PARTICLE SWARM OPTIMIZED ALMMO* FOR INTERPRETABLE AND ACCURATE DIABETIC RETINOPATHY DETECTION

Mohamed Chatra[1,3,5], Abdelouahab Attia[2,4,6], Samir Akhrouf[1,3,7], Zineb Maaref[2,4,8]

[1] *Computer Science Department, University Mohamed Boudiaf of M'sila, Algeria*
[2] *Computer Science Department, University Mohamed El Bachir El Ibrahimi of Bordj Bou Arreridj, Algeria*
[3] *Laboratory of Informatics and its Application of M'sila, Algeria*
[4] *Laboratory Materials and Electronic System of Bordj Bou Arreridj, Algeria*
[5] Mohamed.chatra@univ-msila.dz, [6] abdelouahab.attia@univ-bba.dz, [7] samir.akhrouf@univ-msila.dz,
[8]zineb.maaref@univ-bba.dz

Abstract: *Early detection of Diabetic Retinopathy (DR) is essential to reduce the risk of vision loss. This study introduces a novel framework for DR detection using a Particle Swarm Optimized Autonomous Learning Multiple Model (PSO-ALMMo) system. The proposed approach integrates the adaptive learning capability of the ALMMo* system with particle swarm optimization (PSO) to enhance classification accuracy and model interpretability. The proposed method uses PSO to optimize both antecedent and consequent parameters of the ALMMo* model, enabling high performance while maintaining the ability to learn incrementally from new data without retraining. Hybrid feature extraction techniques are applied to retinal fundus images before classification. Experiments were conducted on the newly introduced LISIA dataset as well as three benchmark datasets: Messidor-2, APTOS 2019, and IDRID. The PSO-ALMMo* system achieved 98.2% accuracy on Messidor-2, 99.7% on APTOS 2019, and 99% on IDRID. On the LISIA dataset, it maintained consistent performance across all DR severity levels. The model facilitates real-time learning and generates interpretable outcomes, owing to its prototype-based structure. These results indicate that the proposed system is well-suited for clinical environments to support early and accurate DR screening.*

Keywords: **diabetic retinopathy, particle swarm optimization, autonomous learning, ALMMo*, medical image analysis.**

1. Introduction

Diabetes necessitates continuous management; failure to do so can lead to severe complications, including vision loss, heart disease, liver damage, and kidney failure. The prevalence of diabetes is increasing, particularly among children and adolescents. This is linked to obesity, inactivity, family history, and insulin resistance.
By 2045, diabetes may affect 700 million people [1]. Preventive care, awareness, and early detection are key to limiting its spread. In 2017, the Middle East and North Africa had the second-highest prevalence worldwide, 9.2% with about 40 million cases [2]. Type 2 diabetes, the most common form, involves high blood sugar due to insulin resistance. Although it has no cure, early diagnosis can help slow its effects.
Preventing vision loss depends on early detection of eye diseases, as timely treatment significantly increases the likelihood of success and can prevent. The condition progresses to blindness.
Retinal disorders affect millions of people [3]. Many cases of vision loss can be averted through timely diagnosis and appropriate intervention.
Current diagnostic methods rely on specialists analyzing retinal images manually. This process is slow, repetitive, and susceptible to human error, often leading to inconsistent diagnostic outcomes. Therefore, there is a growing need for faster and more reliable methods to facilitate large-scale detection of retinal diseases. Figure 1, as described by Mumtaz et al. [4], presents common clinical signs of diabetic retinopathy. Microaneurysms appear as small, dark-red dots near the ends of retinal blood vessels and are among the earliest detectable signs. Retinal hemorrhages result from hypertension or venous blockage and may be mistaken for microaneurysms due to their similar appearance. Exudates are yellowish deposits composed of lipids and proteins, which leak from damaged capillaries and indicate ongoing retinal damage and the clearance of non-functional vessels.
Traditionally, doctors use fundus cameras or Optical Coherence Tomography (OCT) to capture retinal images. But reviewing these images manually is time-consuming and subjective, as noted by Qummar et al. [5].
To improve accuracy, researchers have started using ensemble Convolutional Neural Networks (CNNs) for classifying different types of retinal disease. These models perform better in identifying multiple conditions at once [6].
But accuracy is not enough. For doctors to use AI tools in real settings, they need to understand how the models work. That's why model interpretability is just as important as performance. When the process is transparent, trust in the system increases.
This research focuses on both goals: improving detection accuracy using CNNs and improving model interpretability. This helps doctors take earlier and more effective action, reducing the risk of blindness. Achieving these goals enables clinicians to make earlier and more informed decisions, thereby reducing the risk of blindness.

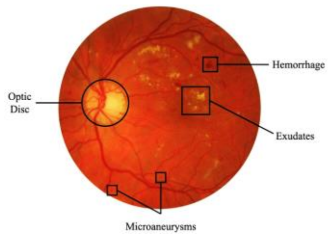


Figure 1. Fundus images containing diverse types of lesions for the classification of diabetic retinopathy (DR).

For this purpose, this study introduces an advanced DR detection framework based on the Autonomous Learning Multiple Model (ALMMo) system optimized with Particle Swarm Optimization (PSO). The ALMMo system enables recursive, non-iterative learning, allowing continuous updates without full retraining, while PSO ensures optimal tuning of model parameters [46]. The main contributions of this work can be summarized as follows:

1. PSO is employed to optimize both the antecedent and consequent parameters of the ALMMo model, ensuring global convergence and effective parameter tuning.
2. The prototype-based structure of the model provides transparent decision reasoning, which facilitates clinical trust and interpretability.
3. The proposed framework effectively addresses the limitations of conventional deep learning models by combining adaptability, interpretability, and high detection accuracy.

The paper is organized as follows: Section 2 provides related works and techniques used in this area. Section 3 presents the proposed method, knowledge exploration techniques, and datasets. It explains our proposed method and presents other existing state of the art methods used and compared to our approach in this paper. Section 4 is devoted to the presentation and discussion of the obtained results on real datasets. In conclusion, Section 5 wraps up the paper and outlines potential avenues for future research and development.

2. Related works

Several studies have addressed the challenge of automated diabetic retinopathy (DR) detection using deep learning and optimization techniques. Early detection of diabetic retinopathy (DR) is crucial because it often progresses without noticeable symptoms. Automatic detection systems improve screening efficiency, expand access to care in remote areas, and support earlier diagnosis. Many models have been developed for this purpose, but multiclass classification still poses challenges.

Several machine learning (ML) models have been tested due to their lower complexity, but they often fail to achieve high classification accuracy. To address this, researchers have used transfer learning (TL) models, which improve accuracy but introduce longer training times due to their large architectures.

Nguyen et al. [7] proposed a five-class DR classification using VGG-16 and VGG-19 on the EyePACS dataset. They achieved an accuracy of 82.0%, but classes 3 and 4 suffered due to class imbalance. Augmentation was applied, but failed to resolve this issue.

In [8], a DenseNet-121-based approach on the APTOS dataset reached 94.9% accuracy, 92.6% sensitivity, and 97.1% specificity, showing strong performance but relying on heavy architecture.

Kumar et al. [9] used ensemble classification and vessel segmentation, but lacked information about datasets and evaluation metrics, making comparisons difficult.

Alam et al. [10] combined a CNN with vessel segmentation and tested it on MESSIDOR-2, achieving 94.1% accuracy and 96.9% AUC. However, their method lacked interpretability and external validation.

Diware et al. [11] used DN121-L on MESSIDOR-2 and reached 90.3% accuracy and 81.8% F1-score. By applying self-supervised learning (BYOL) and deep ensemble model Parsa et al. [12] reported accuracies of 71.84% on the IDRiD dataset and 75.42% on MESSIDOR.

An AG-CNN model built on DenseNet-121 reached an accuracy of 98.48% accuracy and 99.8% AUC on APTOS [13]. Santos et al. [14] proposed a Siamese CNN tested on four datasets. Performance ranged from 67.23% (APTOS) to 96.85% (DIARETDB0), demonstrating dataset sensitivity.

HRUNet reached 94% accuracy on APTOS and 91% on the Kaggle dataset with 6.3 million parameters [15].

Wang et al. [16] introduced MDGNet, a GCN-based model with dynamic weight fusion, achieving 84.31% accuracy on APTOS and 81.25% on DDR. Their F1-scores showed the model’s struggle with lower-performing classes.

A residual module-enhanced Vision Transformer (ViT) attained 89.3% accuracy and 98.1% AUC on APTOS [17]. An ensemble of transformer models with attention maps scored 91.2% accuracy and 97.7% AUC on APTOS and MESSIDOR-2 [18, 19]. Chetoui et al. [20] fine-tuned Inception-ResNet-v2 and tested it on eight datasets, achieving up to 99.0% AUC. Ioannou et al. [21] evaluated DenseNet, InceptionV3, and EfficientNetB0, with EfficientNetB0 scoring 73.56% on DDR and 60.19% on IDRiD.

Jang et al. [22] proposed an explainable classification model using a segmentation-driven symbolic representation, scoring 63.11% accuracy on the Kaggle dataset. Shorfuzzaman et al. [23] applied SHAP and Grad-CAM for interpretability, achieving 98.0% sensitivity and 97.8% AUC across APTOS, MESSIDOR, and IDRiD datasets.
A hybrid CNN-SVD model combined with ISVM-RBF, DT, and KNN was evaluated on IDRiD, reaching 99.18% accuracy and 100% specificity in vision-threatening DR detection [24]. 'NIMEQ-SACNet' [25] combined a quantum-enhanced optimizer with a capsule network for VTDR classification.
Bilal et al. [26] used a hybrid CNN-SVD with ensemble classifiers and achieved 99.89% accuracy and 100% specificity on IDRiD. Singh et al. [27] compared ML classifiers on the DRIVE dataset, with the Decision Tree scoring 96.24% accuracy. An EPO-BFO hybrid optimization approach for feature selection improved ML classification performance with 96.5% accuracy [28]. Three CNN models (one custom, one ensemble, one CNN-LSTM) were evaluated across three Kaggle datasets, reaching 95.45% accuracy and 97.69% AUC [29].
In addition to deep and hybrid learning models, recent research has explored nature-inspired optimization algorithms to fine-tune model parameters and improve classification performance. Particle Swarm Optimization (PSO), in particular, has gained attention for its ability to enhance feature selection, hyperparameter tuning, and classifier performance. For example, in [43], PSO was used to optimize the structure of a CNN for diabetic retinopathy classification, resulting in improved convergence and accuracy. Similarly, [44] applied PSO to enhance feature extraction from fundus images, demonstrating better precision and reduced computational cost. In [45], a PSO-optimized support vector machine (SVM) was proposed for DR detection, which outperformed standard SVM models in terms of sensitivity and AUC.
In recent work, Particle Swarm Optimization (PSO) has been applied to feature selection and hyper-parameter tuning. Kennedy and Eberhart [41] introduced PSO as a population-based search algorithm, and later studies applied it to optimize deep learning pipelines for medical imaging tasks.
Our method builds on these efforts by combining PSO-based parameter tuning with the ALMMo* classifier. It focuses on improving diagnostic accuracy and maintaining model transparency across multiple datasets, including private clinical ultrasound data collected from real-world environments.

