11] D. Yulvida and A. Saikhu, "Optimizing Adaptive Boosting Model for Breast Cancer Prediction Using Principal Component Analysis and Random Oversampling Techniques," in *2024 8th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, IEEE, Aug. 2024, pp. 173–178. doi: 10.1109/ICITISEE63424.2024.10730573.

[12] S. Quinevera and A. Saikhu, "Optimization of Classification Model for Early Detection of Pancreatic Cancer Using GridSearchCV and Autoencoder," in *2024 8th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, IEEE, Aug. 2024, pp. 133–137. doi: 10.1109/ICITISEE63424.2024.10730676.

[13] R. Mardianto and A. Saikhu, "Development of Pre-Processing for Chronic Kidney Disease Prediction Using K-Nearest Neighbors Imputer and Chi-Square," in *2024 8th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, IEEE, Aug. 2024, pp. 179–184. doi: 10.1109/ICITISEE63424.2024.10730259.

[14] Y. Gong, Q. Wu, M. Zhou, and C. Chen, "A diversity and reliability-enhanced synthetic minority oversampling technique for multi-label learning," *Inf Sci (N Y)*, vol. 690, p. 121579, Feb. 2025, doi: 10.1016/j.ins.2024.121579.

[15] J. Li, Q. Zhu, Q. Wu, and Z. Fan, "A novel oversampling technique for class-imbalanced learning based on SMOTE and natural neighbors," *Inf Sci (N Y)*, vol. 565, pp. 438–455, Jul. 2021, doi: 10.1016/j.ins.2021.03.041.

[16] E. D. Gennatas *et al.*, "Expert-augmented machine learning," *Proceedings of the National Academy of Sciences*, vol. 117, no. 9, pp. 4571–4577, Mar. 2020, doi: 10.1073/pnas.1906831117.

[17] E. Avuçlu, "A new data augmentation method to use in machine learning algorithms using statistical measurements," *Measurement*, vol. 180, p. 109577, Aug. 2021, doi: 10.1016/j.measurement.2021.109577.

[18] M. Mujahid *et al.*, "Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering," *J Big Data*, vol. 11, no. 1, p. 87, Jun. 2024, doi: 10.1186/s40537-024-00943-4.

[19] F. Shen, X. Zhao, G. Kou, and F. E. Alsaadi, "A new deep learning ensemble credit risk evaluation model with an improved synthetic minority oversampling technique," *Appl Soft Comput*, vol. 98, p. 106852, Jan. 2021, doi: 10.1016/j.asoc.2020.106852.

[20] A. A. Khan, O. Chaudhari, and R. Chandra, "A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation," *Expert Syst Appl*, vol. 244, p. 122778, Jun. 2024, doi: 10.1016/j.eswa.2023.122778.

[21] T. Suzuki, "TeachAugment: Data Augmentation Optimization Using Teacher Knowledge," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2022, pp. 10894–10904. doi: 10.1109/CVPR52688.2022.01063.

[22] F. J. Moreno-Barea, J. M. Jerez, and L. Franco, "Improving classification accuracy using data augmentation on small data sets," *Expert Syst Appl*, vol. 161, p. 113696, Dec. 2020, doi: 10.1016/j.eswa.2020.113696.

[23] M. Premkumar *et al.*, "Augmented weighted K-means grey wolf optimizer: An enhanced metaheuristic algorithm for data clustering problems," *Sci Rep*, vol. 14, no. 1, p. 5434, Mar. 2024, doi: 10.1038/s41598-024-55619-z.

[24] M. Zhang, X. Wang, L. Jin, M. Song, and Z. Li, "A new approach for evaluating node importance in complex networks via deep learning methods," *Neurocomputing*, vol. 497, pp. 13–27, Aug. 2022, doi: 10.1016/j.neucom.2022.05.010.

[25] M. Negassi, D. Wagner, and A. Reiterer, "Smart(Sampling)Augment: Optimal and Efficient Data Augmentation for Semantic Segmentation," *Algorithms*, vol. 15, no. 5, p. 165, May 2022, doi: 10.3390/a15050165.

[26] H. Hartono and E. Ongko, "Avoiding Overfitting dan Overlapping in Handling Class Imbalanced Using Hybrid Approach with Smoothed Bootstrap Resampling and Feature Selection," *JOIV : International Journal on Informatics Visualization*, vol. 6, no. 2, p. 343, Jun. 2022, doi: 10.30630/joiv.6.2.985.

[27] A. Mumuni and F. Mumuni, "Data augmentation: A comprehensive survey of modern approaches," *Array*, vol. 16, p. 100258, Dec. 2022, doi: 10.1016/j.array.2022.100258.

[28] B. Xiong, X. Zhao, Y. Hu, H. Huang, Y. Liu, and Y. Su, "Machine learning assisted empirical formula augmentation," *Mater Des*, vol. 210, p. 110037, Nov. 2021, doi: 10.1016/j.matdes.2021.110037.

[29] P. Mathur, H. Shaikh, F. Sheth, D. Kumar, and A. K. Gupta, "A Computational Intelligence Framework Integrating Data Augmentation and Meta-Heuristic Optimization Algorithms for Enhanced Hybrid Nanofluid Density Prediction Through Machine and Deep Learning Paradigms," *IEEE Access*, vol. 13, pp. 35750–35779, 2025, doi: 10.1109/ACCESS.2025.3543475.

[30] B. Hettwer, "Deep learning-enhanced side-channel analysis of cryptographic implementations," Dissertation, Ruhr-Universität Bochum, Bochum, 2020.

[31] H. Arslan and H. Arslan, "A new COVID-19 detection method from human genome sequences using CpG island features and KNN classifier," *Engineering Science and Technology, an International Journal*, vol. 24, no. 4, pp. 839–847, Aug. 2021, doi: 10.1016/j.jestch.2020.12.026.

[32] J. Gou, W. Qiu, Z. Yi, X. Shen, Y. Zhan, and W. Ou, "Locality constrained representation-based K-nearest neighbor classification," *Knowl Based Syst*, vol. 167, pp. 38–52, Mar. 2019, doi: 10.1016/j.knosys.2019.01.016.

