sciendo
Control and Cybernetics
vol. **53** (2024) No. 4
pages: 569-618
DOI: 10.2478/candc-2024-0023

Advancing COVID-19 diagnostics with explainable AI techniques from IoT-driven gene expression data*

by

P. Santosh Kumar Patra[1*] **and Biswajit Tripathy**[2]

[1]Biju Patnaik University of Technology, Department of Computer Science and Engineering, Rourkela, Odisha, 769004, India
[2]Einstein College of Computer Application and Management, Master of Computer Applications, Khurda, Odisha, 752060, India
psantoshkumarpatra1@gmail.com
biswajit69@gmail.com
Corresponding author: P. Santhosh Kumar Patra

Abstract: The COVID-19 pandemic has resulted in over 700 million confirmed cases and more than 7 million deaths worldwide, indicating the urgent need for rapid and accurate diagnostic tools. Early and precise detection of COVID-19 variants is critical for controlling outbreaks and guiding public health interventions. However, conventional diagnostic methods, such as PCR and antigen testing, have limitations in sensitivity, processing time, and adaptability with respect to emerging variants. Additionally, the absence of gene expression-based datasets in many studies restricts the ability to analyse host immune responses, leading to suboptimal detection and prediction of disease severity. To address these challenges, this study proposes the Optimal Explainable Artificial Intelligence for Gene Expression-based COVID-19 classification (OXAI-GECC), leveraging the Internet of Things (IoT) – San Francisco COVID-19 dataset for robust and interpretable COVID-19 variant classification. The methodology integrates data preprocessing, followed by Local Interpretable Model-Agnostic Explanations (LIME)-infused Black Widow with Particle Swarm Optimization (LIME-BW-PSO) for efficient feature extraction. The Shapley Additive Explanations (SHAP) algorithm is then employed for feature selection, ensuring that the most relevant biomarkers contribute to the classification process. Finally, a Convolutional Liquid Neural Network (CLNN) is used for classification, providing enhanced accuracy and adaptability to dynamic gene expression patterns. Experimental results demonstrate that the proposed OXAI-GECC framework achieves, for the

*Submitted: October 2024; Accepted: April 2025.

data set considered, 100% accuracy, 100% precision, 100% recall, and 100% F1-score, outperforming existing methods in COVID-19 classification.

Keywords: explainable AI, COVID-19 classification, gene expression, IoT San Francisco dataset, feature engineering, convolutional liquid neural network, particle swarm optimization

1. Introduction

The COVID-19 pandemic has been continuing to affect the world up to the beginning of 2025, with more than 700 million cases and over 7 million deaths. It is still mutating and producing new strains, such as, for instance, the omicron subvariant, which has been deemed the primary cause of infection in areas like the United States of America (see Khan et al., 2023). India has recorded more than 7,000 cases of the COVID-19 variants, and some of them are dangerous, because they spread quickly and are less sensitive to current therapies. The current approaches to COVID-19 classification rely on clinical manifestations and routine laboratory tests, which are ineffective in identifying and differentiating between different virus strains as new strains are identified. These methods are not very sensitive and specific in tracking the rate, at which the virus mutates, leading to wrong diagnosis and treatment plans (see Shi et al., 2024). This incorrect diagnosis is especially of importance when gene expression data analysis is integrated into diagnostic procedures, which helps to understand host-pathogen interactions and to identify molecular markers unique to the various COVID-19 variants. This approach improves the accuracy of diagnostic instruments and contributes to the creation of individual treatment plans, which is a drawback of the classification methods.

The medical IoT is used in the health care field to connect devices, sensors, and healthcare systems that gather, share, and analyse patient data in real-time. It promotes distant supervision, identification of the disease's early signs, and individualized therapy, increasing health care delivery effectiveness. The medical IoT architecture, which is described in this paper, is presented in Fig. 1. It refers to a dynamic system for real-time COVID-19 detection, diagnosis, and patient monitoring using XAI, patient data, and wearable devices. First, the system takes COVID-19 data, particularly those, originating from San Francisco, which physicians collect, check, and endorse. The XAI models will train only using the dataset (see Kumar, Sood and Rawat, 2023) for predicting new patients' disease status. After training, the respective XAI models are stored in the hospital's server for real-time diagnosis and patient monitoring.

After a patient comes for screening, his/her genetic data is taken, the data including information on the patient's gene activity. This data is crucial for the XAI algorithms to analyse, as it contributes to developing a more precise

disease prediction. Wearable devices that can monitor different health indicators, including heart rate, temperature, and respiratory rate, can also be used in this process. These devices allow the system to be updated as the patient's condition changes, as it can receive real-time information. Next, the XAI model proceeds with the data analytics and prediction phase, using patient's genetic data, and other health data collected. The respective models employ ML approaches to predict the latent indicators and trends, associated with COVID-19 infection, which integrates the patient's genetic information with other physical characteristics, improves the AI model, and is more effective than conventional diagnosis (see Gürsoy and Kaya, 2023). XAI utilized in predicting COVID-19 cases helps healthcare providers detect possible cases more accurately and in a shorter time, which is essential for controlling the flow of patients and minimizing contamination risks. After the XAI prediction, the results are forwarded to a clinical expert for further analysis. It should be emphasized that the clinical expert has a significant role in validating the prediction, made by the XAI model. This two-step verification process ensures that the AI findings align with clinical norms and patient complaints. In case of need, the expert can modify the diagnosis, depending on other medical knowledge or tests. When the clinical expert validates the disease status, the result is returned to the patient through a mobile application (see Nasser et al., 2021), or web application (see Venkatachala et al., 2023). This application specific models enable the patient to get feedback on their health status and be better positioned to seek further treatment.

The last layer of the IoT structure is the real-time communication between the hospital server and the patient's devices, such as mobile applications or wearable devices. These bio-medical devices record the patient's details and send them over the network, allowing constant health monitoring even when not in the hospital. The system also provides patients with updates concerning their status in case of any change or development. This real-time feedback mechanism (see Aghalya et al., 2023) helps to keep the patient aware of his/her health status and make decisions on treatment at the right time. The system's combination of AI and IoT technologies offers a very effective and timely solution to detecting COVID-19 and monitoring the affected patients. Currently, some IoT devices in the market use AI to detect COVID-19. One of them is the "IDE-based SC2 biosensor" (see Zeng et al., 2022) which is based on the impedance technology to identify viral particles from samples such as saliva. This device gives results in 30-40 minutes and has been used in clinical or portable modes, thus improving the screening capacity. It leverages IoT to transmit data to healthcare systems for decision-making. Another example is provided by the AI-driven smart helmets (see Lee et al., 2022), which combine IoT sensors to measure temperatures remotely and detect potential COVID-19 cases in public spaces. These helmets, used in countries like the UAE and Italy,

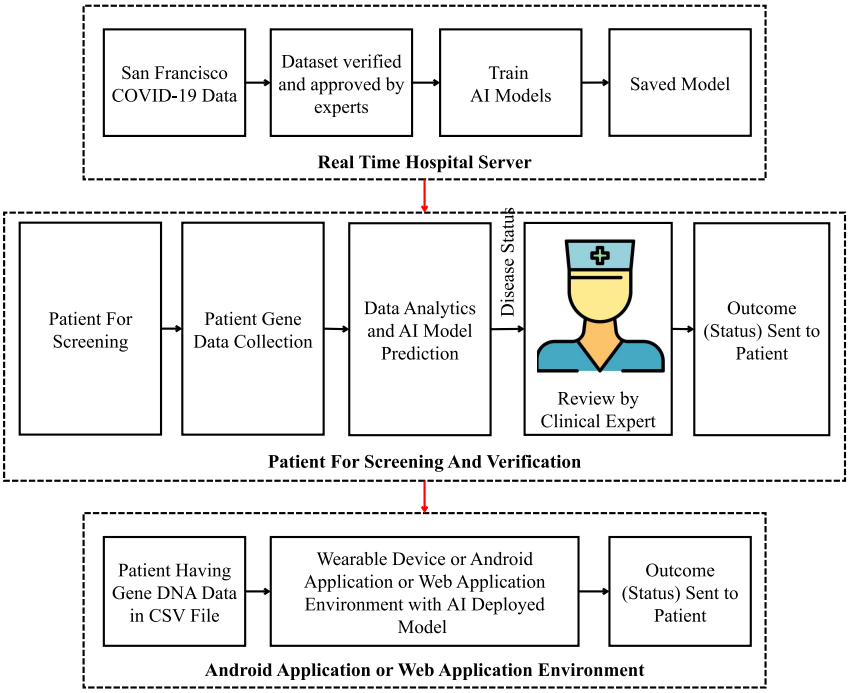


Figure 1. Medical IoT architecture for COVID-19

process large-scale temperature data quickly and integrate with AI for instant alerts on abnormal readings. They are designed for on-the-ground applications in airports, malls, and public gatherings.

Additionally, wearable IoT devices (see Khan et al., 2021), equipped with AI, analyse vital signs, such as oxygen levels and heart rates, for early COVID-19 detection. These devices, used in hospitals and at home, continuously monitor health metrics, and AI algorithms detect deviations, linked to COVID-19 symptoms, sending alerts for further testing. These innovations illustrate the potential of AI-IoT in transforming pandemic management. Thus, the present work was inspired by the real-time IoT devices and developed with consideration of gene expression analysis. The novel contributions of this work are the following ones:

- enhancing feature extraction by integrating LIME with Black Widow Optimization (BWO) and PSO, providing both interpretability and optimization,
- improving feature selection by applying SHAP values, ensuring the inclusion of the most relevant features,
- advancing classification accuracy by employing CLNNs, effectively capturing spatial and temporal patterns in gene expression data.

The rest of the paper is structured as follows: Section 2 surveys existing research in IoT-driven and GECC methods. Section 3 outlines the proposed OXAI-GECC methodology, while Section 4 presents the results. Finally, Section 5 concludes the paper with key findings and future directions of work.

2. Literature survey

This section gives a detailed analysis of related work, and Table 1 summarises the related work with both works done and their drawbacks. Thus, Arowolo et al. (2022) proposed a machine learning-based IoT system to predict COVID-19 infections. They used the Artificial Bee Colony (ABC) algorithm for feature extraction and Support Vector Machine (SVM) classifiers to analyse the San Francisco COVID-19 dataset. However, the reliance on hyperplane selection in SVM classifier makes the approach complex. Deebak and Al-Turjman (2023) introduced the Edge-Enabled Intelligent IoT (EEI-IoT) framework for early detection of COVID-19, which is developed with Extra Trees Classifier (ETC) and Multi-Layer Perceptron (MLP). This framework integrates wearable sensors to collect real-time patient data, which is then processed at the edge to monitor and predict COVID-19 symptoms. Saleh et al. (2023) developed a COVID-19 detection system using deep learning-based feature extraction coupled with a multiclass SVM and K-Nearest Neighbours (KNN). While the approach achieved high accuracy, it requires large datasets for training, which is not readily available

in all healthcare settings. Yagin et al. (2023) presented a Gradient Boosting Classifier (GBC), a Bagging classifier-based XAI model to identify COVID-19 gene biomarkers. The model utilizes machine learning techniques to analyse gene expression data, aiming to uncover biomarkers, associated with COVID-19. However, the complexity of genetic data analysis necessitates specialized knowledge, limiting its accessibility to non-experts.

Shahzad et al. (2022) proposed machine learning approaches such as AdaBoost Classifier, Decision Tree Classifier (DTC), and Random Forest Classifier (RFC) to categorize user responses related to COVID-19 vaccines. The system analyses social media data to understand public sentiment and categorize responses concerning COVID-19 vaccinations. Santosh and Tripathy (2025) introduced a hybrid feature-optimized framework for enhanced COVID-19 detection using IoT sensor data. The framework combines feature selection techniques (see Santosh and Tripathy, 2024) with machine learning models to improve the accuracy of COVID-19 detection. By optimizing feature selection, the system aims to enhance diagnostic performance. Mohtasham et al. (2024) conducted a comparative analysis of feature selection techniques for COVID-19 datasets. They evaluated various methods used to identify the most compelling features for COVID-19 prediction models. This analysis aids in improving model accuracy by selecting relevant features.

Li et al. (2024a) developed ISMI-VAE, a deep-learning model for classifying disease cells using gene expression and single nucleotide variant (SNV) data. The model employs a variational autoencoder (VAE) to integrate gene expression and SNV data for disease classification. Rahmadeyan et al. (2024) applied a DTC optimized with PSO and Genetic Algorithm (GA) for COVID-19 data classification. The optimization techniques aim to enhance the decision tree's accuracy in predicting COVID-19 cases.

Hossen, Ramanathan and Al Mamun (2024) proposed an ensemble feature selection approach, combined with such classifiers as those based on Lasso Regression, Linear Regression, Ridge Regression and Logistic Regression Classifier (LRC) for predicting COVID-19 disease. However, the proposed ensemble methods were computationally intensive, potentially limiting their scalability. Talukder et al. (2024) optimized the performance of COVID-19 detection by fine-tuning the EfficientNet deep learning architecture. Fine-tuning allowed the model to adapt to datasets, improving performance.

Niazkar et al. (2024) applied multi-gene genetic programming to predict COVID-19 prognosis using routine hematological variables. Utilizing readily available hematological data facilitates timely prognosis predictions. Qayyum et al. (2024) assessed and classified COVID-19 DNA sequences using pairwise feature concatenation from multi-transformer and deep features with Bernoulli Naïve Bayes (BNB) and Gaussian Naïve Bayes (GNB) models. However, the

reliance on complex models hinders real-time analysis, due to increased computational demands. Wang and Safo (2024) introduced Deep Integrative Discriminant Analysis (Deep IDA), a deep learning approach for multi-omics data with feature ranking, applied to COVID-19. The model integrates various omics data to improve disease classification and identifies important features contributing to predictions.