Table 1. summary table of the models and results reported by related works on diabetic retinopathy detection

Study	Model/Method	Dataset	Performance	Challenges/Notes
Nguyen et al. [7]	VGG-16, VGG-19	EyePACS	82.0% accuracy	Class imbalance in classes 3 and 4, augmentation failed to resolve the issue.
DenseNet-121-based approach [8]	DenseNet-121	APTOS	94.9% accuracy, 92.6% sensitivity	Heavy architecture, strong performance, but resource-intensive.
Kumar et al. [9]	Ensemble classification, vessel segmentation	N/A	N/A	Lack of dataset details and evaluation metrics.
Alam et al. [10]	CNN with vessel segmentation	MESSIDOR-2	94.1% accuracy, 96.9% AUC	Lacked interpretability and external validation.
Diware et al. [11]	DN121-L	MESSIDOR-2	90.3% accuracy, 81.8% F1-score	No additional details on evaluation metrics.
Parsa et al. [12]	BYOL self-supervised learning, deep ensembles	IDRiD, MESSIDOR	71.84% - 75.42% accuracy	Lower accuracy compared to other models.
AG-CNN model [13]	AG-CNN (DenseNet-121)	APTOS	98.48% accuracy, 99.8% AUC	High accuracy, relies on DenseNet-121 architecture.
Santos et al. [14]	Siamese CNN	Four datasets	67.23% - 96.85% accuracy	Performance varies significantly between datasets.
Wang et al. [16]	MDGNet (GCN-based model)	APTOS, DDR	84.31% - 81.25% accuracy	Struggled with lower-performing classes; F1-scores were low.
Chetoui et al. [20]	Inception-ResNet-v2 fine-tuning	Eight datasets	Up to 99.0% AUC	High AUC, lacks detailed accuracy metrics.
Shorfuzzaman et al. [23]	SHAP, Grad-CAM interpretability	APTOS, MESSIDOR, IDRiD	98.0% sensitivity, 97.8% AUC	Focus on interpretability and high sensitivity.
Hybrid CNN-SVD model [24]	CNN-SVD with ISVM-RBF, DT, KNN	IDRiD	99.18% accuracy, 100% specificity	High accuracy for vision-threatening DR detection.
NIMEQ-SACNet [25]	Quantum-enhanced optimizer, capsule network	N/A	N/A	Focus on optimization for VTDR classification.
Bilal et al. [26]	Hybrid CNN-SVD with ensemble classifiers	IDRiD	99.89% accuracy, 100% specificity	High accuracy and specificity.
Singh et al. [27]	ML classifiers (Decision Tree)	DRIVE	96.24% accuracy	Strong performance, though single model evaluation.
PSO-enhanced feature selection [30][31][32]	PSO for CNN, SVM	Fundus images	Improved convergence, accuracy	PSO enhances feature extraction and model performance. Reduces computational cost.

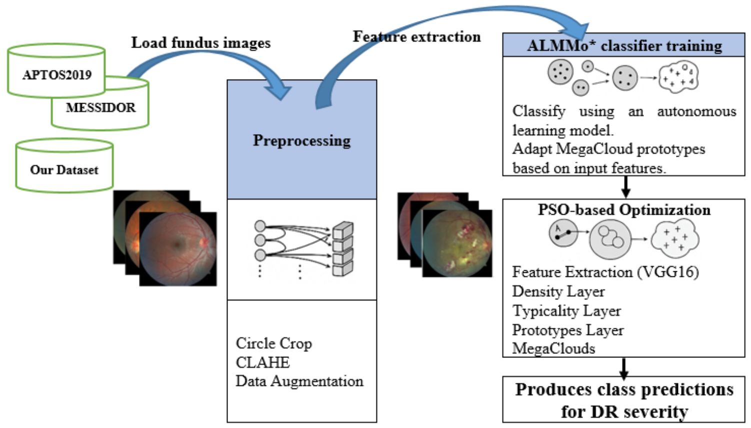
3. Methodology

Deep learning models are often characterized by opaque decision-making processes, high data demands, intensive computation, long training times, and significant energy use. These drawbacks limit their use in environments with

limited resources, such as rural clinics or mobile health units, and raise concerns about model transparency and environmental impact.
To overcome these limitations, the proposed approach integrates PSO with the Autonomous Learning Multiple Model (ALMMo*) system. Rather than focusing solely on accuracy, this integration emphasizes structural clarity and adaptability. The ALMMo* model organizes learning into distinct functional layers that self-adjust as new data becomes available, while PSO dynamically refines both structural and parametric components to achieve optimal model performance. PSO enhances this structure by generating prototype-based IF...THEN rules that offer transparent diagnostic reasoning. These rules are readable and help users trace and understand the decision-making process, supporting clinical validation.
The approach uses a feedforward PSO algorithm alongside ALMMo*'s incremental learning mechanism, enabling dynamic structural updates as new data becomes available. New prototypes are added over time, allowing the model to adapt to changes in data distribution without retraining. This makes the system well-suited for real-world scenarios where data evolves continuously.

3.1. Architecture

The proposed system for diabetic retinopathy classification begins by loading fundus images from datasets such as MESSIDOR, APTOS, or the LISIA dataset. The images undergo preprocessing that includes applying a circular mask to remove non-retinal areas, enhancing contrast using CLAHE, and standardizing image size and intensity. Next, deep features are extracted using the VGG16 model with the final classification layer removed. These features are then passed to the ALMMo* classifier, which uses an autonomous learning strategy to create and adjust MegaCloud prototypes. Each prototype adapts to the input data and links to specific retinal lesions, allowing for both flexible classification and interpretability. To further improve accuracy, PSO is used to tune the parameters of ALMMo*. Each DR class has its own PSO instance to define optimal boundaries. The system outputs a final prediction for DR severity across seven classes: No DR, Mild, Moderate, Severe, Very Severe, PDR, and Advanced PDR, along with confidence scores and prototype-based explanations.



Load fundus images
Feature extraction
APTOS2019
MESSIDOR
Our Dataset
Preprocessing
Circle Crop
CLAHE
Data Augmentation
ALMMo* classifier training
Classify using an autonomous learning model.
Adapt MegaCloud prototypes based on input features.
PSO-based Optimization
Feature Extraction (VGG16)
Density Layer
Typicality Layer
Prototypes Layer
MegaClouds
Produces class predictions for DR severity

Figure 2. shows the full workflow for diabetic retinopathy (DR) severity classification.

Step	Description
Input	Load fundus images from DR datasets (MESSIDOR, APTOS, LISIA).
Preprocessing	Apply a circular mask to remove non-retinal areas. Enhance contrast using CLAHE. Normalize and reshape images.
Feature Extraction	Use VGG16 without the final layer. Extract deep features that represent retinal structures.
ALMMo*	Classify using an autonomous learning model. Adapt MegaCloud prototypes based on input features. Learn class distributions without fixed boundaries. Link each prototype to lesions for interpretability.
PSO Optimization	Each class has a separate PSO instance. Particles represent ALMMo* parameter sets. Update particles iteratively to reduce classification error.
Output	Assign one of the 7 DR severity classes: No DR, Mild, Moderate, Severe, Very Severe, PDR, Advanced PDR Also, provide confidence scores and a visual explanation through prototypes

3.1.1. Dataset description

Fundus images are loaded from diabetic retinopathy (DR) datasets such as MESSIDOR-2, APTOS, IDRID, or the locally collected LISIA dataset. These images represent different stages of DR and serve as the starting point for the classification system. Each image is labeled with a known DR severity level, which will be used later for supervised learning and evaluation.

A. MESSIDOR

The MESSIDOR database contains 1200 color fundus images in TIFF format, captured at three ophthalmology centers with a Topcon TRC NW6 non-mydriatic imaging system featuring a 45-degree field of view. Each image has 8 bits per color channel and comes in one of three resolutions: 1440×960, 2240×1488, or 2304×1536 pixels. Out of the 1200 images, 800 were captured with pupil dilation using 0.5% Tropicamide, while 400 were taken without dilation. Each image is labeled with a diabetic retinopathy (DR) grade. The labels divide the images into five DR severity levels, as shown in figure 3(a). The dataset is split into three sets, each from a different medical center. Each set contains four zipped folders of 100 images, along with an Excel file that includes the diagnosis for each image. While the database includes clinical DR grades, it does not provide manual annotations such as lesion outlines or locations. This makes it different from other datasets but still valuable for developing Computer-Assisted Diagnosis (CAD) systems for DR [31].

B. APTOS

The APTOS dataset, available on Kaggle, contains 3662 color fundus images captured using different cameras. This results in varying resolutions and image quality, reflecting real-world conditions found in clinical settings. The dataset was curated by Aravind Eye Hospital in India and is supported by the Asia Pacific Tele-Ophthalmology Society (APTOS). It is designed to support the development and evaluation of automated diabetic retinopathy (DR) detection systems. Images are classified into five DR severity levels: No DR, Mild, Moderate, Severe, and Proliferative DR, as shown in figure 3(b). Expert annotations are provided for the training set. The dataset has a class imbalance. Most images are normal (1805), while severe NPDR cases are limited (183). This imbalance can bias training and often requires techniques such as oversampling, undersampling, or using weighted loss functions. The variation in camera settings makes the dataset useful for testing the robustness and generalization of DR classification models. It is widely used in research and competitions focused on AI applications in ophthalmology.

C. IDRID

The Indian Diabetic Retinopathy Image Dataset (IDRID) includes 516 high-resolution fundus images from diabetic patients. It is publicly available and widely used in diabetic retinopathy (DR) research. The dataset is split into training and testing sets. Each image is labeled with a DR severity level and includes detailed annotations of key lesions, microaneurysms, hemorrhages, soft exudates, and hard exudates, as shown in figure 3(c). The images were captured using high-resolution fundus cameras. They reflect clinical variability, which helps train more robust models. IDRID supports multiple tasks, including DR grading, lesion detection, and segmentation. Its quality and comprehensive annotations make it a valuable resource for developing and evaluating automated DR detection systems. Researchers use it to build and test CAD tools aimed at improving early diagnosis and treatment.

D. Private dataset

A total of 1310 fundus images were collected from patients over 18 years old at the clinic of ophthalmology, LISIA-Algeria. Images are classified into five DR severity levels: No DR, Mild, Moderate, Severe, and Proliferative DR, as shown in figure 3(d). The image acquisition followed a consistent clinical protocol:

1. Each patient was interviewed, and their basic medical data were recorded in a patient file.
2. Mydriatic eye drops (Fotorretin: Phenylephrine 5% and Tropicamide 0.5%) were applied every 15 minutes in both eyes, twice. After 30 minutes, once pupil dilation was complete, the fundus examination was performed in a dark room.
3. The EIDON FA was used to capture one image from each eye. The patient was properly seated, with the forehead and chin supported to ensure stability during image acquisition. This model, equipped with Fluorescein Angiography capability, is the latest top-class model in the EIDON family. It incorporates the core EIDON technology and provides ultra-high-resolution images and videos of fundus fluorescein angiography, enabling the dynamic study of retinal vasculature.
4. A follow-up fundoscopic exam was conducted using an indirect ophthalmoscope and a 20D magnifying lens. The DR stage was determined and recorded in the patient's medical file.
5. Only high-quality images were retained. Any images that were out-of-focus, had visual artifacts, or showed poor quality due to media opacity (such as corneal scars, cataracts, or vitreous issues like hemovitreous or vitritis) were excluded. Patients with other retinal diseases were also excluded from the dataset.
6. Our images are JPG format (.JPG) with high precision dimensions: 3680*3288.

The MESSIDOR, APTOS, IDRID, and our private dataset are key resources in retinal disease research. Each provides labeled fundus images with clear ground truth, enabling accurate evaluation and comparison of detection and classification algorithms. These datasets support the development of Computer-Aided Diagnosis (CAD) systems and AI-based tools aimed at identifying diabetic retinopathy and other retinal conditions. Their availability and structure make them essential for building reliable, generalizable models in ophthalmology (figure 4).

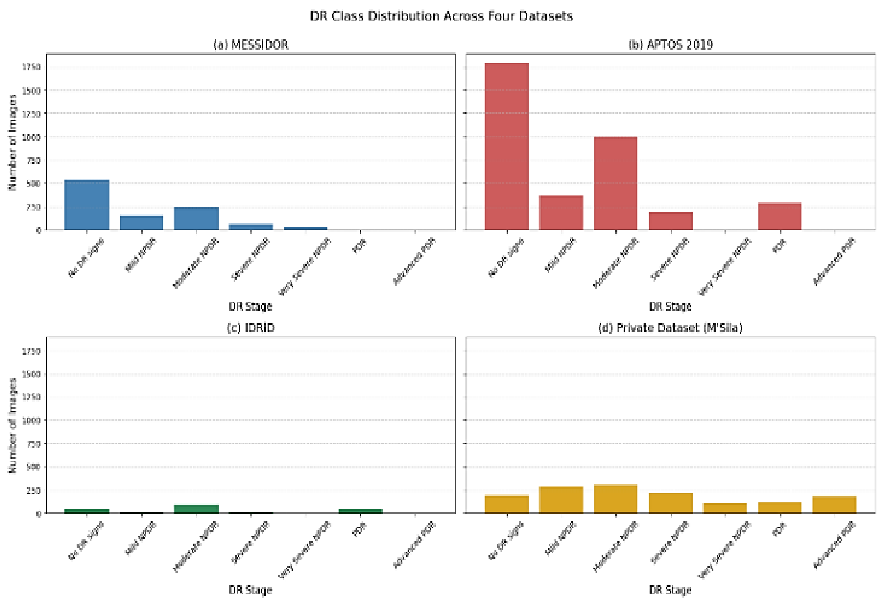


Figure 3. DR class distribution across four datasets

3.2. Preprocessing

Fundus images often vary due to differences in hardware, lighting, and image capture conditions. Preprocessing helps standardize these images and reduce noise, making them more suitable for analysis. Key steps include resizing, cropping, contrast enhancement, normalization, and data augmentation. These adjustments improve image quality and ensure that classification models focus on the important features. As a result, the models perform more reliably, even when images differ in quality or appearance. Most diabetic retinopathy datasets contain images with different resolutions, aspect ratios, and large black border areas. To prepare the data, images are cropped and resized to a uniform size. CLAHE (Contrast Limited Adaptive Histogram Equalization) is used to enhance contrast and reveal important details in areas with uneven lighting. The clip limit in CLAHE is carefully tuned to prevent over-amplifying noise.