[33] J. Gou, H. Ma, W. Ou, S. Zeng, Y. Rao, and H. Yang, "A generalized mean distance-based k-nearest neighbor classifier," *Expert Syst Appl*, vol. 115, pp. 356–372, Jan. 2019, doi: 10.1016/j.eswa.2018.08.021.

[34] H. Tyralis and G. Papacharalampous, "A review of predictive uncertainty estimation with machine learning," *Artif Intell Rev*, vol. 57, no. 4, p. 94, Mar. 2024, doi: 10.1007/s10462-023-10698-8.

[35] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., in Proceedings of Machine Learning Research, vol. 70. PMLR, May 2017, pp. 1321–1330. [Online]. Available: https://proceedings.mlr.press/v70/guo17a.html

[36] U. Johansson, C. Sonstrod, T. Lofstrom, and H. Bostrom, "Confidence Classifiers with Guaranteed Accuracy or Precision," in *Proceedings of the Twelfth Symposium on Conformal and Probabilistic Prediction with Applications*, H. Papadopoulos, K. A. Nguyen, H. Boström, and L. Carlsson, Eds., in Proceedings of Machine Learning Research, vol. 204. PMLR, May 2023, pp. 513–533. [Online]. Available: https://proceedings.mlr.press/v204/johansson23a.html

[37] G. F. Shidik *et al.*, "Bird Species Classification Enhancement via Adaptive Inertia Weight Particle Swarm Optimization-Based Image Augmentation Selection," *IEEE Access*, vol. 12, pp. 197048–197060, 2024, doi: 10.1109/ACCESS.2024.3521455.

[38] F. Zhao, F. Ji, T. Xu, N. Zhu, and Jonrinaldi, "Hierarchical parallel search with automatic parameter configuration for particle swarm optimization," *Appl Soft Comput*, vol. 151, p. 111126, Jan. 2024, doi: 10.1016/j.asoc.2023.111126.

[39] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[40] S. N. Makhadmeh *et al.*, "Recent Advances in Grey Wolf Optimizer, its Versions and Applications: Review," *IEEE Access*, vol. 12, pp. 22991–23028, 2024, doi: 10.1109/ACCESS.2023.3304889.

[41] Y. Shen *et al.*, "Improved differential evolution algorithm based on cooperative multi-population," *Eng Appl Artif Intell*, vol. 133, p. 108149, Jul. 2024, doi: 10.1016/j.engappai.2024.108149.

[42] W. Deng, S. Shang, X. Cai, H. Zhao, Y. Song, and J. Xu, "An improved differential evolution algorithm and its application in optimization problem," *Soft comput*, vol. 25, no. 7, pp. 5277–5298, Apr. 2021, doi: 10.1007/s00500-020-05527-x.

[43] K. A. Alnowibet, S. Shekhawat, A. Saxena, K. M. Sallam, and A. W. Mohamed, "Development and Applications of Augmented Whale Optimization Algorithm," *Mathematics*, vol. 10, no. 12, p. 2076, Jun. 2022, doi: 10.3390/math10122076.

[44] M. Braik, "Hybrid enhanced whale optimization algorithm for contrast and detail enhancement of color images," *Cluster Comput*, vol. 27, no. 1, pp. 231–267, Feb. 2024, doi: 10.1007/s10586-022-03920-9.

[45] K. Meidani, A. Hemmasian, S. Mirjalili, and A. Barati Farimani, "Adaptive grey wolf optimizer," *Neural Comput Appl*, vol. 34, no. 10, pp. 7711–7731, May 2022, doi: 10.1007/s00521-021-06885-9.

[46] J. Yin and N. Li, "Ensemble learning models with a Bayesian optimization algorithm for mineral prospectivity mapping," *Ore Geol Rev*, vol. 145, p. 104916, Jun. 2022, doi: 10.1016/j.oregeorev.2022.104916.

[47] J. L. Potharlanka and N. B. M, "Feature importance feedback with Deep Q process in ensemble-based metaheuristic feature selection algorithms," *Sci Rep*, vol. 14, no. 1, p. 2923, Feb. 2024, doi: 10.1038/s41598-024-53141-w.

[48] T. Shaqarin and B. R. Noack, "A Fast-Converging Particle Swarm Optimization through Targeted, Position-Mutated, Elitism (PSO-TPME)," *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, p. 6, Jan. 2023, doi: 10.1007/s44196-023-00183-z.

[49] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[50] M. H. Nadimi-Shahraki, H. Zamani, Z. Asghari Varzaneh, and S. Mirjalili, "A systematic review of the whale optimization algorithm: theoretical foundation, improvements, and hybridizations," *Archives of Computational Methods in Engineering*, vol. 30, no. 7, pp. 4113–4159, 2023.

[51] M. Chen, C. Feng, and R. Cheng, "Metade: Evolving differential evolution by differential evolution," *IEEE Transactions on Evolutionary Computation*, 2025.

[52] J. A. Cook and J. Ranstam, "Overfitting," *British Journal of Surgery*, vol. 103, no. 13, pp. 1814–1814, Nov. 2016, doi: 10.1002/bjs.10244.

[53] H. Li, G. K. Rajbahadur, D. Lin, C.-P. Bezemer, and Z. M. Jiang, "Keeping Deep Learning Models in Check: A History-Based Approach to Mitigate Overfitting," *IEEE Access*, vol. 12, pp. 70676–70689, 2024, doi: 10.1109/ACCESS.2024.3402543.

[54] C. Aliferis and G. Simon, "Overfitting, Underfitting and General Model Overconfidence and Under-Performance Pitfalls and Best Practices in Machine Learning and AI," 2024, pp. 477–524. doi: 10.1007/978-3-031-39355-6_10.

[55] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA: ACM, Jul. 2019, pp. 2623–2631. doi: 10.1145/3292500.3330701.

[56] S. Dhanka and S. Maini, "A hybridization of XGBoost machine learning model by Optuna hyperparameter tuning suite for cardiovascular disease classification with significant effect of outliers and heterogeneous training datasets," *Int J Cardiol*, vol. 420, p. 132757, Feb. 2025, doi: 10.1016/j.ijcard.2024.132757.