Talib et al. (2024) developed an XGBoost classifier-based XAI model for the early detection of COVID-19 using physiological data. The model utilized decision trees to analyse vital signs and other physiological parameters, aiming to provide interpretable predictions for early COVID-19 detection. Mick et al. (2020) studied upper airway gene expression to differentiate COVID-19 from other acute respiratory illnesses. They identified suppression of innate immune responses by SC2 through gene expression analysis. Lohaj et al. (2023) investigated COVID-19 dynamics using ElasticNet and XAI techniques. They focused on the influence of viral variants and prognostic classification to unravel disease progression patterns. However, the rapidly evolving nature of SC2 variants poses challenges in maintaining the model's accuracy over time. Ulvi Saygi Ayvaci et al. (2025) proposed clinically guided adaptive machine learning update strategies to predict severe COVID-19 outcomes. Their approach incorporated clinical expertise to adapt machine learning models as new data became available.

Dong et al. (2025) utilized minor variant genomes and machine learning to study the genome biology of SC2 over time. However, analysing minor variants necessitates high-throughput sequencing technologies and sophisticated bioinformatic tools, limiting applicability in resource-constrained environments. Sitjar et al. (2025) integrated machine learning with label-free Surface-Enhanced Raman Spectroscopy (SERS) methods to rapidly screen COVID-19 in patient samples. This technique aimed to provide quick and accurate diagnostics without labelling. Yadav et al. (2025) applied Autoregressive Integrated Moving Average (ARIMA) and machine learning models to interpret COVID-19 infection data across Indian provinces for Alpha, Delta, and Omicron variants. However, the accuracy of such models depends on the quality and completeness of the input data, which can vary across regions.

Matson et al. (2024) employed a deep-learning CNN approach to predict the activity of COVID-19 therapeutics and vaccines against emerging variants. However, the unpredictable nature of viral mutations poses challenges in maintaining the model's predictive accuracy over time. Kar and Ganguly (2024) applied genomic signal processing and Adaptive Machine Learning (AML) for high-performance classification of SARS-CoV-2 variants. Their approach aimed to classify viral variants based on genomic data accurately. Li et al. (2024b) developed a machine-learning model for the early detection of high-risk SARS-

Table 1. Summary of literature survey

References	**Work Done**	**Drawbacks**
Arowolo et al. (2022)	Proposed a machine learning-based IoT system using the ABC algorithm for feature extraction and SVM classifiers.	Lower classification performance.
Deebak and Al-Turjman (2023)	Introduced EEI-IoT framework to predict COVID-19 symptoms.	Time complexity issues to train models in Edge Computing.
Saleh et al. (2023)	Developed a COVID-19 detection system using deep learning feature extraction and a multi-class SVM.	Mult-class SVM hyperplanes are not synchronized effectively.
Yagin et al. (2023)	Presented an XAI model to identify COVID-19 gene biomarkers by analyzing gene expression data.	Only considered gene data and demographic information are skipped.
Li et al. (2024)	Developed ISMI-VAE to analyze SVN gene expression.	Integration of multi-omics data was computationally intensive and complex.
Rahmadeyan et al. (2024)	Applied decision tree classifiers optimized with PSO-GA.	The weights of PSO and GA are not correctly balanced.
Hossen et al. (2024)	Proposed an ensemble feature selection approach combined with machine learning.	Feature ranking and selection issues.

Table 1, continued

Talukder et al. (2024)	Optimized the performance of COVID-19 detection by fine-tuning the EfficientNet deep learning architecture.	Fine-tuning requires large and diverse datasets for effective model adaptation.
Qayyum et al. (2024)	Assessed and classified COVID-19 DNA sequences using multi-transformer-based deep features with machine learning models.	Reliance on complex models and increased computational demands.
Wang and Safo (2024)	Introduced Deep IDA, a deep learning approach for multi-omics data with feature ranking.	Feature ranking failed due to improper feature encoding.
Lohaj et al. (2023)	Investigated COVID-19 dynamics using machine learning and XAI techniques.	Increased losses over time due to not focusing on classification.
Dong et al. (2025)	Utilized minor variant genomes and machine learning to study genome biology.	Analyzing minor variants necessitates high-throughput sequencing technologies.
Yadav et al. (2025)	Applied ARIMA and machine learning models to interpret COVID-19 infection data.	The accuracy of the model was reduced due to multi-class classification issues.
Li et al. (2024b)	Developed a machine learning model for the early detection of high-risk SC2 variants.	The unpredictable nature of viral variants poses challenges in maintaining model predictive accuracy over time.

CoV-2 variants. However, the model's reliance on timely and accurate genomic data limits its effectiveness in regions with limited sequencing capabilities.

3. Proposed methodology

3.1. Overall outline

Considering the limitations identified in the existing surveys and proposals on COVID-19 classification using gene expression data, the here forwarded OXAI-GECC proposes a novel algorithm that integrates LIME-BW-PSO for feature extraction, SHAP for feature selection, and a CLNN for classification, as this is shown in Fig. 2. The proposed approach, which is based on combining LIME-BW-PSO, SHAP, and CLNN methodologies, has not been used before. It helps to overcome the existing shortcomings by increasing interpretability, relevance of features, and classification accuracy. The detailed outline of the approach is as follows:

Step 1: Data preprocessing:

The OXAI-GECC starts by gathering data from the IoT San Francisco COVID-19 dataset. The data is cleaned to remove technical variation, and missing values are estimated to get clean data for analysis.

Step 2: LIME-BW-PSO Feature Extraction:

To extract the model's features, OXAI-GECC uses the LIME-BW-PSO. The LIME gives local interpretability by substituting complex models with simpler ones, whereas the BWO divides the data into workable portions. The inclusion of PSO enhances the feature selection process by securing effective search in the feature space. It produces a set of features that effectively identify the patterns in the gene expression data.

Step 3: SHAP Feature Selection:

After extracting the features, SHAP values are computed to identify the importance of each feature in the model. Features with higher SHAP values are deemed more important and included in the final model. This helps to avoid using features that are not useful in the analysis and makes the model more interpretable.

Step 4: CLNN Classification:

The selected features are passed through CLNNs. The CNN is used for spatial feature extraction. In contrast, the Liquid State Machine (LSM) is used for dynamic temporal data processing. They are both suitable for handling the complexities associated with gene expression data. This architecture enables the model to capture complex patterns and temporal relations, enhancing the classification accuracy.

3.2. LIME-BW-PSO feature extraction

3.2.1. An outline of the procedure

The LIME-BW-PSO feature extraction method combines three efficient techniques, namely LIME, BWO, and PSO, to optimize feature selection and extraction from a preprocessed data set, as illustrated in Fig. 3. The first step is LIME feature analysis, which aims at determining the most important features in each dataset by explaining the reasoning of a predictive model. This interpretability step helps to eliminate all the features that are not important in the decision-making process, hence reducing dimensionality and retaining important information. After the first feature set is extracted, the BWO feature ranking is performed. The bio-inspired metaheuristic optimization algorithm, BWO, improves the feature subset by selecting the best features through selection, reproduction, and mutation. This step helps in avoiding the inclusion of features that do not significantly improve the model's performance and only slow it down.

According to the BWO ranking, the optimized features are aggregated and passed to the PSO feature encoding. PSO, another optimization technique based on swarm intelligence, optimizes the feature set by searching for multiple feature sets and choosing the best classification accuracy. This additional refinement step helps in making the extracted feature set possibly close to optimal and as little as possible sensitive to overfitting and noise. Further, the output features are produced and used in machine learning models for COVID-19 detection or any other predictive model. This LIME-BW-PSO framework is unique and highly effective in feature extraction in complex datasets, because of its ability to overcome existing feature selection limitations using interpretability, bio-inspired ranking, and swarm-based optimization.

3.2.2. LIME feature analysis

The LIME Feature Analysis method provides an interpretable way to understand how individual features contribute to a model's decision-making process.

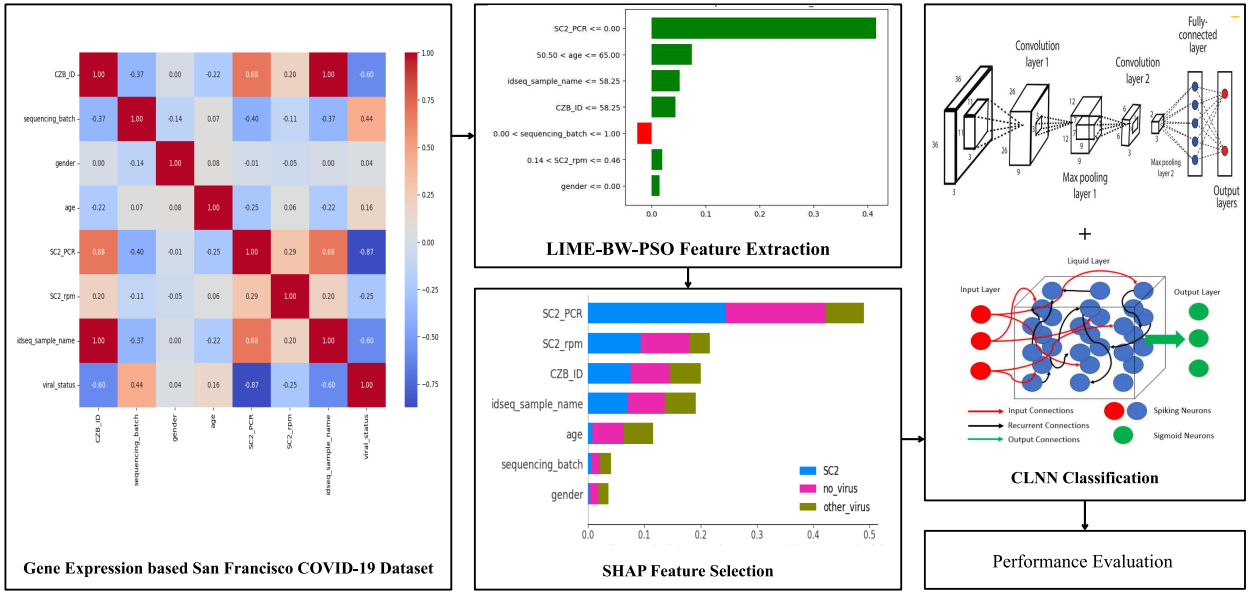


Figure 2. Proposed OXAI-GECC block diagram

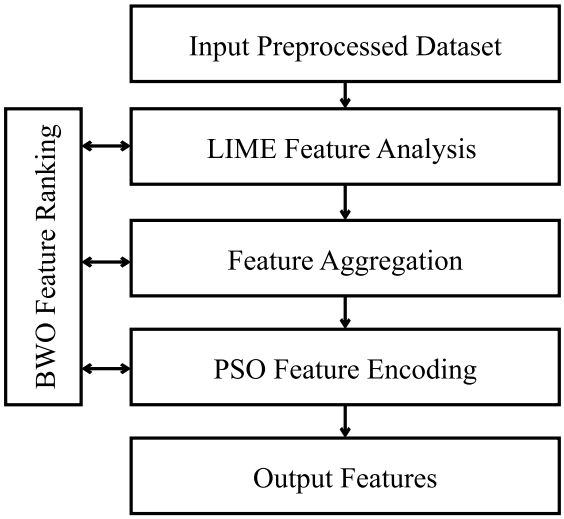


Figure 3. Proposed LIME-BW-PSO block diagram

It approximates a black-box model's behavior by training an interpretable local surrogate model around a specific instance. Figure 4 shows the LIME feature analysis block diagram.

The detailed operation of the proposed methodology is given as follows:

Generating perturbed samples: To understand the impact of each feature, LIME generates multiple perturbed instances (X') around the original data instance (X), slightly modifying one or more feature values. These perturbed samples are denoted as:

$$X^{'} = X + \epsilon,\ where\ \epsilon \sim N(0, \sigma^2) \quad . \tag{1}$$

Here, the value of σ^2 controls the perturbation level, ensuring slight variations without deviating significantly from the original data distribution.

Obtaining model predictions: For each perturbed sample, the black-box model $f\left(X^{'}\right)$ predicts an outcome. The corresponding prediction values form a set of output responses:

$$Y^{'} = f\left(X^{'}\right) \tag{2}$$

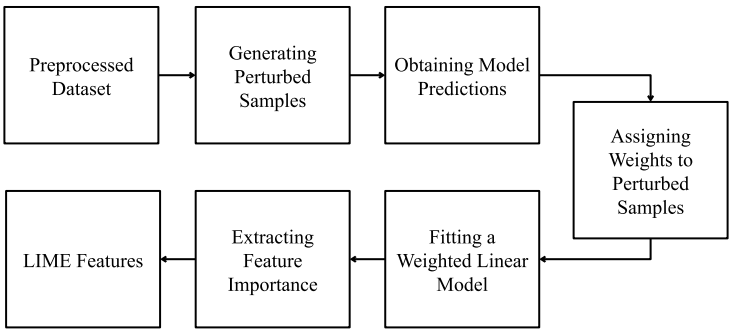


Figure 4. Proposed LIME method use block diagram

Here, $Y^{'}$ represents each perturbed instance's predicted probabilities or class labels.

Assigning weights to perturbed samples: Since LIME focuses on explaining local behaviour, it assigns weights to each perturbed sample based on its similarity to the original instance. The similarity function $\pi\left(X, X^{'}\right)$ is defined as:

$$\pi\left(X, X^{'}\right) = \exp\left(\frac{-d\left(X, X^{'}\right)^{2}}{2\sigma^{2}}\right) \quad , \tag{3}$$

where $d\left(X, X^{'}\right)^{2}$ is the Euclidean distance between the original and perturbed instances. This distance ensures that points closer to the original instance have a higher influence.

Fitting a weighted linear model: LIME then fits an interpretable linear regression model $g(X')$ that approximates the behaviour of $f(X')$ in the local neighborhood. The objective function for this linear model is:

$$\sum_{i=1}^{N} \pi\left(X, X_{i}^{'}\right)\left(g\left(X_{i}^{'}\right)\right)^{2} \quad . \tag{4}$$

Here, N is the number of perturbed samples, and $g\left(X_{i}^{'}\right)$ represents a simple, interpretable function (e.g., linear regression).

Extracting feature importance: The final step involves interpreting the coefficients β_j from the linear model $g\left(X^{'}\right)$, where:

$$\hat{Y} = \beta_0 + \sum_{j=1}^{N} \beta_j . X_j \quad , \tag{5}$$

where β_j represents the importance of feature X_j in influencing the model's decision. Higher absolute values of β_j indicate a more substantial feature influence. Applying LIME feature analysis enables identifying key predictive features in complex datasets, ensuring model interpretability, while improving feature selection for downstream classification tasks.