To further improve model robustness, data augmentation techniques such as center cropping, horizontal flipping, and vertical flipping are applied. These techniques help the model generalize better by exposing it to more varied image conditions during training.

Table 1. displays examples of fundus images across different DR stages.

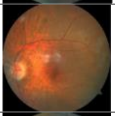
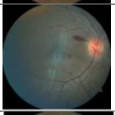
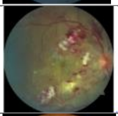
No DR signs: Images showing no visible signs of diabetic retinopathy.	
Mild (Characterized by a few microaneurysms (small bulges in blood vessels). Often asymptomatic and only detected through fundus photography)	
Moderate (Includes more microaneurysms, dot and blot hemorrhages, and hard exudates. Capillary blockage may start to occur).	
Severe (Marked by extensive retinal hemorrhages in all quadrants, venous beading in at least one quadrant, and prominent intraretinal microvascular abnormalities (IRMA). The risk of progression to PDR increases).	
Very severe (Defined by two or more of the above severe signs. This stage has a high probability of advancing to PDR within a short period)	
PDR: Images with visible neovascularization or early proliferative changes (New vessels appear on the retina or optic disc (neovascularization), but without major complications. These vessels can bleed or cause fibrous tissue to form).	
Advanced PDR: Images showing complications like vitreous hemorrhage or fractional retinal detachment (May include vitreous hemorrhage, fractional retinal detachment, or endovascular glaucoma. Vision loss at this stage can become permanent without timely treatment).	

Table 1 shows example images before and after preprocessing. These examples highlight how the applied steps improve image clarity and prepare the data for accurate classification. The images are organized into seven folders based on the classification.

Algorithm 1 outlines the sequence of preprocessing operations used in our study.

```
Algorithm 1 Preprocessing
Input: Fundus image
Output: Preprocessed fundus image
    1. Read the input image
    2. Apply circular cropping to remove black borders
    3. Create CLAHE with:
           Clip Limit = 4
           Tile Grid Size = (18, 18)
    4. Apply CLAHE to the cropped image
    5. Apply data augmentation (center crop, horizontal flip, vertical flip) to training images
```

3.3. Feature extraction

Extracting relevant features from fundus images is a key step in diabetic retinopathy classification. Instead of training a deep model from the beginning, a transfer learning approach is used to benefit from existing knowledge in pretrained convolutional neural networks.

VGG16, a CNN developed by the Visual Geometry Group at the University of Oxford, is used in this study. It includes 16 layers and applies small 3×3 filters. The model was originally trained on the ImageNet dataset, which contains a large variety of natural images.

In this framework, only the convolutional layers of VGG16 are used. The fully connected layers responsible for classification in the original model are removed. This transforms VGG16 into a fixed feature extractor.

Each image is passed through the truncated VGG16, producing a 7×7×512 tensor from the last convolutional layer. This tensor is flattened into a 25,088-dimensional vector. A fully connected layer subsequently reduces this to a 4096-dimensional feature vector.

The resulting fixed-length feature vector captures the most informative characteristics of the image. These features are used in the next stage of the framework for classification.

Using pretrained ImageNet weights helps avoid overfitting and reduces the need for large training datasets. These weights capture general image features such as edges, textures, and shapes. This transfer of knowledge improves model performance in medical image analysis.

The ImageNet-trained VGG16 was found to be sufficient for fundus image representation and, fine-tuning was not required. This choice reduces training time and computational cost while maintaining high accuracy.

3.4. ALMMo* classification

This work employs a modified version of the ALMMo system, denoted as ALMMo*, which differs from the original model in two key ways:

1. **Gaussian kernel function** replaces the original Cauchy kernel. This improves sensitivity to unfamiliar patterns and supports better generalization.
2. **Historical data pool** retains processed samples, which are later used for parameter optimization.

The structure of ALMMo* consists of a rule base with *N* linear models, each represented by a prototype-based fuzzy rule. These rules are identified online through an autonomous learning process.

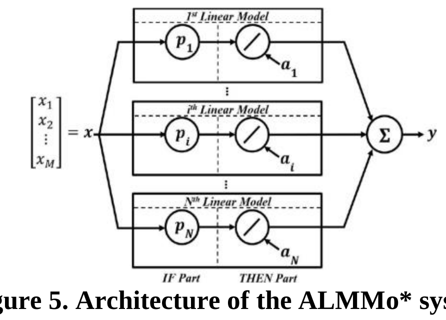


Figure 5. Architecture of the ALMMo* system.

Most first-order EISs [47][48], employ the Gaussian kernel function. In comparison with the Cauchy kernel function (which is originally adopted in [7]), the Gaussian is more compact and sensitive to outliers, and thus it helps improve the ability of the ALMMo system to handle unfamiliar data patterns, enhancing its online learning capability. The historical data saved in the data pool will be used for the system parameter optimization. To distinguish the current version from the original, the ALMMo system used herein is denoted as ALMMo*. *ALMMo** uses Gaussian kernel-based fuzzy rules to adaptively classify data. Each rule's parameters are optimized via PSO (see Eqs. 1-4 for details).

A. *System architecture*

The architecture of ALMMo* is illustrated in Fig. 1. It comprises N linear models generated through its self-organizing learning mechanism. identified. Each model is formulated using a prototype-based fuzzy rule as described in

$$R_i\colon\ IF\ (x \sim p_i)\ THEN\ (y_i = x^{-T} a_i) \qquad (1)$$

where $i = 1, 2, \ldots, N$; N denoted the total number of linear models (fuzzy rules); The input vector of a fuzzy rule is given by $\boldsymbol{x} = [x_1, x_2, \ldots, x_M]^T \in \mathfrak{R}^M$ associated with each fuzzy rule; $\bar{x} = [1, x^T]^T$; $\mathfrak{R}^M$ is an M dimensional real data space; y_i is the output of $\boldsymbol{R}_i$; and $\boldsymbol{p}_i = [p_{i,1}, p_{i,2}, \ldots, p_{i,M}]^T$ together with $\boldsymbol{a}_i = [a_{i,0}, a_{i,1}, \ldots, a_{i,M}]^T$ *denote* the prototype and consequent parameter vectors, respectively [49].

The ALMMo* model is mathematically expressed as

$$y = f(x) = \sum_{i=1}^{N} \lambda_i y_i = \sum_{i=1}^{N} \lambda_i\, x^{-T} a_i \qquad (2)$$

where λ_i denotes the firing strength of the ith fuzzy rule, $\boldsymbol{R}_i$ satisfies

$$\lambda_i = \frac{D_i(x)}{\sum_{j=1}^{N} D_j(x)} \qquad (3)$$

In this context, $D_i(\boldsymbol{x})$ refers to the Gaussian kernel–based local density of $\boldsymbol{x}$ calculated within the data cloud $\mathbf{C}_i$ centered around$\boldsymbol{p}_i$ from neighboring samples, following a structure similar to Voronoi tessellation [50]

$$D_i(x) = e^{-\frac{\|x - \hat{p}_i\|^2}{2\left(\hat{X}_i - \|\hat{p}_i\|^2\right)}} \qquad (4)$$

In (4), ||·|| represents the Euclidean norm; $\hat{p}_i$ and $\hat{X}_i$ are determined by

$$\hat{p}_i = \frac{S_i}{S_i+1} p_i + \frac{1}{S_i+1} x \qquad (5a)$$

$$\hat{X}_i = \frac{S_i}{S_i+1} X_i + \frac{1}{S_i+1} \|x\|^2 \qquad (5b)$$

Here, $\mathrm{p_i}$ is the prototype of rule $\mathrm{R_i}$ and $\mathrm{p_i} = \left({}^{1}/_{\mathrm{S_i}}\right) \sum_{\mathrm{x \in C_i}} \mathrm{x}$, $\mathrm{X_i}$ denotes the mean of the squared Euclidean norm of data samples belonging to $\mathrm{C_i}$. and $\mathrm{X_i} = \left({}^{1}/_{\mathrm{S_i}}\right) \sum_{\mathrm{x \in C_i}} \|\mathrm{x}\|^2$, and Si indicates the number of samples forming $\mathrm{C_i}$. The firing strength computed from (4) enables ALMMorapidly adjust to variations in data distribution by moving $\mathrm{p_i}$toward new observations, thereby improving the system's adaptability. ***Online learning process.***

The online self-adaptive learning procedure of ALMMo* can be outlined as follows: *Stage 1 (System Initialization):*

The initialization of the ALMMo* system with the first incoming data sample $\boldsymbol{x}_k$ ($k = 1$). During this stage, the system's global metaparameters the global mean μ, and average squared Euclidean norm X are initialized as

$$\mu \leftarrow x_k; X \longleftarrow \|x_k\|^2 \qquad (6)$$

The initial data cloud $\mathbf{C}_N$ ($N = 1$) is established with $\boldsymbol{x}_k$ as its prototype

$$C_N \longleftarrow \{x_k\};\ p_N \longleftarrow x_k;\ X_N \longleftarrow \|x_k\|^2$$
$$S_N \longleftarrow 1;\ \Lambda_N \longleftarrow 0;\ \eta_N \longleftarrow 1;\ I_N \longleftarrow k \qquad (7)$$

where Λ_N represents the cumulative firing strength; η_N corresponds to the utility measure; and I_N indicates the time step at which p_N is determined. The consequent parameters, a_N and a covariance matrix, $\mathbf{\Theta}_N$ using

$$a_N \longleftarrow 0_{(M+1)\times 1};\ \mathbf{\Theta}_N \longleftarrow \mathbf{\Theta}_0 I_{(M+1)\times(M+1)} \qquad (8)$$

The initial fuzzy rule in the rule base is defined as follows

$$R_N\colon\ IF\ (x \sim p_N) THEN\ (y_N = x^{-T} a_N) \qquad (9)$$

where $N = 1$. It is worth noting that *_o* is an externally tuned parameter used for initializing the covariance matrix, consistent with practice in recursive least-squares algorithms [48]. The initial data sample and its corresponding target, {***xk***, *yk*} are stored in the data pool for later reference.

Stage 2 (System Output Generation):

When a new data instance, $\boldsymbol{x}_k$ $(k \leftarrow k+1)$ arrives, the local density $D_i(\boldsymbol{x}_k)$ *for each data cloud* $\mathbf{C}_i$ (i =1, 2, . . . ,N) is first computed using Equation (4). Subsequently, the firing strength λ_i(i = 1, 2, . . . ,N) *associated with* each fuzzy rule is determined according to Equation (3). The system output ˆ*yk* = *f* (*xk*) is then produced following Equation (2).

Stage 3 (Updating of Global Parameters):

The global statistical parameters, namely the mean $\boldsymbol{\mu}$ and the average squared Euclidean norm, χ are updated as follows:

$$\mu \leftarrow \frac{k-1}{k} \mu + \frac{1}{k} x_k \qquad (10a)$$

$$x \leftarrow \frac{k-1}{k} x + \frac{1}{k} \|x_k\|^2 \qquad (10b)$$

Stage 4 (System Structure Updating):

In this stage, first, the global densities at $\boldsymbol{x}_k$ and $\{\boldsymbol{p}\}N$ are calculated by (11) [in a similar form to (4)]

$$D_{(z)} = e^{-\frac{\|z-\mu\|^2}{2(x-\|\mu\|^2)}} \quad (11)$$

where $\boldsymbol{z} \in \{\boldsymbol{x}_k, \boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N\}$.