[57] P. Srinivas and R. Katarya, "hyOPTXg: OPTUNA hyper-parameter optimization framework for predicting cardiovascular disease using XGBoost," *Biomed Signal Process Control*, vol. 73, p. 103456, Mar. 2022, doi: 10.1016/j.bspc.2021.103456.

[58] A. K. Gupta, P. Mathur, F. Sheth, C. M. Travieso-Gonzalez, and S. Chaurasia, "Advancing geological image segmentation: Deep learning approaches for rock type identification and classification," *Applied Computing and Geosciences*, vol. 23, p. 100192, Sep. 2024, doi: 10.1016/j.acags.2024.100192.

[59] F. Sheth, P. Mathur, A. K. Gupta, and S. Chaurasia, "An advanced artificial intelligence framework integrating ensembled convolutional neural networks and Vision Transformers for precise soil classification with adaptive fuzzy logic-based crop recommendations," *Eng Appl Artif Intell*, vol. 158, p. 111425, Oct. 2025, doi: 10.1016/j.engappai.2025.111425.

[60] R. Sarno, K. Triyana, S. I. Sabilla, D. R. Wijaya, D. Sunaryono, and C. Fatichah, "Detecting Pork Adulteration in Beef for Halal Authentication Using an Optimized Electronic Nose System," *IEEE Access*, vol. 8, pp. 221700–221711, 2020, doi: 10.1109/ACCESS.2020.3043394.

[61] D. R. Wijaya, R. Sarno, and E. Zulaika, "DWTLSTM for electronic nose signal processing in beef quality monitoring," *Sens Actuators B Chem*, vol. 326, p. 128931, Jan. 2021, doi: 10.1016/j.snb.2020.128931.

[62] Scikit-learn, "KNeighborsClassifier." Accessed: Aug. 08, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

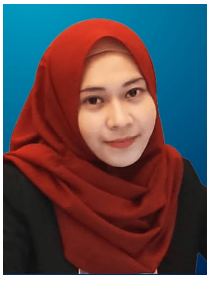
[63] A. Naimi, J. Deng, S. R. Shimjith, and A. J. Arul, "Fault Detection and Isolation of a Pressurized Water Reactor Based on Neural Network and K-Nearest Neighbor," *IEEE Access*, vol. 10, pp. 17113–17121, 2022, doi: 10.1109/ACCESS.2022.3149772.

[64] G. Muhammad *et al.*, "Enhancing Prognosis Accuracy for Ischemic Cardiovascular Disease Using K Nearest Neighbor Algorithm: A Robust Approach," *IEEE Access*, vol. 11, pp. 97879–97895, 2023, doi: 10.1109/ACCESS.2023.3312046.

[65] X. Han, H. Zheng, and M. Zhou, "CARD: Classification and Regression Diffusion Models," Jun. 2022, [Online]. Available: http://arxiv.org/abs/2206.07275

DWI SUNARYONO (Member, IEEE) was born in Surabaya, in May 1972. He received the bachelor's degree in computer engineering, the master's and doctoral degree in informatics engineering from the Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember (ITS), in 1995, 2009, 2024, respectively. He is currently a Lecturer with the Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, ITS. His research interests include face recognition, signal processing, the Internet of Things, intelligent systems, and management information.

RIYANARTO SARNO (Member, IEEE) received the Doctor (Ph.D.) degree, in 1992. He is currently a Professor with the Informatics Department, Institut Teknologi Sepuluh Nopember (ITS). He is interested in research projects about machine learning, the Internet of Things, and knowledge engineering. He has researched E-nose for a period of five years. Being an author of more than five books and over 300 scientific articles led him incorporated in the top 2% world ranking scientist, in 2020, by Standford University.



SHOFFI IZZA SABILLA (Member, IEEE) received the Doctoral. degree in informatics engineering from the Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember (ITS) in 2021. Currently, she is a lecturer in department of medical technology at the Institut Teknologi Sepuluh Nopember (ITS). Her academic and research interests are primarily focused on the intersection of machine learning, optimization, and medical technology. Over the past three years, she has conducted extensive research on E-nose technology, exploring its applications in various fields such as healthcare, diagnostics, and environmental monitoring. Her expertise in machine learning and optimization has significantly contributed to the advancement of these areas, particularly in the development of intelligent systems and sensors for medical and environmental purposes.

RIZQY AHSANA PUTRI was born in Gresik, Jawa Timur, Indonesia, in 2001. She received the master's degree in informatics from Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia, in 2025. She is currently pursuing a doctoral degree in the informatics department, Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. She received a scholarship from the Ministry of Research, Technology, and Higher Education for batch 7 in the PMDSU program. Her research interests include artificial intelligence, machine learning, and signal processing.

TAUFIQ CHOIRUL AMRI was born in Banyuwangi, Jawa Timur, Indonesia. He received a master's degree in Informatics from Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. He is currently pursuing the doctoral degree in Computer Science at the Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. His research interests include Internet of Things, machine learning, artificial intelligence, and signal processing.

IRFAN MIRDA was born in Mataram Baru Lampung Timur, Lampung, in 2002. He received his bachelor's degree in electrical engineering from Universitas Lampung, Indonesia, in 2024. He is currently pursuing a master's degree in the informatics department at Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia. He received a scholarship from the Ministry of Research, Technology, and Higher Education for batch 8 in the PMDSU program. His research interests include Internet of Things, data science, artificial intelligence, and robotics.

JOKO SISWANTORO received the B.Sc. degree in Mathematics from Airlangga University, Surabaya, Indonesia, in 1997, the M.Sc. degree in Applied Mathematics from Institut Teknologi Bandung, Bandung, Indonesia, in 2002, and the Ph.D. degree in Computer Science from Universiti Kebangsaan Malaysia, Bangi, Malaysia, in 2016. He is currently an Associate Professor and Head of the Department of Informatics Engineering at the University of Surabaya, Surabaya, Indonesia. His research interests include data science, machine learning, digital image processing, computer vision, and optimization.