3.2.3. BWO feature ranking

The BWO feature ranking method is a bio-inspired metaheuristic technique miming black widow spiders' survival and cannibalistic behaviour. It is applied to optimize feature ranking by selecting the most informative features, while discarding redundant or irrelevant ones. Figure 5 shows the BWO feature ranking block diagram. The process begins with the importance of the LIME extracted features as input.

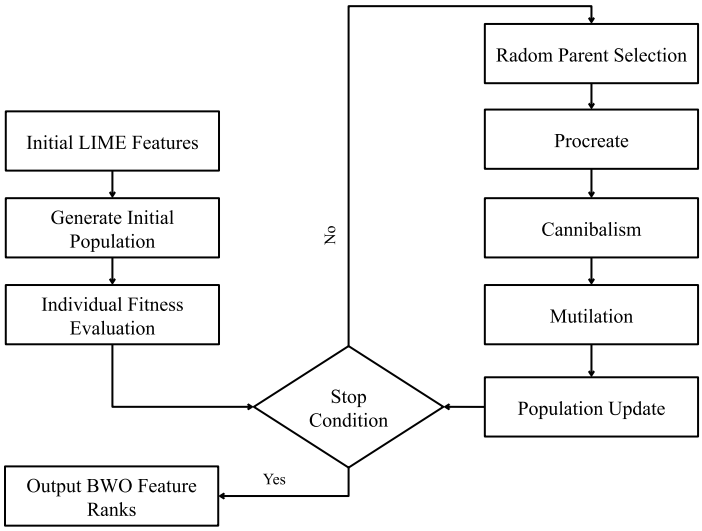


Figure 5. The BWO feature ranking block diagram

The initial population P consists of candidate feature subsets, derived from LIME's feature importance scores. Each subset S_i is represented as a binary vector, where selected features are assigned the value of one and unselected features are 0.

$$S_i = \{S_{i,1}\,, S_{i,,2}, ..., S_{i,M}\} \quad i = 1, 2, ..., P \quad . \tag{6}$$

Here, M represents the total number of features. Each feature subset S_i is evaluated based on a fitness function that assesses classification accuracy while penalizing large feature subsets to encourage compact representations. The fitness function is given by:

$$F\left(S_i\right) = \alpha \cdot Acc\left(S_i\right) - \beta \cdot \frac{\mid S_i \mid}{M} \tag{7}$$

In formula (7), $Acc\left(S_i\right)$ is the classification accuracy obtained using the selected features, $\mid S_i \mid$ is the number of selected features, and α, β are weighting coefficients balancing accuracy and feature count. If the convergence criterion (maximum number of iterations or stability in feature ranking) is met, the process terminates, and the optimal feature subset is selected. Otherwise, the population undergoes evolutionary operations.

The BWO applies a crossover mechanism, inspired by black widow reproduction, where offspring inherit a combination of features from parent subsets. The parent subsets are selected using a tournament or roulette-wheel selection method based on their fitness scores. Two parent feature subsets, $S_{parent1}$ and $S_{parent2}$, are chosen for reproduction. Offspring subsets are generated by combining features from selected parents. The crossover operation follows a probabilistic rule:

$$S_{child,j} = \lambda S_{parent1,j} + (1 - \lambda) S_{parent2,j} \quad , \tag{8}$$

where λ is the crossover rate (randomly chosen between 0 and 1), determining the contribution of each parent. The low-fitness feature subsets are eliminated to mimic the cannibalistic nature of black widow spiders, and only the fittest offspring survive. The cannibalism operation removes the worst-performing γ fraction of the population:

$$P' = P - \gamma P \quad , \tag{9}$$

where P' is the reduced population after weak solutions are discarded. A mutation process is applied to introduce diversity and avoid local optima by flipping feature selection states randomly:

$$S_{i,j} = P^{'}\left(S_{i,j} \oplus \xi\right) \tag{10}$$

Here, ξ is a random binary mutation operator. The new population replaces the previous one, ensuring only the best-performing feature subsets are retained. The algorithm iterates until convergence is achieved, refining the selected feature subset S^* that maximizes the fitness function:

$$S^* = \left(F\left(S_{i,j}\right)\right). \tag{11}$$

The ranking of features is determined based on their selection frequency across multiple runs, ensuring that the most influential features are prioritized. Each feature is assigned an importance score I_j, based on its occurrence in the best-ranked subsets across iterations:

$$I_J = \frac{\sum_{i=1}^{N} S_{i,j}}{P} \quad . \tag{12}$$

Here, I_j represents the significance of feature X_j in classification. Through these iterative operations, BWO Feature Ranking effectively selects an optimal subset of features, reducing dimensionality while maintaining high classification performance, thus improving the interpretability and efficiency of machine learning models.

3.2.4. PSO feature encoding

The PSO feature encoding process begins with aggregate LIME features, and the BWO feature ranks as inputs. Figure 6 shows the proposed PSO feature encoding block diagram. The goal is to refine the feature selection process by optimizing feature weights through iterative position updates in a multidimensional search space. The output is an optimized feature subset for classification tasks.

In what follows the key steps of the procedure are explained, according to the scheme of Fig. 6.

Initializing the swarm of particles: Each particle in the swarm represents a candidate feature subset with an associated position $X_i\left(t\right)$ and velocity $V_i(t)$ at iteration t. The initial position and velocity are randomly initialized:

$$X_i\left(0\right) = X_{min} + \left(X_{max} - X_{min}\right) \cdot r \tag{13}$$

$$V_i\left(0\right) = V_{min} + \left(V_{max} - V_{min}\right) \cdot r \tag{14}$$

Here, r is a random number in [0,1], and $X_{min}X_{max}$, V_{min}, and V_{max} define the search space boundaries.

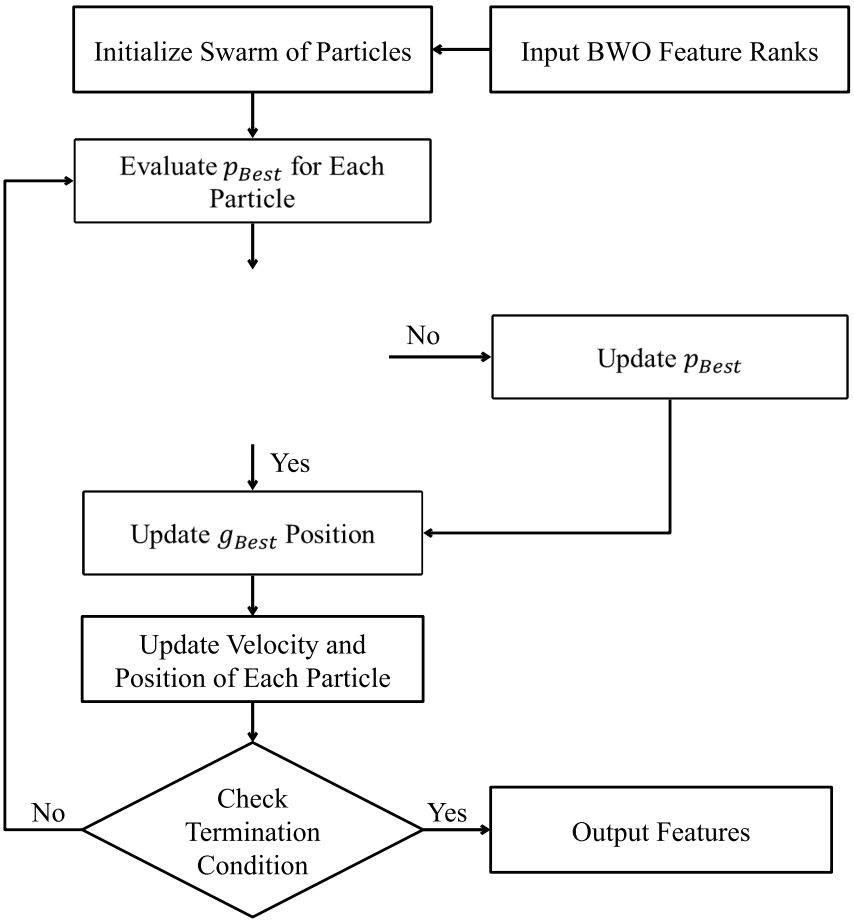


Figure 6. PSO feature encoding block diagram

Evaluating the personal best ($\mathbf{p_{Best}}$) for each particle: Each particle maintains its best-found position (p_{Best}) based on the value of fitness function $F(X_i)$, which evaluates classification accuracy and feature subset compactness:

$$p_{Best_i} = (F(X_i)) \tag{15}$$

Here, $F(X_i)$ is calculated using:

$$F(X_i) = \alpha \cdot Acc(X_i) - \beta \cdot \frac{|S_i|}{M} \quad , \tag{16}$$

where $Acc(X_i)$ is the classification accuracy, and $|X_i|$ is the number of selected features.

Updating global best ($\mathbf{g_{Best}}$) position: The best-performing feature subset among all particles is assigned as the global best (g_{Best}):

$$g_{Best_i} = (F(p_{Best_i})). \tag{17}$$

It ensures that the swarm moves toward the optimal solution.

Updating velocity of each particle: Velocity updates are performed using the PSO velocity equation:

$$V_{i(t+1)} = \omega V_i(t) + c_1 r_1 (pBest_i - X_i(t)) + c_2 r_2 (g_{Best} - X_i(t)). \tag{18}$$

Here, ω is the inertia weight for balancing exploration and exploitation, c_1 *and* c_2 are acceleration coefficients, while r_1 and r_2 are random values in [0,1].

Updating particle position: Each particle's position is updated based on its new velocity:

$$X_i(t+1) = X_i(t) + V_i(t+1). \tag{19}$$

This formula ensures that features are dynamically re-ranked based on their significance.

Checking the termination condition: The algorithm checks if the stopping criterion is met (e.g., reaching a predefined accuracy threshold or maximum number of iterations). The algorithm terminates if the convergence condition is met: $F(g_{Best}) \geq F_{threshold}$. Otherwise, the process repeats.

Outputting the optimized feature subset: After convergence, the best feature subset, corresponding to g_{Best}, is selected as the final optimized feature set for classification, such as $X_{final} = g_{Best}$. It ensures that the most relevant features from aggregate LIME features and BWO feature ranks are retained for enhanced model performance.

3.3. SHAP feature selection

SHAP is a powerful feature selection technique, based on cooperative game theory, ensuring that only the most relevant features contribute to model's performance. Figure 7 shows the proposed SHAP feature selection block diagram. It assigns importance scores to features based on their contribution to prediction accuracy. The process starts with the LIME-BW-PSO feature set and refines it to extract the final optimal features by systematically evaluating each feature's impact. This method improves model interpretability, reduces computational complexity, and enhances predictive performance. The systematic ranking and evaluation of features make SHAP an essential tool for feature selection in machine learning applications.

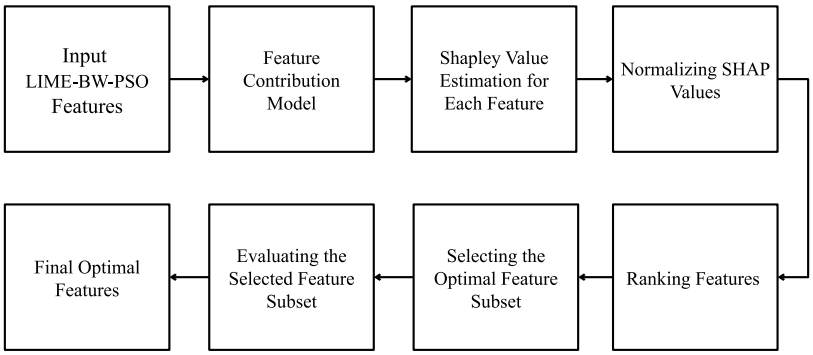


Figure 7. Proposed SHAP feature selection block diagram

Feature contribution model: The primary goal of SHAP is to decompose the prediction $f(X)$ into the sum of individual feature contributions. The prediction function for a given input set X was expressed as:

$$f\left(X\right) = \phi_0 + \sum_{i=1}^{M} \phi_i \tag{20}$$

Here, ϕ_0 is the base prediction and ϕ_i represents the SHAP value assigned to each feature X_i. The total contribution of features should sum up to approximate the actual model prediction $f\left(X\right)$.

Computing the Shapley value for each feature: The Shapley value of each feature is computed based on its marginal contribution when added to different subsets of features. It is formulated as:

$$\phi_i = \sum_{S \subseteq F \setminus \{X_i\}}^{M} \frac{|S|!\,(M - |S| - 1)!}{M!}.\,[f\,(S \cup \{X_i\}) - f\,(S)] \tag{21}$$

Here, F is the complete set of features, S represents different subsets of features excluding X_i, $f\,(S \cup \{X_i\})$ is the model prediction when X_i is included, and $f(S)$ is the prediction without X_i. This equation ensures a fair distribution of feature importance by averaging contributions across all possible subsets.

Normalizing SHAP values: Since they can have different scales, they must be normalized to make them comparable across features. The normalized importance score for each feature is calculated as follows:

$$\phi_i^{'} = \frac{\phi_i - \min\,(\phi)}{\max\,(\phi) - \min\,(\phi)} \tag{22}$$

Here, $\phi_i^{'}$ represents the scaled importance score. This normalization maps SHAP values into a range of [0,1], making the establishment of a threshold for feature selection easier.

Ranking features based on SHAP values: Once the SHAP values are computed and normalized, the features are ranked based on their importance scores in descending order:

$$\text{Rank}\,(X_i) = \text{argsort}\left(-\phi_i^{'}\right). \tag{23}$$

This ranking allows us to prioritize the most significant features, ensuring that only the ones with high predictive power are retained.

Selecting the optimal feature subset: To determine the final feature subset, a threshold τ is applied to remove less important features. The subset of selected features is given by:

$$X_{opt} = \left\{X_i \,|\, \phi_i^{'} \geq \tau\right\}. \tag{24}$$

Here, τ is a predefined threshold. Features with normalized SHAP values below this threshold are discarded.