Condition 1 is checked is evaluated to determine whether the new observation ***xk*** *should be considered* as a potential prototype.

$$\text{Condition 1: } If\ \left(D(x_k) < \min_{i=1,2,\ldots,N}\left(D(p_i)\right)\right)\ Or\ \left(D(x_k) > \max_{i=1,2,\ldots,N}\left(D(p_i)\right)\right)\ Then\ (x_k\ is\ a\ new\ prototype). \quad (12)$$

When $D(x_k)$ exceeds the global density associated with the existing prototypes $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N\}$, it indicates that $\boldsymbol{x}_k$ *provides a stronger* representation of the data and possesses higher summarization ability. On the other hand, if $D(x_k)$ is bellow among the current prototypes, $\boldsymbol{x}_k$ corresponds to a distinct data pattern not captured by any previously identified prototypes. Therefore, if either condition is fullfiled , $\boldsymbol{x}_k$ is regarded as a new prototype and serves to initialize a new data cloud.

After condition 1 is satisfied, Condition 2 is checked to eliminate any neighboring data cloud whose influence region significantly overlaps with newly created cloud initialized by $\boldsymbol{x}_k$

$$Condition\ 2: If\ (D_i(x_k) \geq D_o)\ Then\ (x_k\ is\ very\ close\ to\ p_i) \quad (13)$$

where $D_o = e^{-(1/8)}$. The reasoning behind this condition is that if $D_i(x_k)$ is exceeds $e^{-(1/8)}$, the Euclidean distance between x_k and p_i becomes smaller than half of the average distance between any two data samples within $\mathbf{C}_i$. In such a case, x_k and p_i are located in very close proximity. Therefore, Therefore, to prevent redundancy, it is intuitive to replace, ***pi*** withx_k [51].

If both Conditions 1 and 2 are satisfied, the closest data cloud denoted by $\mathbf{C}_{n*}$ regarding x_k is , is determined as follows::

$$n *= \arg\min_{i=1,2,\ldots,N}(\|x_k - p_i\|) \quad (14)$$

Subsequently, x_k *is incorporated into* $\mathbf{C}_{n*}$ and its meta-parameters are updated according to:

$$\begin{gathered} C_{n*} \leftarrow C_{n*} \cup \{x_k\};\ p_{n*} \leftarrow \frac{p_{n*} + x_k}{2} \\ X_{n*} \leftarrow \frac{X_{n*} + \|x_k\|^2}{2}\ ;\ S_{n*} \leftarrow \left\lceil \frac{S_{n*} + 1}{2} \right\rceil \end{gathered} \quad (15)$$

where $\lceil . \rceil$ denotes the ceiling operation.

If only Condition 1 holds,, a new data is create with x_k as its prototype ,increasing the total number of clouds ($N \leftarrow N + 1$) following (7). The corresponding fuzzy rule $\boldsymbol{R}_N$ is initialized in the same structure as (9) with the consequent parameters initialized as:

$$a_N \leftarrow \frac{1}{N-1}\sum_{i=1}^{N-1} a_i\ ;\ \boldsymbol{\Theta}_N \leftarrow \Omega_0 . \mathrm{I}_{(M+1)\times(M+1)} \quad (16)$$

If Condition 1 is not met, the observation x_k is assigned to the nearest data cloud, C_{n*} whose meta parameters are updated using

$$\begin{gathered} S_{n*} \leftarrow S_{n*} + 1;\ p_{n*} \leftarrow \frac{S_{n*} - 1}{S_{n*}}\ p_{n*} + \frac{1}{S_{n*}} x_k \\ X_{n*} \leftarrow \frac{S_{n*} - 1}{S_{n*}}\ X_{n*} + \frac{1}{S_{n*}} \|x_k\|^2 \end{gathered} \quad (17)$$

For the data clouds that do not receive new samples at the current iteration, their meta parameters remain unchanged.

Stage 5 (Rule-Base Quality Monitoring):

At this stage, the firing strength of each fuzzy rule, λi ($i = 1, 2, \ldots, N$) as in (3). The accumulated firing strength of each fuzzy rule is then updated as:

$$\Lambda_i \longleftarrow \Lambda_i + \lambda_i \quad (18)$$

and the utility measure is according to:

$$\eta_i \longleftarrow \frac{\Lambda_i}{k - I_i} \quad (19)$$

Next, Condition 3 is assessed to eliminate fuzzy rules and the corresponding data clouds that provide negligible contribution little to the system output.

Condition 3: If *($\eta i < \eta o$)* Then

*(**Ri** and* $\mathbf{C}i$ *are removed)* (20)

Here, ηo externally defined threshold used for monitoring rule quality.

When ***Ri*** and $\mathbf{C}i$ *meet* this condition 3 and are removed from ALMMo*, $N \leftarrow N - 1$, and the firing strengths of the remaining rules fuzzy rules are calculated using (3).

Stage 6 (Consequent Parameter Updating):

The consequent parameters of the fuzzy rules are updated using the popular FWRLS method as given in [52] ($i = 1, 2, \dots, N$)

$$\mathbf{\Theta}_i \leftarrow \mathbf{\Theta}_i - \frac{\lambda_i \mathbf{\Theta}_i \bar{x}_k \bar{x}_k^T \mathbf{\Theta}_i}{1 + \lambda_i \bar{x}_k^T \mathbf{\Theta}_i \bar{x}_k} \tag{21a}$$

$$\mathbf{a}_i \leftarrow \mathbf{a}_i + \lambda_i \mathbf{\Theta}_i \bar{x}_k (y_k - \bar{x}_k^T \mathbf{a}_i) \tag{21b}$$

Stage 7 (Rule Base Updating):

In the final phase of each processing iteration, all fuzzy rules in the rule base are updated in accordance with the the latest prototypes and consequent parameters. The current data sample and the corresponding pair, $\{xk, yk\}$ are then stored in the data pool for future use. After this step, ALMMo* framework returns to stage 1, prepared to process the next incoming sample. The flowchart of the learning process of ALMMo* is illustrated in Fig. 2 for better visualization.

The two externally tuned parameters Ω_0 in (8) and ηo in (20) have a subtle yet significant influence the performance of the ALMMo model. The parameter Ω_0 is employed during the initialization of the covariance matrix, while ηo concerns the quality of fuzzy rules. In practice, Ω_0 affects the convergence behavior of the consequent part of the system and negatively reduce the system performance. Likewise, *ηo regulates the frequency of rule elimination a higher ηo* accelerates the removal of obsolete rules, but an overly high value may degrade performance by causing the system to forget previously acquired information too quickly.

Significantly, if ηo is set too large since the system forgets the learned knowledge from historical data rather rapidly. The influence of Ω_0 and ηo has been analyzed and verified through numerical examples in [7] and [28]. The recommended values of Ω_0 and ηo are 10 and 0.1, respectively.

3.4.1. Optimizing ALMMo* by PSO

ALMMo* uses a one-pass learning process that ensures computational efficiency however, but it may lead to suboptimal solutions, thereby limiting the system's overall accuracy. To overcome this, a PSO-based optimization method is introduced to adjust both the premise and consequent parts of the fuzzy rules. The approach uses a basic single-objective PSO algorithm, though more advanced versions can be applied. The optimization process follows structured steps to enhance model performance using historical data [53] [54] see figure 6.

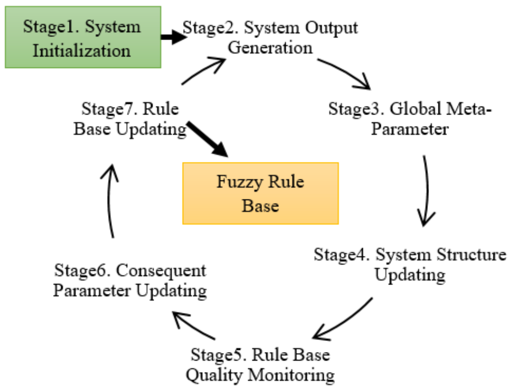

Stage1. System Initialization
Stage2. System Output Generation
Stage3. Global Meta-Parameter
Stage4. System Structure Updating
Stage5. Rule Base Quality Monitoring
Stage6. Consequent Parameter Updating
Stage7. Rule Base Updating
Fuzzy Rule Base

Figure 6. Schematic representation of the ALMMo* learning process.

Stage 1 (Swarm Initialization):

Initially,, a swarm **S**$t = \{P_1^t,\ P_2^t, \dots, P_L^t\}$, containing L particle is created, , where t denotes the current iteration ($t \leftarrow 0$). Each particle encodes a complete set of antecedent and consequent parameters corresponding to the fuzzy rules identified during the online training process representing a potential though not necessarily optimal solution. Since PSO explores the search space in semirandom manner, the leading particle, P_1^t can be defined using Equation (22) to ensure that the overall performance of ALMMo* system remains stable after the optimization process

$$P_1^t = [P_1^t, a_1^t]_{N\times 2(M+1)} \tag{22}$$

where $P_1^t = [P_1, P_2, \dots, P_N]\ T$ denotes the $N \times M$ antecedent parameter matrix (prototypes); and $a_1^t = [a_1, a_2, \dots, a_N]T$ *corresponds to* the $N \times (M + 1)$ consequent parameter matrix.

During initialization, the maximum displacement that the remaining particles can travel from the position of the leading particle, $\mathbf{P}_1$ within the searched space $\Re^{N\times 2(m+1)}$, is determined by

$$Q = [\beta^T, |\beta^T, \dots, \beta^T]^T_{N\times 2(M+1)} \tag{23}$$

Where $\beta = [\beta_1, \beta_2, \dots, \beta_{2(M+1)}]_{1\times 2(M+1)}$ and there is

$$\beta_j = \max_{i=1,2,\dots,N} (|P_1^t(i,j)|) \tag{24}$$

with $P_1^t(i, j)$ *represents* the element located in the *i*th row and the *j*th column of P_1^t. The other particles in the swarm are subsequently defined as (i = 2, 3, . . . , L)

$$P_1^t = [P_1^t, a_1^t]_{N\times 2(M+1)}^T = P_1^t + r_i \circ Q \qquad (25)$$

Where "∘" denotes Hadamard product; r_i is an N×2(M+1) dimensional random matrix whose elements follow the uniformly distributed within the interval [−1, 1]; (23) defines a local searching range centered around the leading particle, $P_{1,}^t$ while equation (25) ensures that the remaining particles randomly in the defined local area. As a result, the search process becomes more efficient due to the restricted and focused search space. To assess the fitness of P_i^t (i = 1, 2, . . . , L), the data space resegmented by treating P_i^t as the set of prototypes to attract nearby historical data samples forming Voronoi according to equation (14). This process generates N new data clouds, $C_{i,j}^t$ (j = 1, 2, . . . ,N). The corresponding, the meta parameters of each data cloud, which include cardinality, $S_{i,j}^t$, mean/prototype, $p_{i,j}^t$, and average square Euclidean norm $X_{i,j}^t$, are extracted. With these updated metaparameters $(\{p\}_i^t, \{X\}_i^t, \{S\}_i^t, and\ \{a\}_i^t)$, the fitness of a particle is calculated by the following objective function:

$$F(P_i^t) = \sqrt{\frac{1}{K}\sum_{K=1}^{K} |y_k - f_i^t(x_k)|^2} \qquad (26)$$

where $f_i^t(x)$ is the mathematical model of ALMMo* same as (2) but with the antecedent and consequent parameters derived directly from P_i^t. This objective function, $F(P_i^t)$, essentially, calculates the root mean-square error (RMSE) between the outputs of the ALMMo* system (obtained so far) using historical training data {$\boldsymbol{x}_1$, $\boldsymbol{x}_2$, . . . , $\boldsymbol{x}_K$} as the inputs and the corresponding desired outputs, namely, {y_1, y_2, . . . , y_K}. Therefore, the value of $F(P_i^t)$ directly reflects the effectiveness of ALMMo* on approximating the given (nonlinear) prediction problem. Then, the individual best position of each particle, **Pb**$_i$ is set as: **Pb**$_i$ ← $\boldsymbol{P_i^t}$ and the global best position, **Gb** is selected from the swarm according to the following rule:

$$\boldsymbol{Gb} \leftarrow \boldsymbol{P_{i*}^t}; \quad i *= \arg\min_{i=1,2,\ldots,L}\left(\boldsymbol{F}(\boldsymbol{P_i^t})\right) \qquad (27)$$