Original Manuscript ID: Access-2025-25851

Original Article Title: "Augmented Data Low Confidence (ADLC): A Confidence-Driven Data Augmentation Framework with Ensemble Optimization for Enhanced Machine Learning Performance"

To: IEEE Access Editor

Re: Response to reviewers

Dear Editor,

Thank you for allowing the resubmission of our manuscript, with an opportunity to address the reviewers' comments.

We are uploading (a) our point-by-point response to the comments (below) (response to reviewers), (b) an updated manuscript with yellow highlighting indicating changes, (c) a clean updated manuscript without highlights (PDF main document), and (d) a request for byline change.

Best regards,

Dr. Dwi Sunaryono et al.

Reviewer #1, Concern #1:
Code Availability (Table III): Table III appears to be unnecessary. Instead, the authors are encouraged to provide their implementation code via a publicly accessible repository (e.g., GitHub) and include a Code Availability section in the manuscript, with the appropriate link for reproducibility.

Author response:
Thank you for the constructive comment. We agree with your suggestion. To improve reproducibility and transparency of our work, we have removed Table III and provided the complete implementation code in a publicly accessible GitHub repository.

Author action:
We updated the manuscript by adding a Code Availability statement in **Section 2.A, last paragraph, last sentence**:

"In addition to ensuring reproducibility, the complete source code used for data processing, model training, and metaheuristic algorithm implementation is made publicly available in a GitHub repository (https://github.com/Irfanmrd12/ADLC)."

Reviewer #1, Concern #2:
Justification for Optimization Methods and Ensembling Technique: The selection criteria for the optimization methods should be clearly justified—especially if they depend on a specific variable such as rrr. Additionally, specify whether the ensemble strategy used is soft voting, hard voting, or another method, and explain the rationale for the choice.

Author response:
Thank you for the valuable comment. We have clarified the justification for using the probability variable *r* as well as the rationale behind the selection of optimization methods. We also elaborated on the ensemble mechanism used in this study, including the type of ensemble strategy and why it was selected over alternatives.

Author action:
We have revised the manuscript in **Section 2.E.6**, Paragraphs 6–7, to provide a detailed justification for the variable rrr in each optimization algorithm and a clearer explanation of the ensemble mechanism used in this study.

"This study adopts an ensemble-based metaheuristic optimization [47] approach by integrating Bayesian optimization (via Optuna) with a set of metaheuristic algorithms—PSO, GWO, WOA, and DE. The ensemble employs predetermined selection probabilities to guide the use of each algorithm, enabling layered optimization that adapts effectively to varying dataset complexities. The proposed PSO_GWO_DE_AP_WOA Ensemble thus combines four powerful metaheuristic methods and adaptive parameter tuning to improve convergence, solution quality, and computational efficiency.

The ensemble mechanism randomly selects one of the four optimization strategies for each agent based on a predefined selection probability *r*. This mechanism allows each agent to update its position using a different algorithm, enhancing the diversity and adaptability of the optimization process. The choice of probability *r* is justified by its role as an adaptive selection mechanism that balances exploration and exploitation across different phases of the search. The range of *r* is divided into four intervals so that each algorithm has a fair yet adaptive allocation.

PSO is assigned to lower *r* values because it is effective in early global exploration and demonstrates fast convergence in initial iterations [48]. GWO occupies the lower-middle range, stabilizing the process through adaptive social hierarchy and encircling mechanisms that balance exploration and exploitation [49]. WOA is positioned in the upper-middle interval, where it exploits promising regions while maintaining population diversity as its exploration–exploitation ratio naturally shifts during iterations [50]. Finally, DE is assigned to higher r values due to its strength in intensive exploitation and fine-tuning, which refines solutions in the final

stages of optimization [51]. This balanced allocation ensures efficient convergence, reduces the risk of overfitting, and improves the stability of solutions across diverse datasets.

The selection of the interval for the *r* value in each optimization method is based on a series of comprehensive experiments with systematically tested parameter configurations. Five different interval configurations are evaluated using four machine learning models, namely SVC, KNN, Decision Tree, and Random Forest, as follows:

1. PSO [0,0.25), GWO [0.25,0.5), WOA [0.5,0.75), DE [0.75,1]
2. PSO [0,0.1), GWO [0.1,0.5), WOA [0.5,0.6), DE [0.6,1]
3. PSO [0,0.33), GWO [0.33,0.66), WOA [0.66,0.8), DE [0.88,1]
4. PSO [0,0.2), GWO [0.2,0.55), WOA [0.55,0.8), DE [0.8,1]
5. PSO [0,0.4), GWO [0.4,0.5), WOA [0.5,0.9), DE [0.9,1]

Based on the evaluation of all configurations, the configuration that yields the best performance is PSO [0,0.33), GWO [0.33,0.66), WOA [0.66,0.8), and DE [0.88,1]. This configuration achieves an average accuracy of 92.47%, precision of 92.65%, recall of 92.47%, F1-score of 92.33%, and convergence at iteration 26.45, outperforming the other configurations.

Therefore, the selection of the optimization method in the ensemble metaheuristic optimization is as follows:

- If r <0.33, the PSO strategy is used.
- If 0.33≤ r <0.660, the GWO strategy is selected.
- If 0.66≤ r <0.80, the WOA strategy is applied.
- If r ≥0.80, the DE strategy is used."

Reviewer #1, Concern #3:

Overfitting and Validation Strategy: The manuscript mentions the use of L2 regularization and early stopping. Please also provide the learning rate and decay settings used in the training process. Moreover, although K-fold cross-validation is employed, some models report 100% accuracy, which is indicative of potential overfitting. This issue must be addressed and justified clearly in the discussion.

Author response:

Thank you for this important observation. Regarding the first point, our study does not employ models based on gradient descent (e.g., neural networks, boosting methods such as XGBoost/LightGBM, or deep learning). Instead, we only use **kNN, Decision Tree, and SVC**, which do not involve hyperparameters such as learning rate and decay. Similarly, L2 regularization is a hyperparameter of logistic regression, which is also not part of our model set. Therefore, these details cannot be reported in our manuscript.