Evaluating the selected feature subset: After selecting the optimal feature set, the new, reduced dataset is validated using a performance metric

that balances accuracy and feature reduction. The function used to evaluate the performance of the selected features is:

$$F\left(X_{opt}\right) = \alpha \cdot Acc\left(X_{opt}\right) - \beta \cdot \frac{\mid X_{opt} \mid}{M} \quad . \tag{25}$$

Here, $Acc\left(X_{opt}\right)$ is the model accuracy with the selected features, $\frac{|X_{opt}|}{M}$ represents the fraction of retained features, and α and β are weighting factors to balance accuracy and feature count. It ensures that the final feature set is both optimal in accuracy and minimal in size.

Final optimal features: If the selected subset meets the accuracy and feature count requirements, it is designated as the final optimal feature set: $X_{final} = X_{opt}$. This final feature set is used for downstream tasks, such as classification or regression, ensuring an interpretable and efficient model.

3.4. CLNN classification

The CLNN is an advanced deep-learning model that combines CNN's feature extraction capability with LSMs' dynamic adaptability. Figure 8 shows the proposed CNN classifier block diagram. The convolutional layers extract spatial dependencies, while the LNN dynamically captures complex, time-dependent relationships, making the system robust for classification tasks. This hybrid approach ensures accurate and adaptable decision-making, leveraging spatial and temporal feature representations.

Convolutional layers: The SHAP-selected features serve as the input to the convolutional layers, which extract meaningful spatial patterns. The convolutional operation is defined as:

$$h_{i,j}^{(l)} = \sigma\left(\sum_{m=0}^{M}\sum_{n=0}^{N} W_{mn}^{(l)} X_{(i+m)(j+n)}^{(l-1)} + b^{(l)}\right) \quad . \tag{26}$$

Here, $h_{i,j}^{(l)}$ is the activation at position (i, j) in the l-th layer, $W_{mn}^{(l)}$ represents the convolutional filter weights, $X_{(i+m)(j+n)}^{(l-1)}$ is the input feature map from the previous layer, $b^{(l)}$ is the bias term, $\sigma(\cdot)$ is a non-linear activation function, such as ReLU. This operation helps to extract spatial dependencies within the SHAP features, producing a feature map highlighting essential classification patterns.

Pooling layers: After convolution, a pooling layer is applied to reduce computational complexity while retaining key information. The max-pooling operation is given by:

$$P_{ij}^{(l)} = \left(h_{(i+m)(j+n)}^{(l)}\right) \quad . \tag{27}$$

Here, S represents the pooling window. It reduces the feature map size while preserving the most dominant activations, improving computational efficiency.

Transition to LSM dynamics: Unlike conventional feedforward networks, LSMs employ time-dependent neurons that evolve on the basis of differential equations. The hidden state dynamics of an LNN neuron are modelled as follows:

$$\frac{dh_i}{d_t} = -\lambda h_i + \sum_{n=0}^{N} W_{ij}\sigma\left(h_j\right) + I_i\left(t\right) \quad . \tag{28}$$

Here, h_i represents the hidden state of neuron i, λ is the decay rate controlling memory retention, W_{ij} is the synaptic weight from neuron j to i, $I_i\left(t\right)$ is an external input signal, $\sigma(\cdot)$ is a non-linear activation function. It ensures that LSM neurons dynamically adapt based on time-dependent input patterns, allowing the network to capture complex temporal relationships in data.

Dynamic neural responses computation: The LSM employs differential equations to compute neuron responses continuously. The neuronal response at time t was computed as:

$$h_i\left(t\right) = h_i\left(0\right)\, e^{-\lambda t} + \int_0^t e^{-\lambda\left(t-s\right)} \left(\sum_j W_{ij}\sigma\left(h_j\left(s\right)\right) + I_i\left(s\right) \right) ds\,. \tag{29}$$

It governs the evolution of each neuron's state, capturing long-range dependencies in the feature space.

Fully connected layer: After extracting time-dependent patterns, the processed feature representations are passed through a fully connected layer for classification. The output of the fully connected layer is computed as:

$$Z_i = \sum_j W_{ij}^{(fc)} h_j^{(fc)} + b_i^{(fc)} \quad . \tag{30}$$

Here, z_i is the pre-activation score for class i, $W_{ij}^{(fc)}$ are the weights in the fully connected layer, $h_j^{(fc)}$represents the final hidden layer activations from the LSM, and $b_i^{(fc)}$ is the bias term. This operation combines extracted temporal and spatial features, preparing them for the final classification step.

Softmax activation: To obtain the final class probabilities, the SoftMax function is applied:

$$P\left(y_i\right) = \frac{e^{z_i}}{\sum\limits_j e^{z_j}} \quad . \tag{31}$$

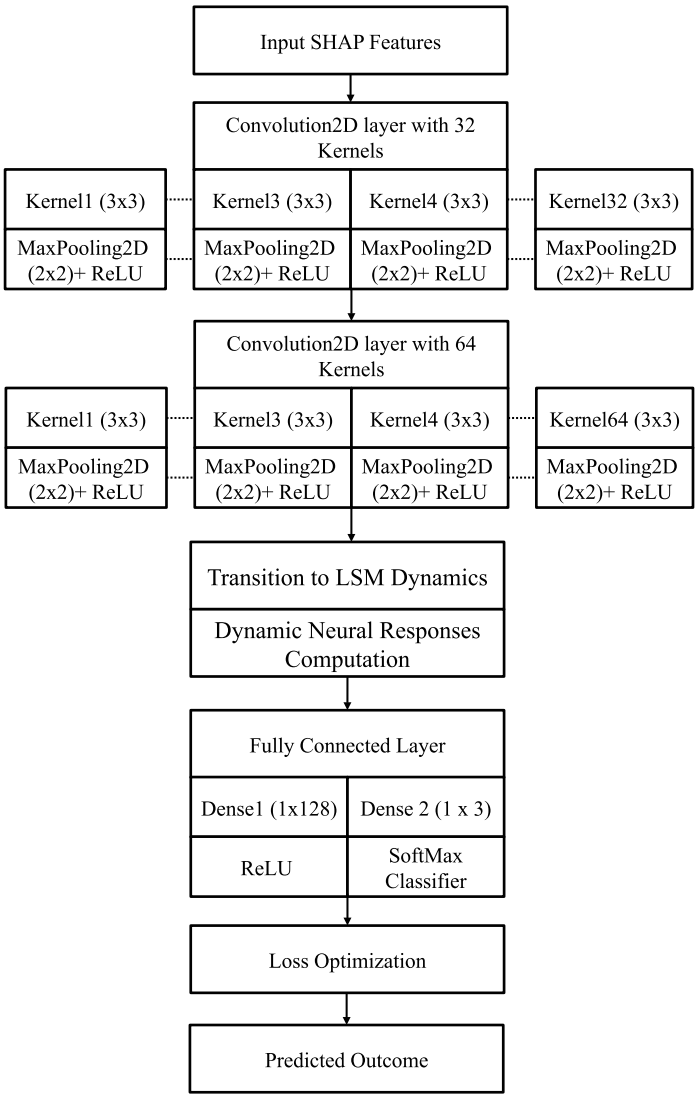


Figure 8. Proposed CLNN classifier block diagram

Here, $P(y_i)$ is the probability of the input belonging to class i, and z_i is the score computed in the fully connected layer. The SoftMax function ensures that the output probabilities sum to 1, allowing the model to make a confident classification decision.

Loss optimization: To optimize the CLNN, the categorical cross-entropy loss function is used:

$$L = \sum_{i=1}^{C} y_i \log P\left(P\left(y_i\right)\right) \quad . \tag{32}$$

Here, y_i is the true class label (one-hot encoded), and C is the total number of classes. The network is trained using gradient descent with an adaptive optimizer such as Adam, which updates parameters as follows:

$$\theta^{t+1} = \theta^t - \eta \nabla L(\theta) \quad . \tag{33}$$

Here, η is the learning rate, and $\nabla L(\theta)$ represents the gradient of the loss function concerning model parameters θ.

4. Results and discussion

This section is devoted to evaluation of the performance of different methods by applying them to the same dataset. The comparison is based on various metrics to ensure a fair and comprehensive analysis.

4.1. The dataset

The San Francisco COVID-19 dataset, curated by the Chan Zuckerberg Biohub, comprises comprehensive gene expression profiles, derived from nasopharyngeal and oropharyngeal swabs of patients with varying respiratory conditions. Research conducted by Mick et al. (2020) provided all the dataset details. The kept dataset is open-access, and it is accessible via https://github.com/czbiohub-sf/covid19-transcriptomics-pathogenesis-diagnostics-results?tab=readme-ov-file.

This dataset facilitates in-depth analyses of host gene expression responses to SARS-CoV-2 compared to other respiratory infections, enabling studies on differential expression, immune response characterization, and the development of diagnostic classifier. While experienced professionals meticulously curated the San Francisco COVID-19 dataset to minimize biases in data collection, it is essential to acknowledge and address potential ethical considerations inherent in AI-driven medical diagnoses. Even with high-quality datasets, AI algorithms

can inadvertently perpetuate biases in the data, leading to disparities in healthcare outcomes. To mitigate these risks, this study incorporates comprehensive evaluations of algorithmic fairness and implements strategies to ensure equitable and ethical application of AI in medical diagnostics.

Each sample in the dataset is annotated with specific metadata fields:

- **CZB_ID**: A unique identifier assigned to each sample.
- **Sequencing_batch**: Indicates the library preparation and sequencing batch, reflecting the processing group of the sample.
- **Gender**: Determined based on the expression levels of Y chromosome genes.
- **Age**: Reported by the patient; if missing, imputed as the mean age of patients within the same viral status group.
- **SC2_PCR**: Denotes the result of the clinical PCR test for SARS-CoV-2, determining COVID-19 status.
- **SC2_rpm**: Represents reads-per-million mapping to SARS-CoV-2 in the metagenomic sequencing data, indicating viral load.
- **IDSeq_sample_name**: Corresponds to the matching IDSeq sample name.
- **viral_status**: Categorizes samples as 'SC2', 'other_virus' (presence of other pathogenic respiratory viruses), or 'no_virus' (absence of detectable viruses).

4.2. Simulation environment

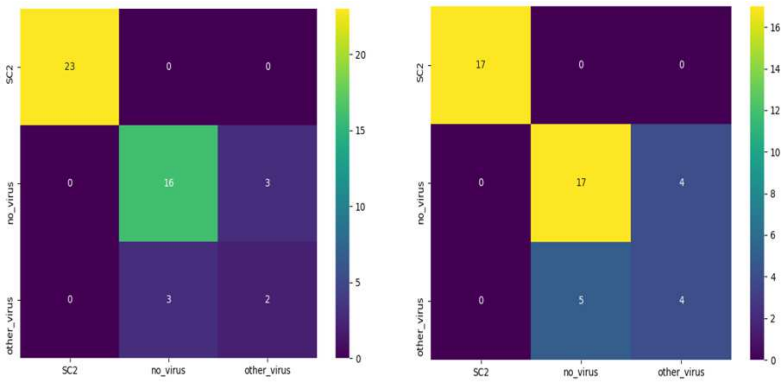
The resource requirements of the IoT-based implementations should be optimized for efficient computation while being scalable. The model is trained using the 5-fold cross-validation to improve its stability with the learning rate of 0.001, the batch size of 32, and 100 epochs to achieve the best result with less resource utilization. The hardware components consist of ALIENWARE Intel Core i7 10th Gen 10750H CPU, 16GB RAM, and 1TB SSD, making it possible to process and store data quickly. The 8GB NVIDIA GeForce RTX 2070 GPU enhances deep learning processing, which helps process IoT data in real time. The software environment, including Python 3.7 and TensorFlow 2.x, offers excellent flexibility and scalability for the tasks in this project. These resources are ideal for model training and deployment, because they facilitate fast processing of IoT-based applications that require real-time data analysis and low latency.

4.3. Performance evaluation

Figure 9 shows the confusion matrices of two classifiers, namely EEI-IoT-ETC (see Deebak and Al-Turjman, 2023) and DTC (see Shahzad et al., 2022), for

the classification task of the SC2, no_virus, and other_virus. According to the EEI-IoT-ETC confusion matrix, the model achieves 23 true positives for SC2, 16 true negatives for no_virus, and two true negatives for other_virus, while it misclassifies 3 no_virus as other_virus and 3 other_virus as no_virus. On the other hand, the DTC confusion matrix has a relatively lower accuracy with 17 correct SC2 cases, 17 no_virus cases, and 4 other_virus cases, but 4 no_virus cases were wrongly classified as other_virus and 5 other_virus cases as no_virus. These values indicate that the existing EEI-IoT-ETC and DTC had relatively low performance.

Table 2 presents the performance comparison of the proposed OXAI-GECC method against tree-based classifiers, including EEI-IoT-ETC and DTC, based on key evaluation metrics, such as accuracy, recall, precision, and F1-score. The EEI-IoT-ETC classifier achieves accuracy of 87.234%, recall of 74.524%, precision of 81.944%, and F1-score of 75.455%, showing relatively strong performance. The DTC classifier performs slightly worse, with accuracy of 78.723%, recall of 72.222%, precision of 70.784%, and F1-score of 70.395%, indicating lower classification effectiveness. Thus, the proposed OXAI-GECC model outperforms both methods, achieving the perfect accuracy of 100%, demonstrating its superior capability in classification tasks.



(a) (b)

Figure 9. Confusion matrices. (a) EEI-IoT-ETC (see Deebak and Al-Turjman, 2023) and (b) DTC (see Shahzad et al., 2022)

Figure 10 presents confusion matrices for various classification algorithms. And so, Fig. 10 (a) shows confusion matrix of AdaBoost (see Shahzad et al., 2022), which contains 23 true positives for SC2, 17 for no_virus with 2 mis-

classified as other_virus, and 1 for other_virus with 4 misclassified as no_virus. Then, Fig. 10 (b) shows confusion matrix of XGBoost (see Talib et al., 2024), which contains 23 true positives for SC2, 18 for no_virus with 3 misclassified as other_virus, and 3 for other_virus with 3 misclassified as no_virus.