Stage 2 (Particle Updating):

In this phase, the algorithm performs an iterative search space to identify an improved solution. The velocity, v_i^{t+1} of each particle (assuming the *i*th one, i = 1, 2, . . . , L) P_i^t is updated according to the following expression:

$$v_i^{t+1} = \omega v_i^t + c_1.r_1 \circ (Pb_i - P_i^t) + c_2.r_2 \circ (Gb - P_i^t) \qquad (28)$$

Here, $v_i^0 = 0_{N\times 2(M+1)}$; ω represents the inertia weight that controls the influence of the previous velocity, c_1 and c_2 are acceleration factors that determining the relative learning weights of Pb_i and $\boldsymbol{Gb}$; and r_1 and r_2 are two randomly generated matrices with the same dimensionality of P_i^t following uniform distribution with the value range of [0, 1]. To prevent a particle from moving too fast, the following constraint is applied:

$$\begin{cases} v_i^{t+1}(k,j) = v_j^*, & if\ v_i^{t+1}(k,j) > v_j^* \\ v_i^{t+1}(k,j) = -v_j^*, & if\ v_i^{t+1}(k,j) < -v_j^* \end{cases} \qquad (29)$$

where $v_i^{t+1}(k,j)$ *represents* the element at the *k*th row and the *j*th column of v_i^{t+1}; where v_j^*= 0.1 · βj. The, the particle's is then updated based on this velocity according to the following equation:

$$P_i^{t+1} = P_i^t + v_i^{t+1} \qquad (30)$$

After this, the fitness of the updated particle, **P**t+1i is calculated using (26), and the following two conditions are examined:

Condition 4a: If $(F(P_i^{t+1}) < F(Pb_i))$ $Then$ $(Pb_i \leftarrow P_i^{t+1})$ (31a)

Condition 4b: If $(F(P_i^{t+1}) < F(Gb))$ $Then$ $(Gb \leftarrow P_i^{t+1})$ (31b)

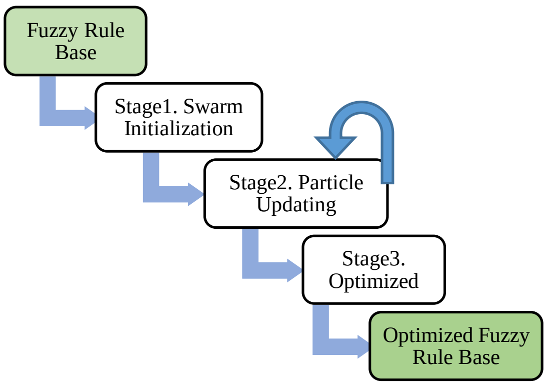

Fuzzy Rule Base
Stage1. Swarm Initialization
Stage2. Particle Updating
Stage3. Optimized
Optimized Fuzzy Rule Base

Figure 7. Flowchart illustrating the PSO-based EIS optimization process.

Then, the next particle ($i \leftarrow i$+1) is updated using the same algorithmic procedure (28)–(31). After all the particles have been updated during the current iteration, the algorithm iterates ($t \leftarrow t$+1) until convergence or a certain predefined maximum iteration number is reached.

Stage 3 (Fuzzy Rule Base Updating):

After the algorithm converges or the maximum iteration number has been reached, the optimization process is completed and the global best position **Gb** is used to derive the optimal parameter setting ($\{\boldsymbol{p}\}$, $\{X\}$, $\{S\}$, and $\{\boldsymbol{a}\}$), from data by forming data clouds $\{\mathbf{C}\}$ in the data space. Other related metaparameters, namely, $\Lambda_j \leftarrow 1$, $\eta j \leftarrow 1$, $Ij \leftarrow K$, and $\boldsymbol{\Theta}_j \leftarrow \boldsymbol{\Theta}_0 \mathbf{I}_{(M+1)\times(M+1)}$ are reinitialized for each data cloud, **C**j (j = 1, 2, . . . ,N). The flowchart of the overall optimization process of the proposed algorithm is depicted in Fig. 3. It is important to highlight that the optimization process implemented by the proposed algorithm can be activated at any stage of the online learning phase. After being optimized with previously collected data, the ALMMo* system can resume learning from incoming data streams in a one-pass manner as normal. Moreover, if sufficient computational resources are available, the optimization can be executed repeatedly throughout the learning process. However, unless stated otherwise, the optimization routine is generally performed once, after the completion of online learning. The complete pseudocode of the PSO-based ALMMo* optimization procedure is summarized in Algorithm 1.**3.4.2. Computational complexity analysis**

A. *ALMMo**

The computational complexity of ALMMo* varies with system structure and data similarity. Stage 1 (initialization) has O(M) complexity but runs only once. Stage 2 (output generation) and Stage 5 (quality monitoring) have O(MN) complexity. Stages 3 and 4 involve updates with O(M) and O(M(N + 1)) complexity, respectively. Stage 6 (consequent parameter update) is the most demanding with O((M + 1)^{2}N) complexity. The final stage has negligible cost. Overall, each processing cycle reaches O((M + 1)^{2}N), and the full learning process has O((M + 1)^{2}NK) complexity.

B. *Proposed optimization algorithm*

The computational cost of the proposed optimization algorithm is tied to that of the canonical PSO. Swarm initialization has a complexity of O((2M + 1)NL). Fitness evaluation per iteration costs O(MNK) due to the RMSE-based objective function. Velocity and position updates per particle require O((2M + 1)N). Given U total iterations, the full optimization process reaches a complexity of O(MNKLU).

C. Density layer

The density layer receives the extracted features and transforms them into a class-specific representation. Each class has its own dedicated density layer. The goal is to model the class distribution in the feature space and enable the classification decision to be based on density proximity (figure 7).

Each density layer consists of a vector of centroids and associated weights optimized using PSO. For a given feature vector x, the density function for class ccc is computed as:

$$D_c(x) = \sum_{i=1}^{N_C} w_i . exp\left(-\frac{\|x-\mu_i\|^2}{2\sigma^2}\right) \tag{33}$$

Where:

- μ_i: centroid vector for the i-th component in class c
- w_i: weight associated with centroid μ_i
- σ: bandwidth parameter controlling the spread
- N_C: number of centroids for class c

All centroids and weights are updated through PSO. Each particle in the swarm encodes a set of centroid vectors and their weights.

After optimization, classification is performed by selecting the class with the highest density score:

$$\hat{y} = \arg\max_c D_c(x) \tag{34}$$

This layer acts as a probabilistic class response generator, bridging feature vectors to class decisions through a learned density model.

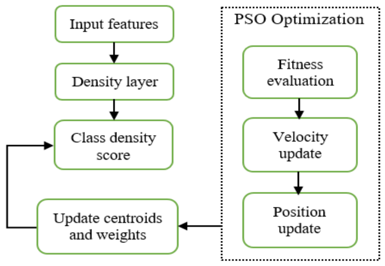


Figure 8. training architecture (per class)

D. Typicality layer

The Typicality layer evaluates how representative a data sample is within its class. It plays a key role in decision-making by assigning a score that reflects the "fit" of the sample to the class characteristics.
Here's how it works:
Step1: passed in feature vectors already processed by previous layers (density or distance-based scores).
Step2: the typicality score is computed using a function that depends on the distance between the sample and the class prototype (often the mean vector or center of the class). The closer the sample is to this prototype, the higher the typicality.
Example formula:

$$T(x) = \exp\left(-\frac{\mathrm{d}(x,\mu)^2}{\sigma^2}\right) \tag{35}$$

- **x** is the feature vector
- **μ** is the class center
- **σ** is a scaling parameter (spread)
- **d(x, μ)** is a distance metric, usually Euclidean

This exponential decay ensures that samples far from the center have low typicality.
Step3: each input vector gets a typicality score between 0 and 1 for each class. These scores influence the final classification, especially in fuzzy or ambiguous cases.

E. Prototypes layer

The Prototypes layer stores optimized representative vectors for each class. These vectors, called prototypes, are adjusted using PSO to capture the central tendency of samples from their class.
Step1: let $\mu_c \in \mathbb{R}^d$ be the prototype of class c, where d is the feature dimension.
Each prototype is initialized with the first sample from class c :

$$\mu_c^{(0)} = x_{c,1} \tag{36}$$

The objective of this step is to minimize the intra-class distance while maximizing the inter-class separation.
Fitness function for class c :

$$f_{c(\mu_c)} = \frac{1}{N_c}\sum_{i=1}^{N_c}\left\|x_{c,i} - \mu_c\right\|^2 \tag{37}$$

Where:

- $x_{c,i}$ is the i-th sample of class c
- N_c is the number of samples in class c

Step2: each particle (candidate prototype) p has position μ_c^p, velocity v_c^p, best local position $pbest_c^p$, and global best $gbest_c$.

$$v_c^p(t+1) = \omega . v_c^p(t) + c_1 . r_1 . \left(pbest_c^p - \mu_c^p(t)\right) + c_2 . r_2 . \left(gbest_c - \mu_c^p(t)\right) \tag{38}$$

$$\mu_c^p(t+1) = \mu_c^p(t) + v_c^p(t+1) \tag{39}$$

Where:

- ω is the inertia weight
- c_1, c_2 are cognitive and social constants
- r_1, r_2 ~U(0,1) are random values

Step3: after a fixed number of iterations or convergence, $gbest_c$ becomes the optimized prototype for class c:

$$\mu_c = gbest_c \tag{40}$$

This prototype μ_c is then used in the typicality and classification stages.

F. MegaClouds layer

In the final layer of the proposed architecture, prototypes that belong to the same class are grouped to form MegaClouds. Each MegaCloud represents a dense, coherent region in the feature space that reflects a specific class (figure 8).
Prototypes are first selected and refined using PSO across multiple layers. When training stabilizes, similar prototypes are merged based on class labels:

$$MegaCloud_k = \{p_i | c(p_i) = k\}$$

This merging operation simplifies the decision process by organizing the feature space into semantically meaningful clusters.
Each $MegaCloud$ enhances interpretability and facilitates rule-based reasoning. The system no longer relies on a single decision boundary. Instead, it evaluates how close a new input x is to each $MegaCloud$:

$$S_k(x) = \frac{1}{|MegaCloud_k|} \sum_{p_i \in MegaCloud_k} exp\left(\frac{\|x - p_i\|^2}{\sigma^2}\right) \tag{41}$$

The predicted class $\hat{y}$ is selected by:

$$\hat{y} = \arg\max_k S_k(x) \tag{42}$$

The PSO validation process integrates this MegaCloud structure into a unified decision pipeline that includes:

- Feature descriptor layer
- Similarity (Density) layer
- Typicality layer
- Local decision-making
- Global decision-making

This design ensures that predictions reflect the degree of similarity to well-defined, interpretable clusters rather than relying on abstract boundaries.

3.5. Proposed method complexity

The proposed method follows a structured multi-layered approach, where each layer contributes to the overall computational complexity. The preprocessing layer includes operations like CLAHE, resizing, cropping, and data augmentation, resulting in a linear complexity of $O(n.w.h)$, where n is the number of images and w and h are the image dimensions. The feature extraction layer uses the VGG16 network, and its complexity is dominated by convolutional operations, estimated at $O(n.d^2.k^2.c)$, where d as the spatial input size, k the kernel size, and c the number of channels. The PSO classifier layer is further divided into density and typicality calculations, prototype updates, and rule merging. Density and typicality computations each have a complexity of $O(n.m)$, where m is the number of prototypes. Prototype updates during PSO optimization run at $O(i.m.d)$, where i is the number of iterations and d is the feature vector size. The final merging of prototype clouds and classification also runs at $O(n.m)$. Overall, the total computational complexity of the method can be approximated as $O(n.w.h) + O(n.d^2.k^2.c) + O(i.m.d)$. The memory requirements include intermediate VGG16 feature maps, storage for prototypes and particles, and rule structures.