To prevent underfitting and overfitting, our work employs **Optuna's pruning mechanism as an early stopping strategy** during hyperparameter search. This method halts unpromising trials early and prevents the optimization process from overfitting to noise within specific folds. This clarification has been added in **Section 2.F, Paragraphs 2–3.**

Concerning the second point about 100% accuracy in some datasets, we have clarified in the **Discussion (Section 4.E, Paragraph 8)** that these results are consistent with the inherent characteristics of the datasets rather than overfitting. Datasets such as Iris, Wine, and Diabetes are relatively small and easily separable, with baseline accuracies already very high (e.g., Iris 98.67%, Wine 96.63%, Diabetes 98.43%). Our method only corrected a few remaining misclassified samples, leading to perfect accuracy. The gains remain statistically consistent across datasets and fall within the expected confidence interval, indicating that these results are not artifacts of overfitting.

Author action:

We revised the manuscript as follows:

In **Section 2.F, Paragraphs 2–3**, we clarified the role of early stopping via Optuna pruning:

"To address both overfitting and underfitting, this study employs early stopping as the primary control mechanism, including the use of Optuna's pruning mechanism as an early stopping strategy during hyperparameter search. Early stopping halts the training or optimization process once the performance metric reaches a predefined threshold, preventing memorization of the training data or wasteful redundant updates. This ensures that the resulting models remain both accurate and efficient."

In **Section 4.E, Paragraph 8**, we expanded the explanation regarding perfect accuracy results:

"It should be noted that several datasets achieved 100% accuracy after optimization (e.g., Iris, Wine, and Diabetes). This result does not necessarily indicate overfitting but rather reflects the relatively small size and clear separability of these datasets. For instance, in the Iris dataset, accuracy improved from 98.67% to 100% (correcting only two samples). Likewise, the Wine dataset increased from 96.63% to 100% (around six samples corrected), and Diabetes improved from 98.43% to 100%. Importantly, these perfect accuracies remain consistent with the overall improvement distribution (average gain +3.99%, σ = 4.48%), well within the 95% confidence interval (−4.78% to +12.78%). Therefore, the results should be regarded as natural outcomes of the dataset characteristics rather than evidence of overfitting."

Reviewer #1, Concern #4:
Data Augmentation Parameters (min_aug=0.1, max_aug=0.5): Justify the selection of these augmentation bounds. Why were these specific values chosen? Were they derived from prior work, empirical tuning, or theoretical considerations?

Author response:
Thank you for raising this concern. The augmentation bounds were determined based on **empirical tuning** to strike a balance between insufficient augmentation (which would not effectively increase data diversity) and excessive augmentation (which could introduce noise and distort the original distribution).

Author action:
We clarified this in **Section 2.G, Paragraph 3**, as follows:

"In this study, the augmentation ratio is controlled using a Dynamic Augmentation Adjustment strategy, which operates within a predefined range bounded by min_augment_ratio and max_augment_ratio, as in Algorithm 1 and Algorithm 2. These boundaries serve as safeguards to prevent under- or over-augmentation. Based on empirical tuning, the values are set by default to 0.1 and 0.5, respectively. A minimum value of 0.1 ensures that even well-performing models still receive a sufficient degree of augmentation to enhance data diversity without significantly altering the original distribution. Meanwhile, the maximum value of 0.5 is chosen to avoid excessive augmentation that could introduce noise and degrade the quality of the original data."

Reviewer #1, Concern #5:
Use of k-NN for Augmented Data Generation: The manuscript states that k-NN was used for generating augmented data. This is unconventional—please provide a detailed justification for using k-NN in this context, and explain its role in the augmentation process.

Author response:
Thank you for pointing out this issue. We would like to clarify that **k-NN is not used to generate new data**, but rather to **identify data samples with low confidence** based on class probability outputs. These low-

confidence samples are then selected for augmentation. The choice of k-NN is not exclusive—other models could also be used for this purpose. However, we selected k-NN because of its **simplicity, computational efficiency, and broad applicability** across diverse datasets.

Author action:
We revised the manuscript in **Section 2.K, Paragraph 2**, to explicitly clarify this point:

"The K-Nearest Neighbors (kNN) algorithm is used—not for generating new data, but to identify which data points are classified with low confidence based on class probability outputs. These low-confidence samples are subsequently targeted for augmentation. Although other models could also be employed for this task, the k-NN algorithm is selected in this research due to its simplicity and computational efficiency, making it a practical and widely applicable approach across diverse applications."

Reviewer #1, Concern #6:
Optimal Model and Position (Figures 2 and 3): Clarify what parameters or evaluation metrics were used to determine "optimality" in these figures. Were accuracy, loss, confidence thresholds, or any other metrics considered? State the specific criteria or conditions used for determining the optimal model and its position.

Author response:
Thank you for your valuable comment. The evaluation metric used to determine the optimal model is accuracy. A model is considered optimal once it reaches 100% accuracy during the training process. This accuracy threshold serves as the primary stopping criterion. To further validate the model's robustness and generalization capability, the final evaluation is performed on the test dataset after the optimal point is reached.

Author action:
We clarified this in **Section 2.K, Paragraph 5**, as follows:
"To determine whether a model is considered optimal, accuracy is used as the primary evaluation metric. The optimization process concludes when the model reaches an accuracy threshold of 1.0 (100%), which serves as the main stopping criterion indicating that the model is fully optimal. Once the optimal model is established, a final test is conducted using the test dataset to validate its generalization ability and robustness on new data."

Reviewer #1, Concern #7:
Confidence Thresholds (U1 = 0.6, U2 = 0.7, U3 = 0.8, U4 = 0.9): Provide a theoretical or empirical basis for choosing these specific threshold values. Were they derived through experimentation, prior research, or heuristics?

Author response:
Thank you for this insightful question. The selection of confidence thresholds was based on empirical experiments. We systematically tested different values to observe how the effectiveness of data augmentation varied under different confidence levels. The chosen thresholds (0.6, 0.7, 0.8, and 0.9) provided a balanced range to evaluate model uncertainty while ensuring that augmentation was applied meaningfully to low-confidence cases.

Author action:
We clarified this in Section 3.B, Paragraph 3, as follows:
"These values were chosen based on empirical experiments that examined how the effectiveness of data

augmentation varies depending on the model's confidence levels. The thresholds of 0.6, 0.7, 0.8, and 0.9 were selected to provide a balanced spectrum for capturing different degrees of model uncertainty and to ensure that augmentation targets samples with lower classification confidence."