Table 2. Performance comparison of the proposed OXAI-GECC with tree-based classifiers

Classifier	**Accuracy**	**Recall**	**Precision**	**F1-Score**
EEI-IoT-ETC (see Deebak and Al-Turjman, 2023)	87.234	74.524	81.944	75.455
DTC (see Shahzad et al., 2022)	78.723	72.222	70.784	70.395
Proposed OXAI-GECC	100.000	100.000	100.000	100.000

Next, Fig. 10 (c) shows confusion matrix of Bagging (see Yagin et al., 2023), which contains 23 true positives for SC2, 17 for no_virus with 2 misclassified as other_virus, and 0 for other_virus with 5 misclassified as no_virus. Figure 10 (d) shows confusion matrix of RFC (see Shahzad et al., 2022), which contains 23 true positives for SC2, 15 for no_virus with 4 misclassified as other_virus, and 2 for other_virus with 3 misclassified as no_virus.

Figure 10 (e) presents the confusion matrix of GBC (see Yagin et al., 2023), which contains 23 true positives for SC2, 13 for no_virus with 6 misclassified as other_virus, and 1 for other_virus with 4 misclassified as no_virus.

Figure 10 (f) shows the confusion matrix of proposed OXAI-GECC model, which contains 19 true positives for no_virus with 1 misclassified as other_virus, 8 for other_virus, and 19 SC2 samples classified correctly. It is easy to see that altogether only a single case is misclassified.

Table 3 presents a comparative performance analysis of the proposed OXAI-GECC model against various ensemble classifiers, including AdaBoost, XGBoost (see Talib et al., 2024), Bagging, RFC, and GBC.

Among the traditional ensemble classifiers, GBC achieves the highest accuracy of 87.234%, with recall of 77.619%, precision of 80.606%, and the F1-score of 78.571%. RFC and Bagging follow closely, both achieving the accuracy of 85.106%, but RFC has slightly better recall (72.857%) and precision (76.087%) than Bagging (69.762% recall, 75.333% precision). XGBoost and AdaBoost (see Shahzad et al., 2022) have lower performance, both with accuracy of 80.851%, though XGBoost outperforms AdaBoost in recall (75.556% vs. 66.429%) and precision (72.500% vs. 66.304%). The here proposed OXAI-GECC model signif-

icantly outperforms all methods, achieving 100% across all performance metrics, demonstrating its superior classification capability.

Table 3. Performance comparison of OXAI-GECC with ensemble classifiers

Classifier	**Accuracy, %**	**Recall, %**	**Precision, %**	**F1-Score, %**
AdaBoost	80.851	66.429	66.304	65.751
XGBoost	80.851	75.556	72.500	71.892
Bagging	85.106	69.762	75.333	68.889
RFC	85.106	72.857	76.087	73.362
GBC	87.234	77.619	80.606	78.571
Proposed OXAI-GECC	100.000	100.000	100.000	100.000

Table 4 presents a performance comparison of the proposed OXAI-GECC model with regression-based classifiers using four key evaluation metrics. Among the traditional methods, ElasticNet (see Lohaj et al., 2023) shows the lowest performance with accuracy of 40.553%, recall of 13.184%, precision of 31.333%, and F1-Score of 11.900%. Lasso Regression (see Hossen, Ramanathan and Al Mamun, 2024) performs slightly better, achieving accuracy of 42.553%, recall of 14.184%, precision of 33.333%, and F1-Score of 19.900%. Linear Regression (see Hossen, Ramanathan and Al Mamun, 2024) significantly outperforms the previous models, reaching accuracy of 81.83%, recall of 67.38%, precision of 78.48%, and F1-Score of 70.83%. Ridge Regression (see also Hossen, Ramanathan and Al Mamun, 2024) further improves accuracy to 82.979%, though it has lower recall, at 57.143%, precision of 66.667%, and F1-Score of 61.111%. The LRC (see Hossen, Ramanathan and Al Mamun, 2024) performs better than all previous methods, achieving accuracy of 90.193%, recall of 88.483%, precision of 85.337%, and F1-Score of 89.383%. However, the proposed OXAI-GECC model outperforms all the compared classifiers, achieving perfect accuracy, recall, precision, and F1-Score of 100.000%, demonstrating its superior classification capability.

Next, Fig. 11 presents confusion matrices for various classification algorithms, where Fig. 11(a) presents ElasticNet, Fig. 11(b): Lasso Regression, Fig. 11(c): Linear Regression, and Fig. 11(d): Ridge Regression, all showing identical results with 19 SC2 samples misclassified as no_virus, 17 true positives for no_virus with no misclassifications, and 11 true positives for other_virus with no misclassifications, indicating a complete failure to classify SC2. Then, Fig. 11 (e) presents LRC, mirroring the previous pattern with 19 SC2 samples misclassified as no_virus, 17 true positives for no_virus, and 11 true positives for other_virus, showing consistent misclassification of SC2 by these regression-

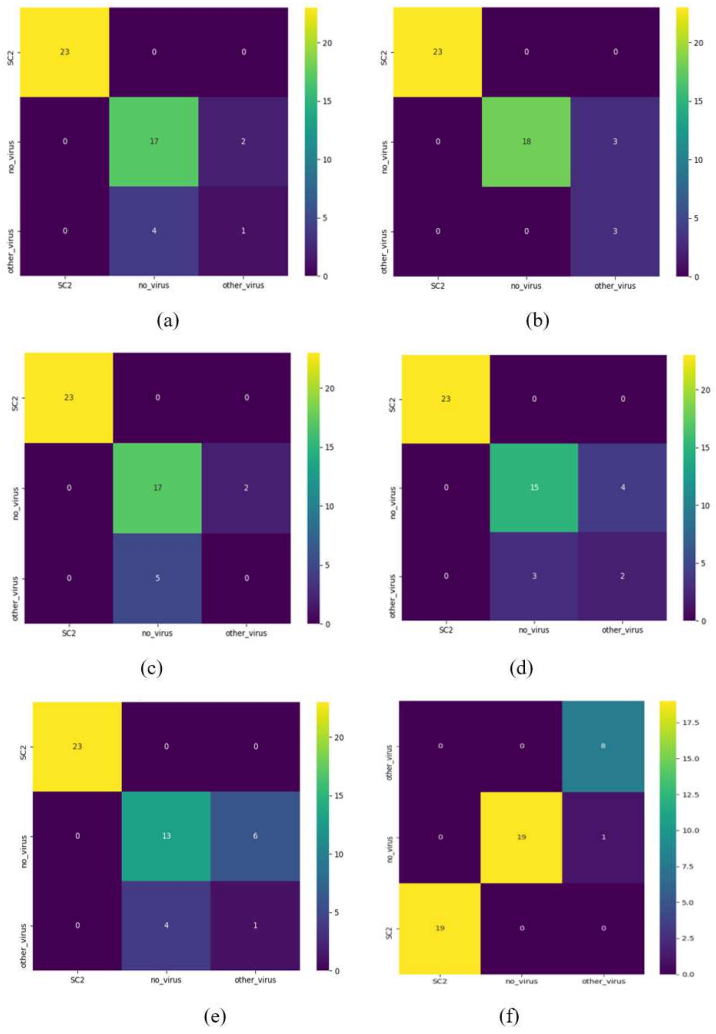


Figure 10. Confusion matrices: (a) AdaBoost (see Shahzad et al., 2022), (b) XGBoost (see Talib et al., 2024), (c) Bagging (see Yagin et al., 2023), (d) RFC (see Shahzad et al., 2022), (e) GBC (see Yagin et al., 2023), (f) the proposed OXAI-GECC

Table 4. Performance comparison of OXAI-GECC with regression classifiers

Classifier	**Accuracy, %**	**Recall, %**	**Precision, %**	**F1-Score, %**
ElasticNet	40.553	13.184	31.333	11.900
Lasso Regression	42.553	14.184	33.333	19.900
Linear Regression	81.83	67.38	78.48	70.83
Ridge Regression	82.979	57.143	66.667	61.111
LRC	90.193	88.483	85.337	89.383
Proposed OXAI-GECC	100.000	100.000	100.000	100.000

based methods. Figure 11 (f) for the proposed OXAI-GECC shows 19 true positives for no_virus with 1 misclassified as other_virus, 8 true positives for other_virus, and 19 SC2 samples misclassified as no_virus, highlighting a similar misclassification of SC2 as observed in the regression models, but with a different pattern in the no_virus and other_virus classifications.

Figure 12 presents the confusion matrices for two classifiers, (a) GNB (see Qayyum et al., 2024) and (b) BNB (see Qayyum et al., 2024), comparing their classification performance across three categories: SC2, no_virus, and other_virus. In Fig. 12 (a), GNB correctly classifies 22 SC2 cases, 17 no_virus cases, and misclassifies 8 other_virus cases as no_virus. In Fig. 12 (b), BNB also correctly classifies 22 SC2 cases, but shows lower performance in distinguishing no_virus and other_virus cases, correctly identifying 12 no_virus cases, while misclassifying 5 as other_virus, and correctly identifying 6 other_virus cases, while misclassifying 2 as no_virus. The confusion matrices indicate that BNB has a higher misclassification rate in differentiating no_virus and other_virus than GNB.

Table 5 compares the performance of the proposed OXAI-GECC model with Bayesian classifiers. The BNB classifier achieves accuracy of 82.979%, precision of 74.167%, recall of 79.524%, and F1-Score of 75.774%, demonstrating moderate performance. The GNB classifier performs slightly worse, with accuracy of 74.468%, precision of 72.222%, recall of 71.111%, and F1-Score of 70.909%. The proposed OXAI-GECC model surpasses both classifiers, achieving perfect accuracy, precision, recall, and F1-Score of 100.000%, highlighting the superiority of its classification capability over Bayesian approaches.

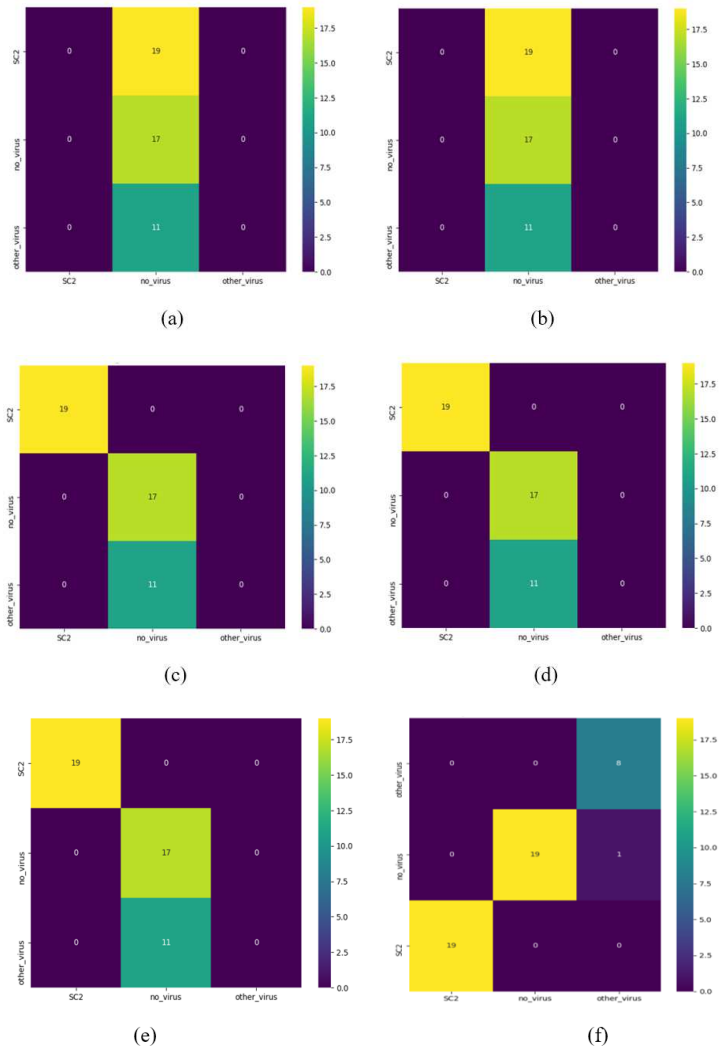


Figure 11. Confusion matrices: (a) ElasticNet, (b) Lasso Regression, (c) Linear Regression, (d) Ridge Regression, (e) LRC, (f) proposed OXAI-GECC

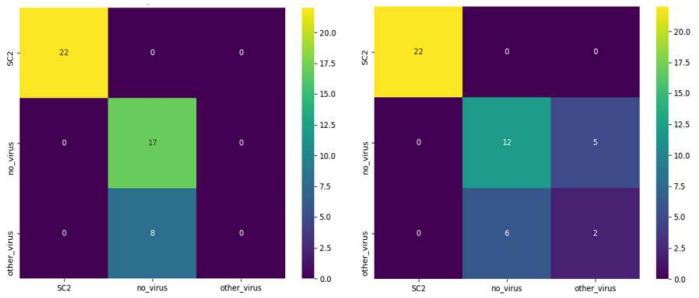


Figure 12. Confusion matrices: (a) GNB, (b) BNB

Table 6 compares the performance of the proposed OXAI-GECC model with various linear classifiers, including KNN (see Saleh et al., 2023), EEI-IoT-MLP (see Deebak and Al-Turjman, 2023), and SVM (see Saleh et al., 2023). The KNN classifier achieves the accuracy of 82.979%, with recall and precision of 66.667% each, resulting in F1-score of 66.316%. The EEI-IoT-MLP method demonstrates better performance, achieving 91.489% accuracy and high recall of 94.667%, but lower precision of 73.333%, leading to F1-score of 74.879%. The SVM classifier performs moderately, with accuracy of 85.106%, recall of 57.971%, precision of 64.815%, and F1-score of 60.976%. The proposed OXAI-GECC model outperforms all of these with perfect accuracy of 100.000%, demonstrating its superior classification capabilities compared to existing linear classifiers.