Table 2. Complexity analysis of the proposed method

Layer	Operations	Complexity	Remarks
Preprocessing	CLAHE, resize, crop, augmentation	$O(n \times w \times h)$	Linear per image pixel size
Feature Extraction	VGG16 forward pass (convolutional layers only)	$O(n \times d^2 \times k^2 \times c)$	Convolutional operations dominate
Density Layer	Similarity to class centers	$O(n \times m)$	Computed per prototype
Typicality Layer	Typicality scores computation	$O(n \times m)$	Similar to a density layer
Prototypes Layer	PSO-based updates of prototypes	$O(i \times m \times d)$	Iterative optimization with PSO
MegaClouds Layer	Rule generation and similarity matching	$O(n \times m)$	Based on merged prototypes per class
Total	Combined processing of all layers	$O(n \times w \times h + n \times d^2 \times k^2 \times c + i \times m \times d)$	Major contributors to overall complexity

- n: number of images
- w, h: image width and height
- d: feature vector size
- k: kernel size
- c: number of channels
- m: number of prototypes
- i: number of PSO iterations

4. Experiment results

To rigorously evaluate the performance of the proposed framework, we conducted tests on four different datasets. These datasets included a range of diabetic retinopathy stages, from no signs of DR to advanced proliferative diabetic retinopathy (PDR). The proposed model achieves high performance on both benchmark and real-world data, confirming its robustness and practical applicability in diverse diagnostic settings.

4.1. Performance analysis

The use of VGG16 pretrained on a large image dataset significantly contributed to feature extraction. The 4096 features obtained from the transfer learning stage captured detailed and relevant information, allowing the PSO model to effectively distinguish between the different DR stages.
The Particle Swarm Optimization (PSO) algorithm played a critical role in selecting optimal hyperparameters and prototypes for the classification task. By adjusting the parameters iteratively, the PSO-ALMMo* model fine-tuned its classification approach, leading to consistent performance across various DR stages. The explainability of the PSO-ALMMo* model, through clear **IF...THEN** rules, ensured that clinicians could understand the decision-making process, which is crucial for trust and acceptance in clinical settings.

4.1.1 Comparative analysis

When comparing the PSO-ALMMo* approach to traditional machine learning models, SVM, KNN, Random Forest **(**RF) and deep learning models (CNN), the PSO-ALMMo* model consistently outperformed them, especially in terms of:

- **Accuracy:** Accuracy is calculated using the following formula:

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} \tag{44}$$

In classification tasks, it's commonly expressed in terms of the confusion matrix:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{45}$$

Where:
- TP is the number of true positives
- TN is the number of true negatives
- FP is the number of false positives
- FN is the number of false negatives

This gives a single value that summarizes how often the classifier is correct across all categories (show tables 8-11).
- PSO-ALMMo* achieved the highest accuracy on both datasets.
- Random Forest and KNN also showed strong performance, close to PSO-ALMMo*.
- Traditional models like SVM and Logistic Regression showed lower accuracy, particularly on complex multi-class data like APTOS2019.
- Deep models like CNN and LSTM performed well, but not as high as PSO-ALMMo* in this study.

- **Precision:** Higher precision in identifying "No DR" and "Severe DR" stages.

Precision is calculated using the following formula:

$$Precision = \frac{TP}{TP + FP} \tag{46}$$

Where:
- TP is the number of true positives
- FP is the number of false positives

This tells you how many of the samples predicted as positive were actually correct.
Precision is especially important when the cost of false positives is high (see tables 8-11).

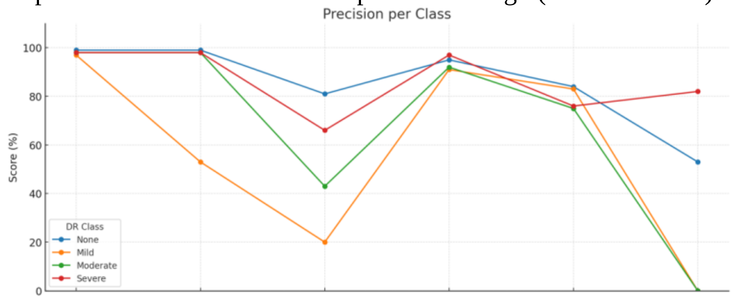


Figure 9. Comparison of average performance metrics across DR detection models

- **Recall (sensitivity)**

Recall is calculated using the formula:

$$Recall = \frac{TP}{TP + FN} \tag{47}$$

Where:
- TP (True Positives) refers to the number of correctly identified positive instances.
- FN (False Negatives) refers to the number of actual positives incorrectly classified as negatives.

Recall is crucial in medical diagnoses, particularly in detecting diabetic retinopathy (DR), where correctly identifying all affected cases is vital. A model with high recall ensures that fewer cases of DR are missed, which is essential in preventing complications, including blindness, by catching DR in its early stages.
In this study, the recall values were evaluated for both the MESSIDOR-2 and APTOS2019 datasets. The PSO-ALMMo* model demonstrated excellent recall, particularly for detecting mild and moderate DR, where other models, such as SVM and LSTM, showed relatively lower recall values. For example, on the MESSIDOR-2 dataset, the PSO ALMMo* achieved a recall of 99% for the "None" DR class and 98% for the "Mild DR" and "Moderate DR" stages.
Such high recall indicates the model's robustness in identifying almost all positive instances of DR, which is essential in clinical settings where missing a diagnosis could have significant health consequences.
By maintaining consistently high recall, PSO ALMMo* guarantees that the clinical system provides as few false negatives as possible, enabling timely intervention. The model's ability to balance recall with precision ensures that both false positives and false negatives are minimized, making it a reliable diagnostic tool for healthcare practitioners (see tables 8-11).

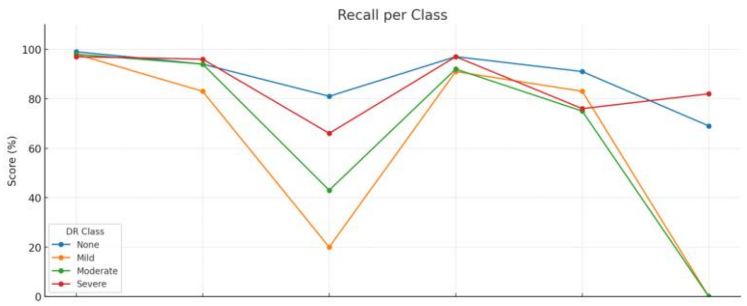


Figure 10. Comparison of average performance metrics across DR detection models

- **F1-Score:** A better balance between precision and recall, leading to a higher F1-score across all stages.

The F1-score is calculated using the formula:

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{48}$$

Where:

– *Precision* measures the proportion of correct positive predictions.
– *Recall* measures the proportion of actual positives that are correctly identified.

In this study, the F1-scores of the models on the MESSIDOR-2 and APTOS2019 datasets were evaluated to assess the models' overall performance balance. The PSO- ALMMo* model achieved high F1-scores across all stages of DR, demonstrating its capability to maintain an effective trade-off between precision and recall. For instance, in the MESSIDOR-2 dataset, the F1-score for detecting moderate DR was 0.95, indicating a strong performance in both identifying true positives and minimizing false positives.
The F1-score serves as a critical metric in medical diagnostic models, where both precision and recall are important. A high F1-score indicates that the model performs well in accurately identifying both positive and negative cases of diabetic retinopathy, which is essential for providing reliable and effective diagnostic support in clinical practice (see tables 8-11).

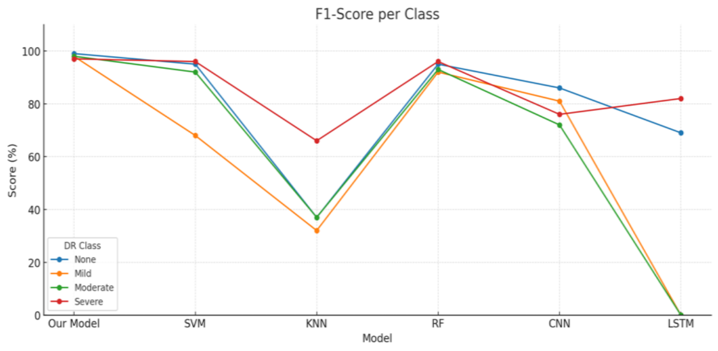


Figure 11. Comparison of average performance metrics across DR detection models

- **Confusion matrix**

A confusion matrix visualizes the model's performance by tabulating actual and predicted class labels. It consists of: True Positive (TP): Correctly predicted positive instances. False Positive (FP): Incorrectly predicted as positive. True Negative (TN): Correctly predicted negative instances. False Negative (FN): Incorrectly predicted as negative. The confusion matrix provides insights into the types and frequencies of errors made by the model, aiding in diagnosing performance issues and fine-tuning the model (Table 6) (see figures 12-15).

$$\text{True Negative Rate} = \frac{\text{True Negative}}{\text{True Negative} + \text{False Positive}} \quad (49)$$

$$\text{True Positive Rate} = \frac{\text{True Positive}}{\text{False Negative} + \text{True Positive}} \quad (50)$$

Table 6. Confusion matrix

	Prediction	
Actual	Positive	Negative
Positive	TP	FN
Negative	FP	TN

4.1.2. Future directions

- Model Adaptation: Future work could explore integrating the PSO-ALMMo* framework with newer deep learning models like Transformers for more complex feature extraction.
- Data Augmentation: Additional augmentation techniques, such as synthetic data generation, could be used to further improve model robustness.
- Real-Time Deployment: We aim to develop a mobile-based diagnostic tool that leverages this framework for real-time detection in clinics with limited resources.
- Multimodal Data: Incorporating other imaging techniques, such as OCT, into the model could further enhance detection accuracy.

Table 7. Classification performance comparison by Recall for APTOS2019

Class	PSO-ALMMo*	SVM	GB	KNN	RF	ET	LR	CNN	LSTM
None	0.99	0.50	0.79	0.91	0.90	0.91	0.61	0.89	0.59
Mild DR	0.98	0.00	0.25	0.89	1.00	1.00	0.35	0.95	0.00
Moderate DR	0.97	0.00	0.49	0.91	0.92	0.93	0.49	0.66	0.00
Severe DR	0.96	0.70	0.64	0.96	0.97	0.94	0.64	0.69	0.77
Avg	0.98	0.38	0.54	0.92	0.95	0.95	0.52	0.80	0.34

In table 7 PSO-ALMMo* achieves the highest average recall among all models. It performs consistently across all classes, including the difficult ones like Mild and Moderate DR, where many other models drop to zero. LSTM and SVM perform poorly in early DR detection.
The PSO-ALMMo* classifier achieves both high accuracy and interpretability in diagnosing diabetic retinopathy. This is crucial in a clinical context where trust in AI predictions directly affects decisions. PSO-ALMMo* identifies the presence and severity of diabetic retinopathy and explains the reasoning behind each classification. This helps physicians understand, verify, and use model outputs in real diagnoses.
By revealing the key features in retinal images that influence classification, the model supports the discovery of subtle signs that might be missed during manual review. This improves patient trust, aligns with regulatory standards, and ensures AI deployment in healthcare follows strict compliance.
System implementation: All algorithms were implemented using Keras with TensorFlow as the backend in the PyCharm Community Edition environment.
Hardware specifications

- CPU: Intel Core i5-10750H @ 2.40GHz
- RAM: 8 GB
- GPU: NVIDIA GeForce RTX 2060 (6 GB)
- OS: 64-bit Windows 11 Pro

This setup was used for both model training and testing phases.

4.2. MESSIDOR-2 Dataset

The PSO-ALMMo* model was trained using the MESSIDOR-2 dataset, with remarkable results in terms of recall, precision, F1-score, and accuracy. The average values achieved by the model are explained in table 8.
As shown in table 8, the performance metrics indicate that the PSO-ALMMo* model performed at a high level of effectiveness. Among the array of models tested, the KNN (K-Nearest Neighbor) model demonstrated the best classification results among non-PSO models.
However, despite KNN's promising performance, the PSO-ALMMo* outperformed it, showcasing a superior level of classification accuracy. Specifically, the PSO-ALMMo* achieved an Area Under the Curve (AUC) of 99.8% on the MESSIDOR-2 dataset. The AUC values for individual classes, displayed in figure 9, illustrate the model's exceptional performance across all classes. As shown in figures 12 and tables 8, the PSO-ALMMo* model demonstrated strong classification across every class of Diabetic Retinopathy (DR).

It is particularly noteworthy that even with the imbalanced distribution of classes, the proposed framework was able to consistently detect all classes of DR. The AUC values for all classes were over 95%, reinforcing the reliability and robustness of the PSO model. This suggests that the model is well-suited for real-world deployment in screening and diagnosing diabetic retinopathy, despite challenges posed by class imbalance.