Reviewer #1, Concern #8:
Model Selection Justification (SVC, DT, k-NN): The selection of Support Vector Classifier (SVC), Decision Tree (DT), and k-Nearest Neighbors (k-NN) for validation of the proposed method should be explicitly justified. Why were these particular classifiers chosen over others?

Author response:
Thank you for this valuable comment. The selection of SVC, DT, and k-NN was intentional because these three algorithms represent fundamental yet diverse machine learning approaches. Each classifier embodies a different learning mechanism—hyperplane separation (SVC), hierarchical decision rules (DT), and instance-based classification (k-NN). Using this combination allows us to demonstrate how the proposed method performs across distinct paradigms rather than being tied to a single family of models.

Author action:
We clarified this in Section 2.K, Paragraph 4, as follows:
"The next stage is training using ML algorithms such as Support Vector Classifier (SVC), Decision Tree (DT), and KNN. These three models are selected because they represent three fundamental approaches in machine learning, each with a distinct learning mechanism. SVC classifies data by finding the optimal hyperplane that separates classes with the maximum margin. DT constructs a decision structure in the form of a tree, where each node represents a condition leading to a classification outcome. Meanwhile, KNN classifies samples based on the majority class of their nearest neighbors [64]. These differences provide a broad view of how various learning strategies respond to the optimization process and dataset characteristics."

Reviewer #1, Concern #9:
Use of PCA (Figure 4): Clarify whether Principal Component Analysis (PCA) was used in Figure 4. What was the objective of applying PCA—dimensionality reduction, visualization, or something else? A brief explanation about PCA and its relevance in the context of this work should be provided.

Author response:
Thank you for highlighting this point. PCA was indeed applied in Figure 4 with the purpose of dimensionality reduction for visualization. Since the diabetes dataset originally contained 14 features, we used PCA to project the data into two principal components, making it possible to visualize the distribution and classification boundaries in a two-dimensional space. This helps readers more easily interpret the results while preserving the most significant variance in the data.

Author action:
We clarified this in Section 3.E, Paragraph 3, as follows:
"Figure 4 presents a comparison of the performance of various optimization algorithms, including PGDAWE, an ensemble method that leverages the strengths of multiple optimization techniques on diabetes dataset. The dataset originally consisted of fourteen features. To simplify interpretation and enable two-dimensional visualization, Principal Component Analysis (PCA) is applied to reduce the feature space to two principal components."

Reviewer #1, Concern #10:
Relevant Literature and Citations: The authors are encouraged to cite recent and relevant works that align with the proposed methodology, such as:
[1] https://doi.org/10.1016/j.engappai.2025.111425
[2] https://doi.org/10.1109/ACCESS.2025.3543475
[3] https://doi.org/10.1016/j.acags.2024.100192

Author response:
Thank you for this helpful suggestion. We have added the recommended recent and relevant works to strengthen the literature review and to demonstrate alignment with current research in the field. These references were integrated into Section 2.I, Paragraph 1, where we discuss the importance of data preprocessing and its role in ensuring high-quality input for the modeling process.

Author action:
We updated Section 2.B, Paragraph 1, Section 2.I, Paragraph 1, and the reference list as follows:

“Data augmentation is a critical technique in ML, aimed at increasing the volume, quality, and diversity of training data. By generating new data points through various transformations or by incorporating information from multiple sources, data augmentation enriches the original dataset [27]. This process not only expands the dataset size, an essential factor for training robust models, but also introduces variability that enhances model generalization, particularly in cases where the original data is limited or imbalanced [28], [29]. Furthermore, data augmentation improves the informativeness of training samples, making them more valuable for the learning process [27].”

“Data preprocessing is a crucial component of the strategy to ensure data quality before use in the modelling process [58], [59]. This step aims to eliminate unreliable data, reduce the risk of delays in the modelling process, address data inaccuracies, and minimize suboptimal model outcomes [60]. Preprocessing encompasses various techniques such as data cleaning, segmentation, and scale adjustment and normalization.”

“[29] P. Mathur, H. Shaikh, F. Sheth, D. Kumar, and A. K. Gupta, “A Computational Intelligence Framework Integrating Data Augmentation and Meta-Heuristic Optimization Algorithms for Enhanced Hybrid Nanofluid Density Prediction Through Machine and Deep Learning Paradigms,” *IEEE Access*, vol. 13, pp. 35750–35779, 2025, doi: 10.1109/ACCESS.2025.3543475.”

“[58] A. K. Gupta, P. Mathur, F. Sheth, C. M. Travieso-Gonzalez, and S. Chaurasia, “Advancing geological image segmentation: Deep learning approaches for rock type identification and classification,” *Applied Computing and Geosciences*, vol. 23, p. 100192, Sep. 2024, doi: 10.1016/j.acags.2024.100192.
[59] F. Sheth, P. Mathur, A. K. Gupta, and S. Chaurasia, “An advanced artificial intelligence framework integrating ensembled convolutional neural networks and Vision Transformers for precise soil classification with adaptive fuzzy logic-based crop recommendations,” *Engineering Applications of Artificial Intelligence*, vol. 158, p. 111425, Oct. 2025, doi: 10.1016/j.engappai.2025.111425.”

Reviewer #2, Concern #1:
The paper briefly mentions generating new data by "randomly sampling the upper and lower limits of eigenvalues" and "adding Gaussian noise" for low-confidence instances. However, the implementation details of these strategies are insufficient—especially for specific data types like images and EEG signals. The methodology should clarify, for example, how Gaussian noise is applied to image pixels or signal time points, and how the augmented samples preserve semantic integrity and correct labeling.

Author response:
Thank you for this comment. We have clarified the implementation details of the augmentation strategies for both image and EEG data. Specifically, for image data, Gaussian noise is applied at the pixel level after flattening the image into feature vectors. For EEG data, augmentation is performed after feature extraction using the Discrete Wavelet Transform (DWT), ensuring that Gaussian noise is added only to the extracted features rather than raw signals, thereby maintaining semantic consistency and correct labeling.