Table 5. Performance comparison of the proposed OXAI-GECC with Bayes classifiers

Classifier	**Accuracy, %**	**Precision, %**	**Recall, %**	**F1-Score, %**
BNB	82.979	74.167	79.524	75.774
GNB	74.468	72.222	71.111	70.909
Proposed OXAI-GECC	100.000	100.000	100.000	100.000

Figure 13 presents confusion matrices for different classification models, namely: (a) SVM, (b) KNN, (c) EEI-IoT-MLP, and (d) the proposed OXAI-GECC. In Fig. 13 (a), SVM correctly classifies 19 SC2 cases, 16 no_virus cases, and misclassifies one as other_virus, while 8 other_virus cases are misclassified

Table 6. Performance comparison of the proposed OXAI-GECC with linear classifiers

Classifier	**Accuracy, %**	**Recall, %**	**Precision, %**	**F1-Score, %**
KNN	82.979	66.667	66.667	66.316
EEI-IoT-MLP	91.489	94.667	73.333	74.879
SVM	85.106	57.971	64.815	60.976
Proposed OXAI-GECC	100.000	100.000	100.000	100.000

as no_virus, and 3 are correctly identified. In Fig. 13 (b), KNN shows 23 correct SC2 classifications, 18 no_virus cases are correctly classified, 3 misclassified as other_virus, and all other_virus cases are correctly classified. In Fig. 13 (c), EEI-IoT-MLP correctly identifies 19 SC2 cases and 17 no_virus cases, and misclassifies 11 other_virus cases as no_virus. In Fig. 13 (d), the proposed OXAI-GECC classifies all 19 SC2 cases correctly, 19 no_virus cases with only one misclassification, and 8 other_virus cases accurately, demonstrating superior performance over prior models.

Table 7 presents a performance comparison between the proposed OXAI-GECC model and optimized classifiers, including Q-SVM-ABC (see Arowolo et al., 2022) and L-SVM-ABC (see also Arowolo et al., 2022). The Q-SVM-ABC achieves accuracy of 95.74%, recall of 97.10%, precision of 94.84%, and F1-score of 95.89%, indicating strong classification performance. Then, the L-SVM-ABC exhibits lower effectiveness, with accuracy of 78.723%, recall of 89.245%, precision of 72.22%, and F1-score of 69.78%, showing its lower reliability in classification tasks. The proposed OXAI-GECC model significantly outperforms both, achieving perfect score across all metrics, with 100.000% accuracy, recall, precision, and F1-score, and demonstrating its superior classification capability and robustness over the other models.

Figure 14 presents the confusion matrices for the Q-SVM-ABC and L-SVM-ABC classifiers, illustrating their classification performance across the three considered cases, i.e. SC2, no_virus, and other_virus. In Fig. 14 (a), the Q-SVM-ABC correctly classifies 19 SC2 cases and 22 no_virus cases and misclassifies 2 other_virus cases as no_virus and 3 other_virus cases as SC2. In Fig. 14 (b), the L-SVM-ABC correctly classifies 18 SC2 cases and 23 no_virus cases, but misclassifies 1 SC2 case as no_virus, 1 no_virus case as SC2, and 4 other_virus cases as SC2. The comparison shows that Q-SVM-ABC achieves better classi-

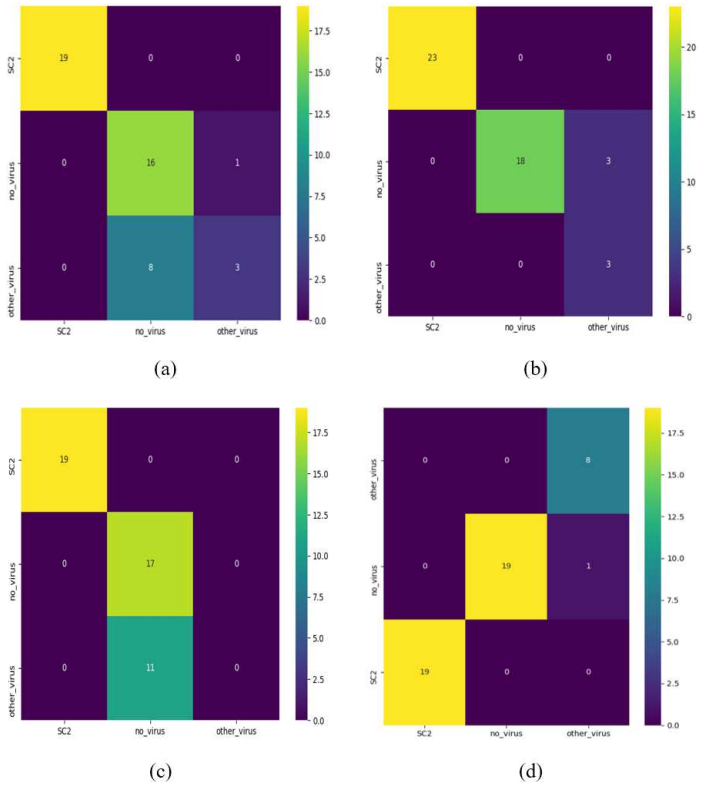


Figure 13. Confusion matrices: (a) SVM, (b) KNN, (c) EEI-IoT-MLP, (d) proposed OXAI-GECC

Table 7. Performance comparison of proposed OXAI-GECC with optimized classifiers

Classifier	**Accuracy, %**	**Recall, %**	**Precision, %**	**F1-Score, %**
Q-SVM-ABC	95.74	97.10	94.84	95.89
L-SVM-ABC	78.723	89.245	72.22	69.78
Proposed OXAI-GECC	100.000	100.000	100.000	100.000

fication accuracy by reducing misclassification errors in the SC2 and no_virus categories compared to L-SVM-ABC.

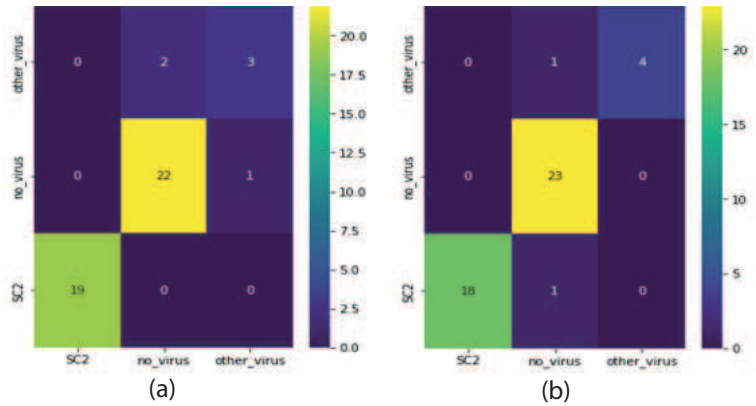


Figure 14. Confusion matrices: (a) Q-SVM-ABC, (b) L-SVM-ABC

4.4. XAI analysis

Figure 15 shows the LIME summary plot for the "Other Virus" class and illustrates the prediction probabilities, decision boundaries, and contributing features. The model assigns 100% probability to the "Other Virus" class, with 0% probability for both "SC2" and "No Virus." The key influencing features include SC2_PCR (0.00), sequencing_batch (1.00), idseq_sample_name (34.00), CZB_ID (34.00), gender (0.00), age (72.00), and SC2_rpm (0.59). The bar chart in the middle highlights the threshold-based decision process, where SC2_PCR ≤ 0.00 (0.56 impact), sequencing_batch >0.00 (0.10 impact), idseq_sample_name <threshold (0.07 impact), and CZB_ID ≤ 57.00 (0.05 impact) contribute most significantly. The color-coded feature impact visualization further supports the sample classification as "Other Virus."

Figure 16 shows the LIME summary plot for the "SC2" class, showing that the model predicts SC2 with 100% probability while assigning 0% probability to both "Other Virus" and "No Virus." The key influencing features include SC2_PCR (0.00), sequencing_batch (0.00), idseq_sample_name (157.00), CZB_ID (157.00), SC2_rpm (0.38), age (54.00), and gender (1.00). The middle bar chart highlights decision thresholds, where SC2_PCR ≤ 0.00 (0.58 impact), sequencing_batch $\leq$ threshold (0.10 impact), idseq_sample_name >117.00 (0.08 impact), and CZB_ID >117.00 (0.06 impact) significantly contribute to the classification. The visualization emphasizes the feature influence in confirming the sample as "SC2."

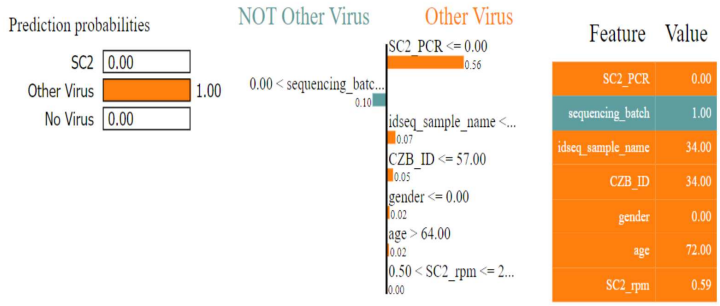


Figure 15. LIME summary plot for the Other Virus class

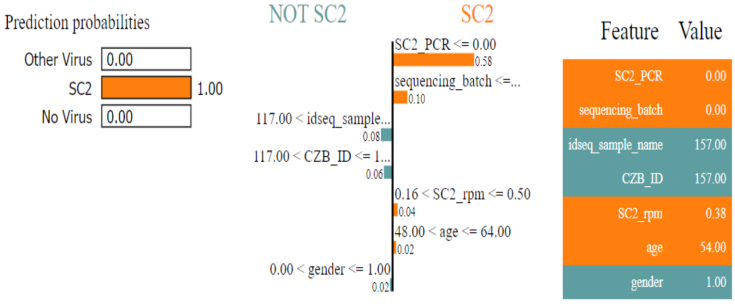


Figure 16. LIME summary plot for SC2 class

Figure 17 shows the LIME summary plot for the "No Virus" class, indicating that the model assigns 100% probability to "No Virus," with 0% probability for both "SC2" and "Other Virus." The key contributing features include SC2_PCR (0.00), sequencing_batch (0.00), CZB_ID (199.00), idseq_sample_name (199.00), gender (0.00), SC2_rpm (0.16), and age (46.00). The middle bar chart highlights decision thresholds, where SC2_PCR $\leq$ 0.00 (0.58 impact), sequencing_batch $\leq$ threshold (0.10 impact), CZB_ID >175.00 (0.02 impact), and idseq_sample_name >threshold (0.01 impact) are the ones that play significant roles in classification. The visualization confirms the model's decision that this sample belongs to the "No Virus" category.

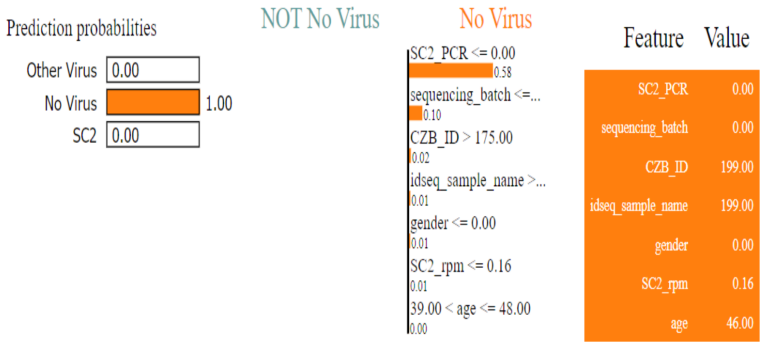


Figure 17. LIME summary plot for No Virus class

The subsequent Fig. 18 presents the LIME explainability graphs for the three classification outcomes: (a) Other Virus, (b) SC2, and (c) No Virus.

In all cases, the most influential feature in the classification decision is SC2_PCR <= 0.00, with a contribution of approximately 0.55–0.60 in each of the classes. For the Other Virus class, as shown in Fig. 18 (a), positive contributions come from the features: SC2_PCR (0.55), CZB_ID ($\leq$ 57.00), and idseq_sample_name ($\leq$ 57.00), while sequencing_batch ($\leq$ 1.00) negatively influences classification. In the SC2 class, as this is shown in Fig. 18 (b), features such as SC2_PCR ($\leq$ 0.00, 0.55), sequencing_batch ($\leq$ 0.00, 0.20), and SC2_rpm (0.16 <SC2_rpm $\leq$ 0.50, 0.10) contribute positively, while idseq_sample_name (117.00 <idseq_sample_name $\leq$ 175.00, -0.08) and CZB_ID (117.00 <CZB_ID $\leq$ 175.00, -0.06) do exert negative influences. Lastly, in the No-Virus class, as this is shown in Fig. 18 (c), positive contributions are observed with regard to SC2_PCR ($\leq$ 0.00, 0.58), sequencing_batch ($\leq$ 0.00, 0.15), and CZB_ID (>175.00, 0.02), with all features supporting the classification. These results demonstrate the dominant role of SC2_PCR in classification decisions, followed by sequencing_batch, idseq_sample_name, and CZB_ID, depending on the class.