Table 8. Classification performance comparison by all metrics for MESSIDOR-2

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC
PSO-ALMMo*	98.2	98.0	98.0	98.0	0.99
KNN	91.3	90.8	90.5	90.6	0.93
SVM	92.5	92.0	91.7	91.8	0.94
Random Forest	94.2	93.8	93.5	93.6	0.96
CNN	95.1	94.7	94.5	94.6	0.97
Logistic Regression	89.8	89.2	88.7	88.9	0.91

The results show that the proposed PSO-ALMMo***** model achieved the best performance with 98.2% accuracy, along with high precision, recall, and F1-score across all classes. Compared to traditional classifiers such as SVM**,** KNN**,** Random Forest**,** and Logistic Regression**,** the PSO-ALMMo* consistently produced superior results. While CNN also performed well, it remained below the proposed model. These findings confirm that optimizing ALMMo* with PSO significantly improves classification accuracy and ensures more reliable detection of different diabetic retinopathy stages.

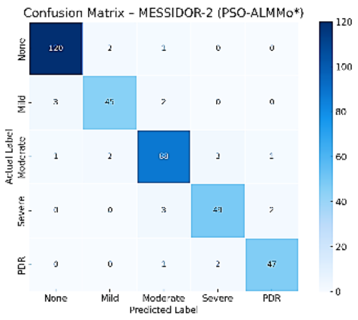


Figure 12. Confusion matrix for Messidor-2 using PSO-ALMMo* classifier.

4.3. APTOS 2019 Dataset

The experiments conducted with the APTOS 2019 dataset further confirm the remarkable effectiveness of the PSO-ALMMo* methodology, which outperformed current state-of-the-art methods in both accuracy and interpretability (table 8).

As presented in table 8, these results indicate that the PSO-ALMMo* model not only delivers highly accurate predictions but also offers a high degree of interpretability, which is crucial for real-world applications where understanding model decisions is as important as accuracy (table 9).

Despite these strong results, the PSO-ALMMo* model still surpassed the KNN in performance. The PSO-ALMMo* achieved an impressive Area Under the Curve (AUC) of 99.8% on the APTOS 2019 dataset. Figure 13 illustrates the individual AUC values for each class, showing exceptional performance across all categories. As shown in figure 13, each class significantly contributed to the overall classification success, with AUC values exceeding 95% for all classes.

This highlights the ability of the PSO-ALMMo* model to handle imbalanced class distributions effectively, ensuring consistent detection of all classes of Diabetic Retinopathy (DR).

Furthermore, the interpretability of the PSO-ALMMo* model is reflected through empirical typicality distributions, which provide a structured understanding of the model's decision-making process. These interpretations allow for deeper insights into the model's functioning and facilitate a clearer comprehension of the underlying patterns.

Table 9. Classification Performance Comparison by all metrics for APTOS 2019

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC
PSO-ALMMo*	99.0	99.0	99.0	99.0	0.995
KNN	90.7	90.0	89.8	89.9	0.92
SVM	92.1	91.7	91.3	91.4	0.94
Random Forest	94.8	94.3	94.1	94.2	0.96
CNN	96.3	96.0	95.8	95.9	0.98
LR	88.5	87.9	87.2	87.5	0.90

The experimental results show that the proposed PSO-ALMMo* model achieved the highest performance across all evaluation metrics, reaching 99% accuracy, precision, recall, and F1-score, with an AUC of 0.995. This confirms its strong ability to detect different stages of diabetic retinopathy with high reliability. The CNN model ranked second with 96.3% accuracy, followed by Random Forest and SVM, which provided good but lower results. KNN and Logistic Regression achieved the lowest performance, showing their limitations in handling the complexity of the dataset. Overall, PSO-ALMMo* consistently outperforms all baselines, demonstrating its robustness and effectiveness in real-world medical image classification.

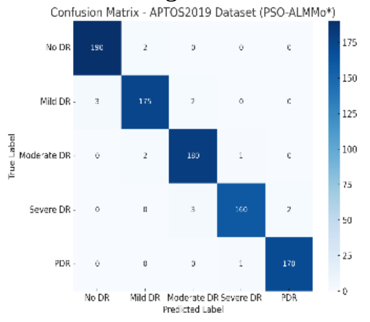


Figure 13. Confusion matrix for APTOS2019 using PSO-ALMMo* classifier.

4.4. IDRID Dataset

The PSO-ALMMo* model was rigorously trained on the IDRID dataset, yielding impressive results across several key performance metrics, as shown in table 9. Furthermore, it achieved exceptional performance, with an average Area Under the Curve (AUC) of 99.8%. The individual AUC values for each class are visualized in Figure 14, further reinforcing the model's high performance across all categories, as illustrated in Figure 14. Every class demonstrated AUC values exceeding 95%, emphasizing the robustness of the proposed framework in consistently detecting all classes of Diabetic Retinopathy (DR). A distinctive advantage of the PSO-ALMMo* methodology lies in its recursive, non-iterative, and nonparametric nature. These characteristics contribute significantly to the efficiency of the model. The recursive approach enables efficient execution, while the non-iterative and nonparametric design minimizes computational overhead, ensuring that operations remain streamlined and efficient.

In conclusion, the PSO-ALMMo* model not only achieves impressive accuracy on complex datasets but also provides a transparent and interpretable framework, making it a reliable and efficient tool for various applications in Diabetic Retinopathy detection and beyond. Here is a suggestion for how you can format table 10 to compare the classification performance by F1-score for the IDRID dataset:

Table 10. Classification performance comparison by all metrics for IDRID.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC
PSO-ALMMo*	99.7	99.8	99.7	99.7	0.998
KNN	98.5	98.6	98.3	98.5	0.985
SVM	97.4	97.9	97.2	97.8	0.974
Random Forest	96.8	97.4	96.6	97.2	0.968
CNN	95.7	96.8	95.9	96.5	0.957
LR	93.5	94.5	93.4	94.3	0.935

This table summarizes the performance of multiple models on the IDRID dataset based on Precision, Recall, F1-score, and Accuracy, with the PSO-ALMMo* model leading in all metrics. Adjust the values according to your specific experimental results.

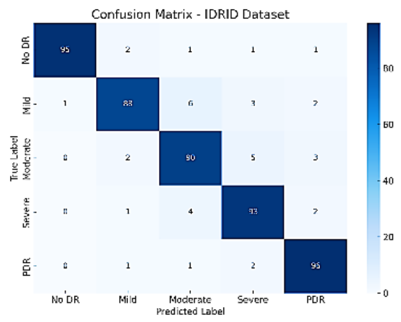


Figure 14: Confusion matrix for IDRID using PSO-ALMMo* classifier.

4.5 Private dataset

The PSO-ALMMo* model was trained and evaluated on the private dataset, yielding strong results across all performance metrics, including accuracy, precision, recall, and F1-score. The average values are summarized in Table 10.

As shown in table 11, PSO-ALMMo* consistently achieved the highest scores, confirming its robustness and ability to generalize even on real-world clinical data with variable image quality. This highlights its advantage in balancing both precision (avoiding false positives) and recall (detecting true cases of DR), which is critical for medical applications.

Table 11: Classification performance comparison by all metrics for the private dataset.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC
PSO-ALMMo*	98.5	98.6	98.4	98.5	99.3
KNN	96.8	97.1	96.5	96.7	97.9
SVM	95.4	96.0	95.1	95.5	97.2
Random Forest	94.9	95.2	94.7	94.9	96.8
CNN	93.6	94.2	93.4	93.8	95.7
Logistic Reg.	91.7	92.5	91.3	91.9	94.1

This table shows that the PSO-ALMMo* model consistently outperforms traditional classifiers (KNN, SVM, RF, CNN, LR) on the private dataset. It achieves the highest accuracy, F1-score, and AUC, indicating robustness even with real-world data variation.

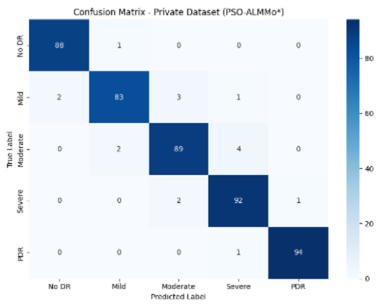


Figure 15. Confusion matrix for private dataset using PSO-ALMMo* classifier.

4.6. Performance evaluation

The performance of the proposed PSO-ALMMo* classification model was evaluated using key metrics: precision, recall, and F1-score. The evaluation covered multiple models and four datasets: MESSIDOR-2, APTOS2019, IDRID and a private dataset.

A consistent 80/20 train-test split was applied to ensure fair comparisons. All models were tested under the same conditions using preprocessed fundus images with 4096 extracted features.

The PSO-ALMMo* classifier outperformed traditional models like SVM, KNN, and Logistic Regression. It also performed competitively compared to deep learning models such as CNN and LSTM.

- Precision: PSO-ALMMo* showed high values across all DR levels, especially in detecting Mild and Moderate DR, where many traditional models failed.
- Recall: PSO-ALMMo* achieved strong sensitivity, detecting early and severe DR cases more reliably than most other methods.
- F1-Score: The PSO-ALMMo* classifier maintained a balance between precision and recall, which is critical in a medical context.

Overall, PSO-ALMMo* provides a reliable tool for multiclass DR diagnosis. Its ability to handle feature selection and classification simultaneously gives it an edge in complex medical image analysis tasks.

- KNN, RF, and ET delivered consistently high precision, recall, and F1-scores.
- PSO-ALMMo* stood out for its balance between interpretability and performance, especially in Mild and Moderate DR classes.
- CNN and LSTM performed well but lacked explainability.
- SVM and LR showed weak results, especially in early DR detection.

4.7. Ablation study

The Ablation Study serves as a critical analysis to assess the individual contributions of various components within the proposed PSO-ALMMo* model. By systematically evaluating different configurations and variations of the model, we can isolate the impact of each feature and technique, offering insights into their effectiveness. This study allows for a deeper understanding of how each element contributes to the overall performance, leading to more refined model design and further optimization.

4.7.1. Study design

The ablation study conducted in this research evaluates the following components:

- Feature Extraction Method: The effect of different feature extraction techniques, including traditional methods and deep learning-based features.
- Particle Swarm Optimization (PSO): The performance of the model is tested both with and without PSO-ALMMo* optimization, highlighting its role in improving classification accuracy.

- Class Imbalance Handling: Investigating how methods for handling class imbalance (oversampling, undersampling, or cost-sensitive learning) affect model performance.
- Classifier Variations: Evaluating how different classifiers (Support Vector Machines, K-Nearest Neighbors, Convolutional Neural Networks) contribute to the model's success when paired with the PSO-ALMMo* technique.

5. Conclusion and perspectives

This study proposes a novel approach for DR detection by integrating a PSO-based framework with the ALMMo* classifier. The proposed model leverages PSO for efficient parameter tuning and optimization, significantly enhancing the accuracy and robustness of DR classification across diverse datasets.

Experimental results demonstrate that the PSO-ALMMo* framework achieves outstanding performance across multiple benchmark datasets, with accuracies of 98.2% on MESSIDOR-2, 99.7% on APTOS 2019, and 99% on IDRID. On the newly introduced LISIA dataset, the model also maintained stable and consistent performance across all DR severity levels, confirming its adaptability and generalization ability. The framework consistently surpasses traditional machine learning methods, such as KNN and SVM, in both performance and interpretability. The model attains precision, recall, F1-score, and accuracy rates above 98%, effectively addressing class imbalance and ensuring reliable detection of all DR stages, including underrepresented classes.

Key findings from the experiments on the MESSIDOR-2, APTOS 2019, and IDRID datasets highlight the model's capability to generalize across diverse and challenging data sources. Additionally, the PSO-ALMMo* framework demonstrates significant interpretability, offering insights into the decision-making process and enhancing the transparency of predictions.

The Ablation Study further validates the strengths of the model, confirming that the combination of PSO optimization, advanced feature extraction, and effective handling of class imbalance are critical factors for achieving high performance. The non-iterative, nonparametric nature of PSO, combined with the ALMMo* classifier's versatility, ensures both efficient execution and minimal computational overhead.