Author action:
We revised the manuscript in Section 2.K, Paragraph 3, as follows:

"Once the low-confidence samples are identified, the data augmentation process begins. This is carried out by introducing Gaussian noise to those specific samples, enhancing their diversity while preserving core features. The application of Gaussian noise depends on the type of data. In numerical datasets, noise is added to each feature value by sampling from a Gaussian distribution. In the case of image data, the Digits dataset is used, where each image is originally in an 8×8 format and is flattened into a 64-element feature vector. Gaussian noise is then applied to each pixel value within this flattened representation. For EEG signal data, the process begins with feature extraction using the Discrete Wavelet Transform (DWT). This involves decomposing the raw EEG signal into several wavelet coefficients using the 'sym2' wavelet at level 3. From each set of coefficients, the mean and standard deviation are calculated to form the final feature vector. After the DWT-based features are extracted, Gaussian noise is added to each feature value. The augmentation is further refined using the PGDAWE optimization technique, which controls the level of noise added to maintain data quality and avoid unnecessary distortion. The newly augmented samples are then combined with the original dataset, resulting in a richer and more informative training set that aims to improve model accuracy."

Reviewer #2, Concern #2:
The initial experiment identifies T=0.8 and N_a=10 as optimal for the Iris dataset. Whether these parameters were held constant across the remaining 11 datasets or independently tuned remains unclear. The absence of dataset-specific or adaptive tuning undermines the framework's claimed generalizability.

Author response:
Thank you for pointing this out. We confirm that the parameters N_a=10 and T=0.8, obtained from the initial experiment on the Iris dataset, were consistently applied across all subsequent datasets. This decision was made to maintain evaluation consistency and enable a fair comparison across datasets. Although dataset-specific tuning might yield slightly better performance, fixing the parameters allows us to demonstrate the robustness and generalizability of the proposed framework under uniform experimental conditions.

Author action:
We clarified this in Section 3.B, last paragraph, as follows:

"Based on the results presented in Tables V and VI, the first experimental scenario using the Iris dataset indicates that the optimal parameters are N_a=10 and T=0.8. These parameter values will be consistently applied in subsequent experiments on other datasets to ensure comparability and maintain consistency in the evaluation process."

Reviewer #2, Concern #3:
The paper notes that kNN estimates prediction confidence but does not explain how this is done in multi-class scenarios. For instance, how are class probabilities derived from kNN neighbors? Moreover, there is

little explanation of how other models like SVC or DT produce confidence scores, despite these being integral to the framework.

Author response:
Thank you for this valuable comment. We have clarified how prediction confidence is calculated for kNN, SVC, and DT. For kNN in multiclass classification, confidence is derived from the proportion of neighbors belonging to each class, with the highest proportion used as the confidence score. For SVC and DT, the built-in probability estimation methods are used. Specifically, SVC employs the Platt scaling method to convert decision function outputs into probabilities, while DT calculates class probabilities based on the distribution of training samples within terminal nodes.

Author action:
We updated Section 2.K, Paragraph 1 (Sentences 6–8) and Paragraph 2, as follows:

"For multiclass classification, the algorithm calculates the proportion of neighbors that belong to each class [62]. The highest proportion becomes the predicted class, while the value of that proportion is interpreted as the confidence score. For example, if among 10 nearest neighbors, 6 belong to class A, 3 to class B, and 1 to class C, the model assigns class A with a confidence of 0.6."

"K-Nearest Neighbors (kNN) algorithm is used—not for generating new data, but to identify which data points are classified with low confidence based on class probability outputs. In addition to kNN, confidence level can also be calculated using other algorithms such as SVC and Decision Tree. SVC determines prediction confidence based on the distance to the hyperplane, which can be calibrated into probabilities using Platt scaling. Decision Tree, on the other hand, determines it from leaf purity or the proportion of samples belonging to each class in the leaf where the instance falls. The k-NN algorithm is selected in this research due to its simplicity and computational efficiency, making it a practical and widely applicable approach across diverse applications [63]."

Reviewer #2, Concern #4:
While the role of adaptive parameters in the ensemble process is acknowledged, the paper does not specify how these parameters interact with individual optimization methods such as PSO, WOA, or DE. Additionally, the fixed selection probabilities (e.g., $r < 0.33$ for PSO) lack justification and should be discussed regarding whether adaptive mechanisms were considered or could be beneficial.

Author response:
Thank you for this insightful comment. We have expanded the explanation of how the probability variable *r* interacts with individual optimization algorithms. The role of *r* is to function as an adaptive selection mechanism that balances exploration and exploitation at different phases of the search. Each algorithm is allocated to a specific interval of *r* according to its strength: PSO for early exploration and fast convergence, GWO for balancing exploration and exploitation through adaptive hierarchies, WOA for mid-phase localized refinement while preserving diversity, and DE for intensive exploitation and fine-tuning in the final stage. Furthermore, We have clarified the justification for using the probability variable *r* as well as the rationale behind the selection of optimization methods.

Author action:
We clarified this in Section 2.E, Subsection 6, Paragraphs 3–6, as follows:

"The ensemble mechanism randomly selects one of the four optimization strategies for each agent based on a predefined selection probability *r*. This mechanism allows each agent to update its position using a different algorithm, enhancing the diversity and adaptability of the optimization process. The choice of probability *r* is justified by its role as an adaptive selection mechanism that balances exploration and

exploitation across different phases of the search. The range of *r* is divided into four intervals so that each algorithm has a fair yet adaptive allocation.

PSO is assigned to lower *r* values because it is effective in early global exploration and demonstrates fast convergence in initial iterations [48]. GWO occupies the lower-middle range, stabilizing the process through adaptive social hierarchy and encircling mechanisms that balance exploration and exploitation [49]. WOA is positioned in the upper-middle interval, where it exploits promising regions while maintaining population diversity as its exploration–exploitation ratio naturally shifts during iterations [50]. Finally, DE is assigned to higher *r* values due to its strength in intensive exploitation and fine-tuning, which refines solutions in the final stages of optimization [51]. This balanced allocation ensures efficient convergence, reduces the risk of overfitting, and improves the stability of solutions across diverse datasets.