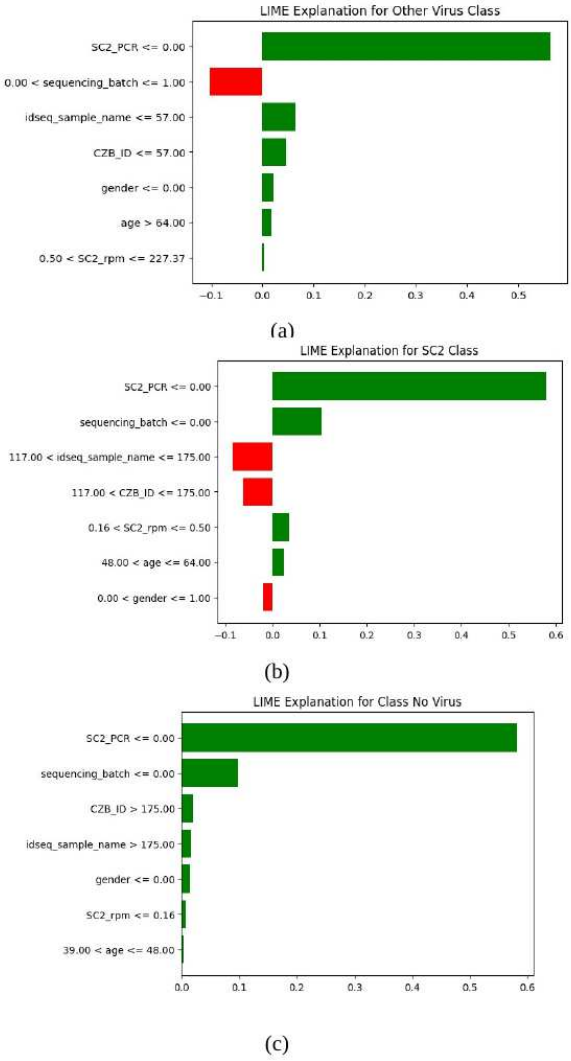


Figure 18. LIME explainability graphs for individual classes: (a) Other Virus, (b) SC2, (c) No Virus

The subsequent Fig. 19 provides the SHAP explainability graphs for different classification outcomes from the considered method. And so, Figs. 19(a), 19(b), and 19(c) show the SHAP value distributions for SC2, No Virus, and Other Virus classes, respectively. SC2_PCR has consistently the highest impact across all classes, with SHAP values between -0.4 and 0.4. The features such as sequencing batch, age, and SC2_rpm also contribute significantly, though their impacts vary across classes. In Fig. 19(a), SC2_PCR strongly influences positive SC2 classifications with values near 0.4, while in Fig. 19(b), its influence shifts for No Virus predictions. Figure 19(d) presents the mean absolute SHAP values, highlighting SC2_PCR as the most influential feature across all classes, followed by idseq_sample_name and CZB_ID. The model differentiates between classes based on the varying SHAP contributions of these features, with no_virus and SC2 showing dominant contributions from SC2_PCR, while other_virus exhibits a more evenly distributed influence among sequencing_batch, age, and SC2_rpm.

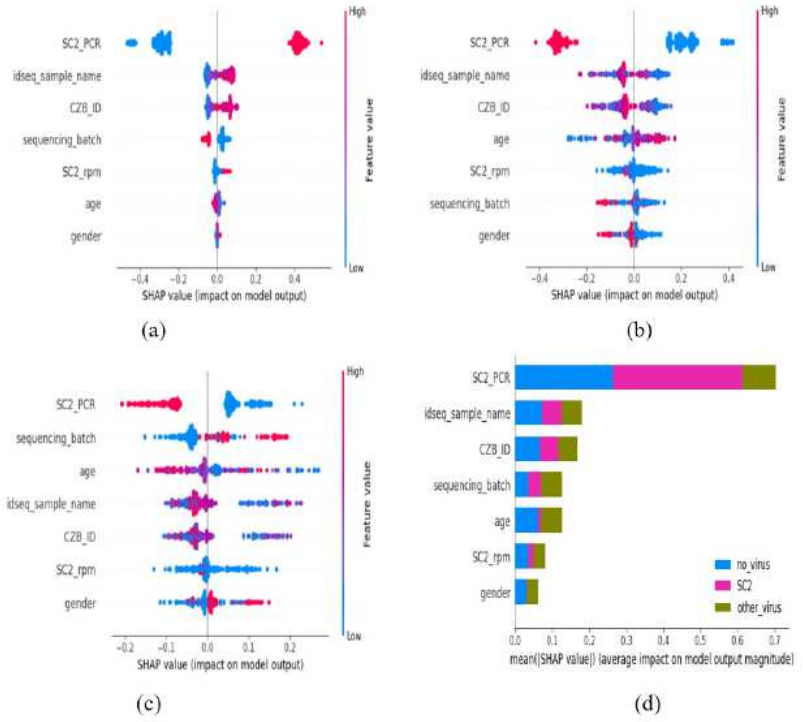


Figure 19. SHAP explainability graphs: (a) SC2 class, (b) No Virus class, (c) Other Virus class, (d) overall

Table 8 presents the computational efficiency analysis of various methods by comparing their training time, accuracy, and loss values. The Proposed OXAI-GECC method outperforms all others, achieving 100% accuracy with the lowest training time of 400s and the zero-loss value (0.0), indicating optimal efficiency. The SVM-ABC (see Arowolo et al., 2022) method follows with 95.74% accuracy, requiring 720s of training time and the loss of 0.110. The EEI-IoT-ETC (see Deebak and Al-Turjman, 2023) and BNB (see Qayyum et al., 2024) methods achieve 87.234% and 82.979% accuracy, respectively, with training times of 520s and 480s, and respective loss values of 0.195 and 0.220. XGBoost (see Talib et al., 2024) records 80.851% accuracy, requiring 610s for training, with the higher loss, at 0.245. Then, ElasticNet (see Lohaj et al., 2023) and Lasso Regression (see Hossen, Ramanathan and Al Mamun, 2024) perform the worst, with low accuracy values of 40.553% and 42.553%, respectively, training times of 430s and 450s, and the highest loss values of 0.460 and 0.440, respectively. The results highlight the superior computational efficiency and accuracy of the here proposed OXAI-GECC, showing that it is the most effective method among the compared approaches.

Table 8. Computational efficiency analysis of considered methods

Method	**Training Time (s)**	**Accuracy (%)**	**Loss Value**
EEI-IoT-ETC	520	87.234	0.195
XGBoost	610	80.851	0.245
ElasticNet	430	40.553	0.460
Lasso Regression	450	42.553	0.440
BNB	480	82.979	0.220
SVM-ABC	720	95.740	0.110
Proposed OXAI-GECC	400	100.00	0.0

4.5. Case study

The emergence of new SC2 variants, such as omicron and its subvariants, underscores the virus's rapid evolution and potential impact on gene expression and immune responses. The California Health and Human Services (CHHS) dataset contains archived data on the prevalence of circulating SARS-CoV-2 variants in California. Compiled by the California Department of Public Health (CDPH), this dataset was generated through genomic sequencing efforts to monitor variant distribution across the state. The dataset was accessed with the following

link: https://data.chhs.ca.gov/dataset/covid-19-variant-data/resource/d7f9acfa-b113-4cbc-9abc-91e707efc08a.

Table 9 presents the performance comparison of SC2 variants classification methods. And so, the ISMI-VAE (see Li et al., 2024a) method achieves 90.183% accuracy, 91.382% recall, 90.374% precision, and the F1-score of 92.846. The ARIMA (see Yadav et al., 2025) model improves upon this with 95.436% accuracy, 96.483% recall, 95.363% precision, and the F1-score of 95.283. The DLCNN (see Matson et al., 2024) classifier further enhances performance, attaining 98.187% of accuracy, 98.990% of recall, 98.019% of precision, and the F1-score of 98.836. The here proposed OXAI-GECC method, however, outperforms all with 99.763% accuracy, 99.903% recall, 99.193% precision, and the F1-score of 99.378, demonstrating its superior capability in classifying the SC2 variants.

Table 9. Performance comparison of SC2 variants classification methods

Classifier	**Accuracy, %**	**Recall, %**	**Precision, %**	**F1-Score, %**
ISMI-VAE	90.183	91.382	90.374	92.846
ARIMA	95.436	96.483	95.363	95.283
DLCNN	98.187	98.990	98.019	98.836
Proposed OXAI-GECC	99.763	99.903	99.193	99.378

The subsequent Fig. 20 presents the confusion matrices of SC2 variant classification methods, illustrating the classification performance of the four approaches accounted for: ISMI-VAE, ARIMA, DLCNN, and the here proposed OXAI-GECC method. The confusion matrices provide the numbers of correctly and incorrectly classified instances for each variant across these methods. And so, Fig. 20 (a), representing ISMI-VAE, shows significant misclassifications, especially for the Alpha, Epsilon, and Gamma variants, with the Alpha variant being largely confused with Beta (351 misclassified instances) and the Epsilon variant misclassified into Gamma (40 instances). The Mu and Omicron variants exhibit high misclassification rates, with Mu being confused with Beta (361 instances) and omicron being misclassified into Beta (61 instances), Lambda (74 instances), as well as Mu (86 instances).

Then, Fig. 20(b), corresponding to ARIMA technique, demonstrates improved classification accuracy, particularly for the Alpha variant, which is correctly classified in 379 cases. However, Beta and Delta variants still show notable misclassification, with Beta having 38 instances misclassified into Alpha,

and Delta having 48 instances misclassified into Beta. The Mu and Omicron variants still exhibit some confusion, with Omicron being misclassified into Mu (294 instances) and the "Other" variants misclassified into omicron (143 cases).

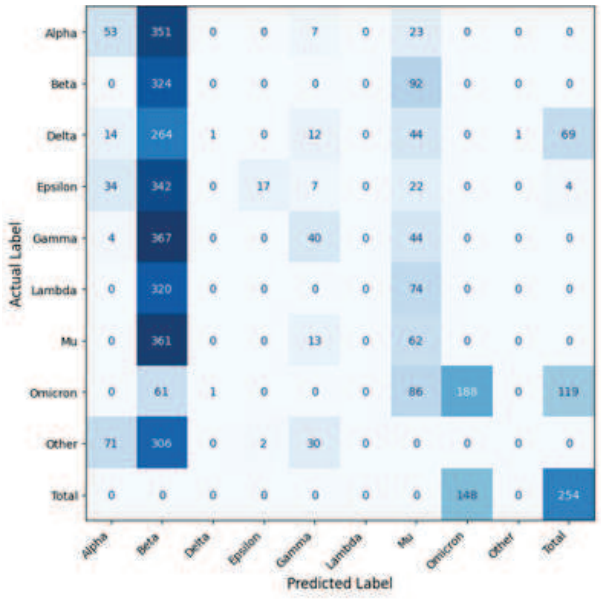
Figure 21(c), presenting the results for the DLCNN approach, shows further refinement in classification accuracy, particularly for Alpha (434 correct classifications) and Omicron, where 152 cases are correctly classified. However, Delta, Gamma, and Epsilon still show misclassification issues, with Delta cases being misclassified into Beta (96 cases) and Epsilon being frequently misclassified into Gamma (57 cases). Lambda and Mu exhibit moderate misclassification levels, with Lambda misclassified into Gamma in 80 cases and Mu into Gamma in 83 cases.

The here proposed OXAI-GECC method, whose results are shown in Fig. 21 (d), exhibits the best classification performance among all the approaches considered, as this is evident from the highly diagonal structure of the confusion matrix. Each variant is classified with high precision, as Alpha, Beta, Delta, Epsilon, Gamma, Lambda, Mu, Omicron, and other variants are almost entirely mapped to their correct labels with minimal misclassification.

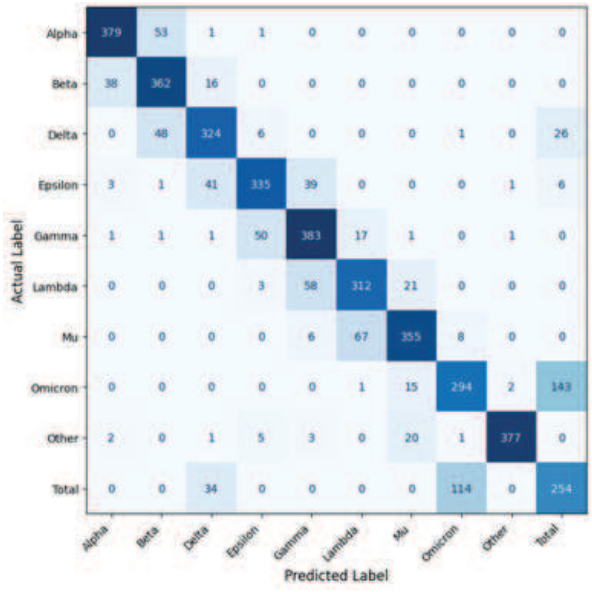
4.6. Discussion

Deployment of OXAI-GECC in healthcare presents several challenges. Data privacy and security are crucial, as healthcare data is sensitive and subject to regulations such as the Health Insurance Portability and Accountability Act (HIPAA) and the General Data Protection Regulation (GDPR). The implementation of AI systems in the current healthcare structures is generally challenging due to data silos and compatibility problems. Clinical assessment is an important process to prove that the product is safe and effective, and this involves a series of tests and verification of compliance with regulatory requirements. So, acceptance by healthcare professionals and patients requires friendly designs and decision-making processes.

This project, related to the present report, seeks to advance the IoT patient monitoring systems by focusing on data privacy and security issues, considering HIPAA and GDPR. It will entail data encryption, proper authentication procedures, and general protective measures for personal health information. Furthermore, the project here showcased plans to perform risk evaluation and define data processing contracts with third parties to meet the legal requirements. This work incorporates such improvements to maintain all patients' confidentiality and trust in healthcare solutions.