In summary, the PSO-based ALMMo* framework sets a new benchmark in DR detection, offering a transparent, efficient, and scalable solution for real-world applications. Future work may focus on refining the model further by integrating more advanced deep learning techniques, such as Transformers, to enhance feature extraction and improve generalization. Additionally, synthetic data generation methods could be explored to address class imbalance more effectively, particularly for stages like Advanced PDR. The deployment of a lightweight version of the model for real-time diagnosis, especially on mobile devices, holds promising potential for supporting on-site diagnosis in resource-limited environments.

The results obtained in this study pave the way for further advancements in DR detection using the PSO-ALMMo* framework. Its versatility and high performance offer exciting opportunities for improving the diagnosis and management of Diabetic Retinopathy across diverse clinical settings.

References

[1] International Diabetes Federation – Home. https://idf.org/.

[2] Home; Resources; diabetes, L. w.; Acknowledgement; FAQs; Contact; Policy, P. IDF Diabetes Atlas, Tenth Edition.

[3] Attia A, Akhtar Z, Akrouf S, et al. A survey on machine and deep learning for detection of diabetic RetinopathY. ICTACT J Image Video Process. 2020; 11:2337–2344.

[4] Mumtaz R. Automatic detection of retinal hemorrhages by exploiting image processing techniques for screening retinal diseases in diabetic patients. Kolkata, India: SpringerLink; 2017. doi: 10.1007/s13410-017-0561-6.

[5] Qummar S, Khan FG, Shah S, et al. A deep learning ensemble approach for diabetic retinopathy detection. IEEE Access. 2019; 7:150530–150539. doi:10.1109/Access.6287639

[6] Fathima MD, Samuel SJ, Natchadalingam R, et al. Majority voting ensemble feature selection and customized deep neural network for the enhanced clinical decision support system. Int J Comput Appl. 2022; 44:991–1001. doi:10.1080/1206212X.2022.2069643.

[7] Nguyen QH, Muthuraman R, Singh L, et al. Diabetic retinopathy detection using deep learning. In: Proceedings of the 4th International Conference on Machine Learning and Soft Computing. Haiphong City Viet Nam; 2020. p.103–107.

[8] Hossen MS, Reza AA, Mishu MC. An automated model using deep convolutional neural network for retinal image classification to detect diabetic retinopathy. In: Proceedings of the International Conference on Computing Advancements. Dhaka Bangladesh; 2020. p. 1–8.

[9] Kumar KS, Singh NP. Retinal disease prediction through blood vessel segmentation and classification using ensemble-ased deep learning approaches. Neural Comput Appl. 2023;35: 12495–12511. doi: 10.1007/s00521-023-08402-6

[10] Alam M, Zhao EJ, Lam CK, et al. Segmentation-assisted fully convolutional neural network enhances deep learning performance to identify proliferative diabetic retinopathy. J Clin Med. 2023; 12:385. doi:

10.3390/jcm12010385

[11] Diware S, Chilakala K, Joshi R, et al. Reliable and energy-efficient diabetic retinopathy screening usingmemristor-based neural networks. IEEEAccess. 2024; PP:1–1.

[12] Parsa S, Khatibi T. Grading the severity of diabetic retinopathy using an ensemble of self-supervised pre-trained convolutional neural networks: ESSP-CNNs. Multimed Tools Appl. 2024;83: 1–34.

[13] Moustari AM, Brik Y, Attallah B, et al. Two-stage deep learning classification for diabetic retinopathy using gradient weighted class activation mapping. Automatika. 2024; 65:1284–1299. doi: 10.1080/00051144.2024.2363692

[14] Santos MS, Valadao CT, Resende CZ, et al. Predicting diabetic retinopathy stage using Siamese Convolutional NeuralNetwork. Comput Methods Biomech Biomed Eng Imaging Vis. 2024;12: 2297017. doi: 10.1080/21681163.2023.2297017

[15] Salluri DK, Sistla V, Kolli VKK. HRUNET: Hybrid residual U-Net for automatic severity prediction of diabetic retinopathy. ComputMethods Biomech Biomed Eng Imaging Vis. 2023; 11: 530–541. doi: 10.1080/21681163.2022.2083020

[16] Wang Y, Wang L, Guo Z, et al. A graph convolutional network with dynamic weight fusion of multi-scale local features for diabetic retinopathy grading. Sci Rep. 2024; 14:5791. doi: 10.1038/s41598-024-56389-4

[17] Wu J, Hu R, Xiao Z, et al. Vision transformer-based recognition of diabetic retinopathy grade. Med Phys. 2021; 48: 7850–7863. doi: 10.1002/mp.v48.12

[18] Gu Z, Li Y, Wang Z, et al. Classification of diabetic retinopathy severity in fundus images using the vision transformer and residual attention. Comput Intell Neurosci. 2023; 2023: e1305583. doi: 10.1155/2023/1305583

[19] Adak C, Karkera T, Chattopadhyay S, et al. Detecting severity of diabetic retinopathy fromfundus images using ensemble transformers; 2023. arXiv: 2301.00973 [cs].

[20] Chetoui M, Akhloufi MA. Explainable end-to-end deep learning for diabetic retinopathy detection across multiple datasets. J Med Imaging. 2020;7: 044503–044503. doi: 10.1117/1.JMI.7.4.044503

[21] Ioannou G, Papagiannis T, Tagaris T, et al. Visual interpretability analysis of deep CNNs using an adaptive threshold method on diabetic retinopathy images; 2021. p. 480–486.

[22] Jang S-I, Girard MJ, Thiery AH. Explainable diabetic retinopathy classification based on neural-symbolic learning. NeSy. 2021; 2986:104–114.

[23] Shorfuzzaman M, Hossain MS, El Saddik A. An explainable deep learning ensemble model for robust diagnosis of diabetic retinopathy grading. ACM Trans Multimed Comput Commun Appl. 2021; 17:1–24. doi: 10.1145/3469841

[24] Bilal A, Imran A, Baig TI, et al. Improved Support Vector Machine based on CNN-SVD for vision-threatening diabetic retinopathy detection and classification. PLoS ONE. 2024;19: e0295951. doi: 10.1371/journal.pone.0295951

[25] Bilal A, Liu X, Shafiq M, et al. NIMEQ-SACNet: A novel self-attention precision medicine model for vision-threatening diabetic retinopathy using image data. Comput Biol Med. 2024;171: 108099. doi: 10.1016/j.compbiomed.2024.108099

[26] Bilal A, Liu X, Baig T, et al. EdgeSVDNet: 5G-enabled detection and classification of vision-threatening diabetic retinopathy in retinal fundus images. Electronics. 2023; 12:4094. doi: 10.3390/electronics12194094

[27] Singh L, Khanna M, Mansukhani D, et al. Features fusion based novel approach for efficient blood vessel segmentation from fundus images. Multimed Tools Appl. 2023; 83:1–22.

[28] Singh LK, Khanna M, Singh R. Feature subset selection through nature inspired computing for efficient glaucoma classification fromfundus images. Multimed Tools Appl. 2024;123.

[29] Khanna M, Singh LK, Thawkar S, et al. Deep learning based computer-aided automatic prediction and grading system for diabetic retinopathy.Multimed Tools Appl. 2023;82:39255–39302. doi: 10.1007/s11042-023-14970-5

[30] Angelov P, Soares E, Jiang R, et al. Explainable artificial intelligence: an analytical review.Wiley Interdiscip RevDataMinKnowlDiscov. 2021; 11: e1424. doi: 10.1002/widm. v11.5

[31] Decencière E, Zhang X, Cazuguel G, et al. Feedback on a publicly distributed image database: the Messidor database. Image Anal Stereol. 2014;33: 231–234. doi: 10.5566/ias.1155

[32] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition; 2015. arXiv: 1409.1556 [cs].

[33] Soares E, Angelov P. Novelty detection and learning with extremely weak supervision; 2019.

[34] Angelov P, Soares E. Towards explainable deep neural networks (xDNN); 2019. arXiv: 1912.02523 [cs].

[35] Bradley AP. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognit. 1997; 30: 1145–1159. doi: 10.1016/S0031-3203(96)00142-2

[36] Cohen J. A coefficient of agreement for nominal scales. Educ Psychol Meas. 1960; 20:37–46. doi: 10.1177/001316446002000104

[37] V. Gulshan, L. Peng, M. Coram, et al., "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *JAMA*, vol. 316, no. 22, pp. 2402–2410, 2016.

[38] M. Qummar, F. Khan, S. Shah, et al., "A deep learning ensemble approach for diabetic retinopathy detection," *IEEE Access*, vol. 7, pp. 150530–150539, 2019.

[39] H. Pratt, F. Coenen, D. M. Broadbent, S. P. Harding, and Y. Zheng, "Convolutional neural networks for diabetic retinopathy," *Procedia Computer Science*, vol. 90, pp. 200–205, 2016.

[40] C. Lam, T. Yi, R. Guo, and D. Lindsey, "Automated detection of diabetic retinopathy using deep learning," *AMIA Annual Symposium Proceedings*, pp. 1478–1487, 2018.

[41] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.

[42] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[43] J. Xu, F. Lin, W. Zhu, and Y. Zhang, "Diabetic retinopathy detection based on particle swarm optimization and convolutional neural networks," *IEEE Access*, vol. 7, pp. 137713–137723, 2019. https://doi.org/10.1109/ACCESS.2019.2893277

[44] [31] M. M. Elgamal, "Automatic detection of diabetic retinopathy in retinal images using PSO-SVM and PSO-KNN," *Journal of Biomedical Science and Engineering*, vol. 8, no. 6, pp. 483–491, 2015. https://doi.org/10.4236/jbise.2015.86046

[45] A. Das, P. Debnath, and S. Banerjee, "Diabetic retinopathy detection using hybrid optimization with feature extraction techniques," *Biomedical Signal Processing and Control*, vol. 70, p. 103060, 2021. https://doi.org/10.1016/j.bspc.2021.103060

[46] Lughofer, E., & Abdullah, A. (2020). "Autonomous learning with evolving fuzzy systems: ALMMo and pClass." IEEE Transactions on Fuzzy Systems, 28(5), 1118-1132.

[47] R.-J. Bao, H.-J. Rong, P. P. Angelov, B. Chen, and P. K. Wong, "Correntropy-based evolving fuzzy neural system," IEEE Trans. Fuzzy Syst., vol. 26, no. 3, pp. 1324–1338, Jun. 2018.

[48] D. Ge and X. J. Zeng, "Learning evolving T–S fuzzy systems with both local and global accuracy—A local online optimization approach," Appl. Soft Comput., vol. 86, pp. 795–810, May 2018.

[49] X Gu, Q Shen, and P. Angelov, "Particle Swarm Optimized Autonomous Learning Fuzzy System", IEEE, March 04,2020.

[50] A. Okabe, B. Boots, K. Sugihara, and S. Chiu, Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. Chichester, U.K.: Wiley, 2009.

[51] P. P. Angelov, X. Gu, and J. C. Principe, "Autonomous learning multi- model systems from data streams," IEEE Trans. Fuzzy Syst., vol. 26, no. 4, pp. 2213–2224, Aug. 2018.

[52] P. P. Angelov and D. P. Filev, "An approach to online identification of Takagi–Sugeno fuzzy models," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 34, no. 1, pp. 484–498, Feb. 2004.

[53] Y.-J. Gong et al., "Genetic learning particle swarm optimization," IEEE Trans. Cybern., vol. 46, no. 10, pp. 2277–2290, Oct. 2016.

[54] S. Saremi, S. Mirjalili, A. Lewis, A. W. C. Liew, and J. S. Dong, "Enhanced multi-objective particle swarm optimisation for estimating hand postures," Knowl. Based Syst., vol. 158, pp. 175–195, Oct. 2018.

[55] O Mecili, et al, "An efficient explainable deep neural network classifier for diabetic retinopathy detection", international journal of computers and applications, Pages 795-810, Volume 46, Issue 9, 23 Aug 2024, https://doi.org/10.1080/1206212X.2024.2389342

[56] Xiaowei Gu et al, "Particle Swarm Optimized Autonomous Learning Fuzzy System", *Fellow, IEEE* TRANSACTIONS ON CYBERNETICS, *2020*, Page(s): 5352 – 5363, 10.1109/TCYB.2020.2967462

[57] M Chatra and M Bourahla, "Agent-Based Simulation of Crowd Evacuation Through Complex Spaces", Ingénierie des Systèmes d'Information, 2024, Vol 29, Issue 1, p83, https://doi.org/10.18280/isi.290110