The selection of the interval for the *r* value in each optimization method is based on a series of comprehensive experiments with systematically tested parameter configurations. Five different interval configurations are evaluated using four machine learning models, namely SVC, KNN, Decision Tree, and Random Forest, as follows:

6. PSO [0,0.25), GWO[0.25,0.5), WOA [0.5,0.75), DE [0.75,1]
7. PSO [0,0.1), GWO [0.1,0.5), WOA [0.5,0.6), DE [0.6,1]
8. PSO [0,0.33), GWO [0.33,0.66), WOA [0.66,0.8), DE [0.88,1]
9. PSO [0,0.2), GWO [0.2,0.55), WOA [0.55,0.8), DE [0.8,1]
10. PSO [0,0.4), GWO[0.4,0.5), WOA [0.5,0.9), DE [0.9,1]

Based on the evaluation of all configurations, the configuration that yields the best performance is PSO [0,0.33), GWO [0.33,0.66), WOA [0.66,0.8), and DE [0.88,1]. This configuration achieves an average accuracy of 92.47%, precision of 92.65%, recall of 92.47%, F1-score of 92.33%, and convergence at iteration 26.45, outperforming the other configurations.

Therefore, the selection of the optimization method in the ensemble metaheuristic optimization is as follows:

- If $\boldsymbol{r}$ <0.33, the PSO strategy is used.
- If 0.33≤ $\boldsymbol{r}$ <0.660, the GWO strategy is selected.
- If 0.66≤ $\boldsymbol{r}$ <0.80, the WOA strategy is applied.
- If $\boldsymbol{r}$ ≥0.80, the DE strategy is used."

Reviewer #2, Concern #5:

The study compares ADLC against baseline configurations such as "no optimization" and "hyperparameters + no optimization." However, it does not benchmark against widely recognized augmentation methods like SMOTE, ADASYN, or GAN-based approaches. Without such comparisons, the distinct advantages of ADLC cannot be fully appreciated.

Author response:

Thank you for this constructive suggestion. We agree that benchmarking against widely recognized augmentation methods is important to highlight the advantages of ADLC. Therefore, we have included additional comparisons with established augmentation techniques, such as SMOTE, SMOTENC, GA-SMOTE-DCNN, and SMOTEHashBoost, using the same datasets as in our study. The results demonstrate that ADLC consistently outperforms these methods, achieving higher accuracy across multiple classifiers and datasets. This strengthens the evidence of ADLC's superiority and practical relevance.

Author action:

We updated Section 3.E, final paragraph, by adding the following text:

"In addition, a comparison with previous studies employing widely recognized augmentation techniques demonstrates the superior performance of the proposed ADLC method. For the CKD dataset, prior work applying SMOTENC achieved 99.4% accuracy, while SMOTE-based augmentation reached 92%; in contrast, ADLC attained 99.5% with SVC and DT, and consistently outperformed the SMOTE baseline across all classifiers. On the Diabetes dataset, SMOTE augmentation reported 99.7% accuracy, whereas ADLC

achieved 99.88% with DT. Similarly, for the Wine dataset, SMOTEHashBoost obtained 99.44%, while ADLC reached 100% with both SVC and DT. These results confirm that ADLC delivers higher accuracy than established augmentation methods across multiple datasets, thereby highlighting its distinct advantage."

Reviewer #2, Concern #6:
Although the paper claims improvements in training efficiency and computational cost on simpler datasets, Table XII reveals significant training times for more complex cases (e.g., SVC on Balance Scale or EEG Bonn). A breakdown of computational costs for each component—confidence estimation, augmentation, and optimization—would help explain these discrepancies and guide future improvements.

Author response:
Thank you for pointing this out. We acknowledge the importance of providing a clear breakdown of computational costs to explain the observed discrepancies. We have now included detailed computational costs for the most time-consuming cases, namely SVC on the Balance Scale and EEG Bonn datasets. The results show that the majority of training time is consumed during the initialization phase, while augmentation requires negligible time. This clarification highlights that the observed computational overhead is not primarily due to the augmentation or optimization stages, but rather the initialization process.

Author action:
We updated Section 3.E, Paragraph 9, by adding the following text:

"On the Balance Scale dataset, the total training duration reached 83,989.49 seconds, consisting of an initialization time of 82,833.38 seconds and an augmentation time of 0.01 seconds. Similarly, on the EEG Bonn dataset, the total duration was 265,409.70 seconds, with an initialization time of 264,237.27 seconds and an augmentation time of 0.01 seconds."

Reviewer #2, Concern #7:
The paper mentions using Optuna for tuning but omits critical information such as the hyperparameter search space, the number of trials, and the stopping criteria used for each model. These omissions hinder reproducibility and limit readers' understanding of the optimization process.

Author response:
Thank you for raising this important point. We agree that a clear description of the Optuna tuning process is necessary for reproducibility and for providing readers with a complete understanding of our methodology. We have now included detailed information about the hyperparameter search space, the number of trials, and the stopping criteria applied to each model. This ensures that the tuning process can be replicated and clarifies how optimal hyperparameters were obtained.

Author action:
We updated Section 2.H, Paragraph 3, by adding the following text:

"In this study, the hyperparameter search space is defined specifically for each model. For the KNeighborsClassifier, the tuned parameters include the number of neighbors ranging from 1 to 30, the distance metric parameter *p* with values of either 1 or 2, and a fixed weight configuration set to 'uniform.' For the SVC model, the parameters optimized are the regularization parameter *C* ranging from 1e−3 to 1e3 and the kernel coefficient *gamma* ranging from 1e−4 to 1e1, both on a logarithmic scale, while the cache size remains fixed. For the Decision Tree model, the tuning includes the splitting criterion ('gini' or 'entropy'), splitter type ('best' or 'random'), maximum depth from 1 to 50, minimum samples required to

split a node from 2 to 20, and minimum samples required at a leaf node from 1 to 20. Across all models, the number of trials is set to 50, and the tuning process is terminated early if the model reaches 100% accuracy as the stopping criterion."

Note: References suggested by reviewers should only be added if it is relevant to the article and makes it more complete. Excessive cases of recommending non-relevant articles should be reported to ieeeaccesseic@ieee.org