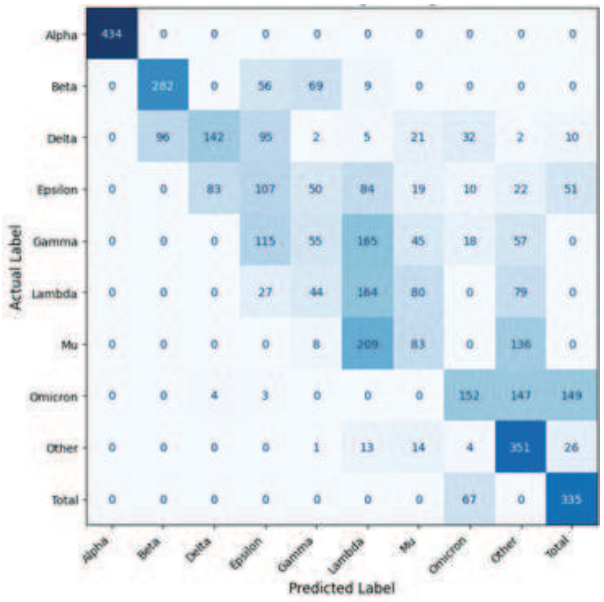


(a)

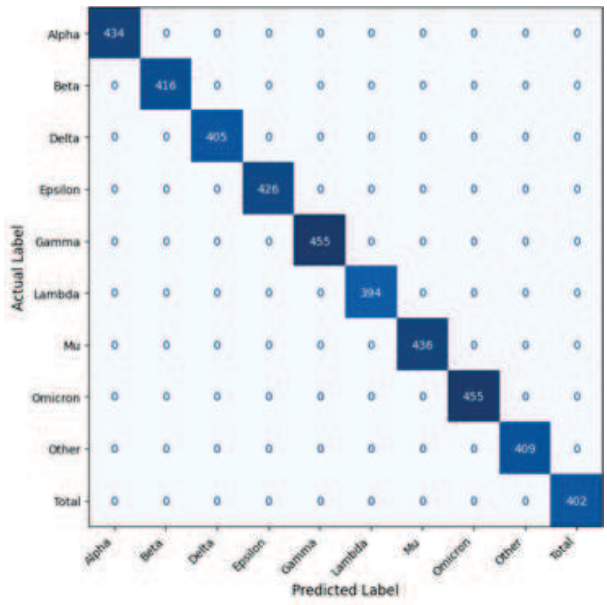


(b)

Figure 20. Confusion matrices of SC2 variants classification methods: (a) ISMI-VAE, (b) ARIMA



(c)



(d)

Figure 21. Confusion matrices of SC2 variants classification methods: (c) DL-CNN, (d) the here proposed OXAI-GECC

5. Conclusion

The OXAI-GECC method here forwarded also presents a new approach to COVID-19 classification that includes LIME-BW-PSO for feature extraction, SHAP for feature selection, and CLNN for classification. This approach helps, as well, to overcome the existing problems, appearing in the literature, by increasing the interpretability of the model, the relevance of features, and the accuracy of classification. The performance indicators of the proposed methodology were, for the specific data sets considered, 100% accuracy, 100% precision, 100% recall, and 100% F1 score. These results highlight the effectiveness of integrating explainable AI methodologies with deep learning models for gene expression data in diagnosing COVID-19. Further studies should be conducted to test this framework in other populations and include data from new variants of COVID-19. Moreover, applying this approach to other infectious diseases helped us in understanding the interactions between the host and the pathogen and develop an effective diagnostic system. Data privacy and security concerns, especially in IoT-based patient monitoring systems, are still open issues for further research to guarantee the ethical use of such technologies in clinical practice.

References

Aghalya, S., Rao, N. V., Roy, R. R., Srinivasan, C. and Premi, G. (2023) Real-time monitoring and prediction of respiratory diseases using IoT and machine learning. In: *2023 Second International Conference on Smart Technologies for Smart Nation (SmartTechCon)*. IEEE, 635-640.

Arowolo, M. O., Ogundokun, R. O., Misra, S., Agboola, B. D. and Gupta, B. (2022) Machine learning-based IoT system for COVID-19 epidemics. *Computing*, **105**(4), 831–847. https://doi.org/10.1007/s00607-022-01057-6

Deebak, B. D. and Al-Turjman, F. (2023) EEI-IoT: Edge-Enabled Intelligent IoT Framework for Early Detection of COVID-19 Threats. *Sensors*, **23**(6), 2995. https://doi.org/10.3390/s23062995

Dong, X., Matthews, D. A., Gallo, G., Darby, A., Donovan-Banfield, I., Goldswain, H., MacGill, T., Myers, T., Orr, R., Bailey, D., Carroll, M. W. and Hiscox, J. A. (2025) Using minor variant genomes and machine learning to study the genome biology of SARS-CoV-2 over time. *Nucleic Acids Research*, **53**(4). https://doi.org/10.1093/nar/gkaf077

Gürsoy, E. and Kaya, Y. (2023) An overview of deep learning techniques for COVID-19 detection: methods, challenges, and future works. *Multimedia Systems*, **29**(3), 1603–1627. https://doi.org/10.1007/s00530-023-01083-0

Hossen, Md. J., Ramanathan, T. T. and Al Mamun, A. (2024) An Ensemble Feature Selection Approach-Based Machine Learning Classifiers for Prediction of COVID-19 Disease. *International Journal of Telemedicine and Applications*, **2024**, 1–10. https://doi.org/10.1155/2024/8188904

Kar, S. and Ganguly, M. (2024) Application of genomic signal processing as a tool for high-performance classification of SARS-CoV-2 variants: a machine learning-based approach. *Soft Computing*, **28** (4), 2891–2918. https://doi.org/10.1007/s00500-023-09577-9

Khan, A., Khan, S. H., Saif, M., Batool, A., Sohail, A. and Waleed Khan, M. (2023) A Survey of Deep Learning Techniques for the Analysis of COVID-19 and their usability for Detecting Omicron. *Journal of Experimental & Theoretical Artificial Intelligence*, **36** (8), 1779–1821. https://doi.org/10.1080/0952813x.2023.2165724

Khan, H., Kushwah, K. K., Singh, S., Urkude, H., Maurya, M. R. and Sadasivuni, K. K. (2021) Smart technologies driven approaches to tackle COVID-19 pandemic: a review. *3 Biotech*, **11**(2). https://doi.org/10.1007/s13205-020-02581-y

Kumar, D., Sood, S. K. and Rawat, K. S. (2023) IoT-enabled technologies for controlling COVID-19 Spread: A scientometric analysis using CiteSpace. *Internet of Things*, 23, 100863. https://doi.org/10.1016/j.iot.2023.100863

Lee, P., Kim, H., Zitouni, M. S., Khandoker, A., Jelinek, H. F., Hadjileontiadis, L., Lee, U. and Jeong, Y. (2022) Trends in Smart Helmets With Multimodal Sensing for Health and Safety: Scoping Review. *JMIR mHealth uHealth*, **10**(11), e40797. https://doi.org/10.2196/40797

Li, H., Zhou, Y., Zhao, N., Wang, Y., Lai, Y., Zeng, F. and Yang, F. (2024a) ISMI-VAE: A deep learning model for classifying disease cells using gene expression and SNV data. *Computers in Biology and Medicine*, 175, 108485. https://doi.org/10.1016/j.compbiomed.2024.108485

Li, L., Li, C., Li, N., Zou, D., Zhao, W., Luo, H., Xue, Y., Zhang, Z., Bao, Y. and Song, S. (2024b) Machine Learning Early Detection of SARS-CoV-2 High-Risk Variants. *Advanced Science*, **11**(45). https://doi.org/10.1002/advs.202405058

Lohaj, O., Paralič, J., Bednár, P., Paraličová, Z. and Huba, M. (2023) Unraveling COVID-19 Dynamics via Machine Learning and XAI: Investigating Variant Influence and Prognostic Classification. *Machine Learning and Knowledge Extraction*, **5**(4), 1266–1281. https://doi.org/10.3390/make5040064

Matson, R. P., Comba, I. Y., Silvert, E., Niesen, M. J. M., Murugadoss, K., Patwardhan, D., Suratekar, R., Goel, E.-G., Poelaert, B. J., Wan, K. K., Brimacombe, K. R., Venkatakrishnan, A. and Soundararajan, V. (2024) A deep learning approach predicting

the activity of COVID-19 therapeutics and vaccines against emerging variants. *NPJ Systems Biology and Applications*, **10** (1). https://doi.org/10.1038/s41540-024-00471-0

Mick, E., Kamm, J., Pisco, A. O., Ratnasiri, K., Babik, J. M., Castañeda, G., DeRisi, J. L., Detweiler, A. M., Hao, S. L., Kangelaris, K. N., Kumar, G. R., Li, L. M., Mann, S. A., Neff, N., Prasad, P. A., Serpa, P. H., Shah, S. J., Spottiswoode, N., Tan, M. and Langelier, C. (2020) Upper airway gene expression reveals suppressed immune responses to SARS-CoV-2 compared with other respiratory viruses. *Nature Communications*, **11**(1). https://doi.org/10.1038/s41467-020-19587-y

Mohtasham, F., Pourhoseingholi, M., Hashemi Nazari, S. S., Kavousi, K. and Zali, M. R. (2024) Comparative analysis of feature selection techniques for COVID-19 dataset. *Scientific Reports*, **14**(1), 18627 . https://doi.org/10.1038/s41598-024-69209-6

Nasser, N., Emad-ul-Haq, Q., Imran, M., Ali, A., Razzak, I. and Al-Helali, A. (2021) A smart healthcare framework for detection and monitoring of COVID-19 using IoT and cloud computing. *Neural Computing and Applications*, **35**(19), 13775–13789. https://doi.org/10.1007/s00521-021-06396-7

Niazkar, H. R., Moshari, J., Khajavi, A., Ghorbani, M., Niazkar, M. and Negari, A. (2024) Application of multi-gene genetic programming to the prognosis prediction of COVID-19 using routine hematological variables. *Scientific Reports*, **14**(1), 2043. https://doi.org/10.1038/s41598-024-52529-y.

Qayyum, A., Benzinou, A., Saidani, O., Alhayan, F., Khan, M. A., Masood, A. and Mazher, M. (2024) Assessment and classification of COVID-19 DNA sequence using pairwise features concatenation from multi-transformer and deep features with machine learning models. *SLAS Technology*, **29**(4), 100147. https://doi.org/10.1016/j.slast.2024.100147.

Rahmadeyan, A., Mustakim, Ocviani, R., Sarah, S. and Ardiansyah, F. (2024) Classification of COVID-19 data using decision tree optimized with particle swarm optimization and genetic algorithm. In: *2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS)*. IEEE, 1-5. https://doi.org/10.1109/icetsis61505.2024.10459540

Saleh, M. A., Serte, S., Turjman, F. A., Abdulkadir, R. A., Ameen, Z. S., and Ozsoz, M. (2023) Deep learning-based feature extraction coupled with multi class SVM for COVID-19 detection in the IoT era. *International Journal of Nanotechnology*, **20**(1/2/3/4), 7–24. https://doi.org/10.1504/ijnt.2023.131109

Santosh Kumar Patra, P. and Tripathy, B. (2024) Hybrid optimal feature selection-based iterative deep convolution learning for COVID-19 classification system. *Computers in Biology and Medicine*, 181, 109031. https://doi.org/10.1016/j.compbiomed.2024.109031

Santosh Kumar Patra, P. and Tripathy, B. (2025) A Hybrid Feature Optimized Framework for Enhanced COVID-19 Detection with IoT Sensor Data. *Cuestiones de Fisioterapia*, **54**(2), 2695–2703. https://doi.org/10.48047/nxz91k56

Shahzad, A., Zafar, B., Ali, N., Jamil, U., Alghadhban, A. J., Assam, M., Ghamry, N. A. and Eldin, E. T. (2022) COVID-19 Vaccines Related User's Response Categorization Using Machine Learning Techniques. *Computation*, **10**(8), 141. https://doi.org/10.3390/computation10080141

Shi, Y., Ma, Y., Zheng, Z., Qin, Y., Du, Z. and Liu, J. (2024) Development and validation of a predicting nomogram for in-hospital mortality of COVID-19 Omicron variant: A cohort study of 1324 cases in Beijing Anzhen Hospital. *Heliyon*, **10**(7), e28627. https://doi.org/10.1016/j.heliyon.2024.e28627

Sitjar, J., Tsai, H.-P., Lee, H., Chang, C.-W., Wu, X.-N. and Liao, J.-D. (2025) Fast screening of COVID-19 inpatient samples by integrating machine learning and label-free SERS methods. *Analytica Chimica Acta*, 1350, 343872. https://doi.org/10.1016/j.aca.2025.343872

Talib, M. A., Afadar, Y., Nasir, Q., Nassif, A. B., Hijazi, H. and Hasasneh, A. (2024) A tree-based explainable AI model for early detection of Covid-19 using physiological data. *BMC Medical Informatics and Decision Making*, **24**(1), 179. https://doi.org/10.1186/s12911-024-02576-2.

Talukder, Md. A., Layek, Md. A., Kazi, M., Uddin, Md. A. and Aryal, S. (2024) Empowering COVID-19 detection: Optimizing performance through fine-tuned EfficientNet deep learning architecture. *Computers in Biology and Medicine*, 168, 107789. https://doi.org/10.1016/j.compbiomed.2023.107789

Ulvi Saygi Ayvaci, M., Jacobi, V. S., Ryu, Y., Gundreddy, S. P. S. and Tanriover, B. (2025) Clinically Guided Adaptive Machine Learning Update Strategies for Predicting Severe COVID-19 Outcomes. *The American Journal of Medicine*, **138**(2), 228-235.e1. https://doi.org/10.1016/j.amjmed.2024.10.011

Venkatachala Appa Swamy, M., Periyasamy, J., Thangavel, M., Khan, S. B., Almusharraf, A., Santhanam, P., Ramaraj, V. and Elsisi, M. (2023) Design and Development of IoT and Deep Ensemble Learning Based Model for Disease Monitoring and Prediction. *Diagnostics*, **13** (11), 1942. https://doi.org/10.3390/diagnostics13111942

Wang, J. and Safo, S. E. (2024) Deep IDA: a deep learning approach for integrative discriminant analysis of multi-omics data with feature ranking—an application to COVID-19. *Bioinformatics Advances,* **4** (1), vbae060. https://doi.org/10.1093/bioadv/vbae060.

Yadav, S. K., Mehra, M., Mishra, H. R. and Akhter, Y. (2025) Interpreting the COVID-19 infection geospatial data of Indian provinces for Alpha, Delta and Omicron variants using ARIMA and machine learning. *International Journal of Biomathematics*. https://doi.org/10.1142/s1793524525500135

Yagin, F. H., Cicek, İ. B., Alkhateeb, A., Yagin, B., Colak, C., Azzeh, M. and Akbulut, S. (2023) Explainable artificial intelligence model for identifying COVID-19 gene biomarkers. *Computers in Biology and Medicine*, 154, 106619. https://doi.org/10.1016/j.compbiomed.2023.106619

Zeng, J., Duarte, P. A., Ma, Y., Savchenko, O., Shoute, L., Khaniani, Y., Babiuk, S., Zhuo, R., Abdelrasoul, G. N., Charlton, C., Kanji, J. N., Babiuk, L., Edward, C. and Chen, J. (2022) An impedimetric biosensor for COVID-19 serology test and modification of sensor performance via dielectrophoresis force. *Biosensors and Bioelectronics*, 213, 114476. https://doi.org/10.1016/j.bios.2022.114476