EAI Endorsed Transactions
on Energy Web
Research Article EAI.EU

Efficient Diagnosis of Liver Disease using Support Vector Machine Optimized with Crows Search Algorithm

D. Devikanniga[1,*], Arulmurugan Ramu[1] and Anandakumar Haldorai[2]

[1]Assistant Professor, Presidency University, Bengaluru-560064, Karnataka, India.
[2]Associate Professor, Sri Eshwar College of Engineering, Coimbatore-641202, Tamil Nadu, India.

Abstract

The early and accurate prediction of liver disease in patients is still a challenging task among medical practitioners even with latest advanced technologies. The support vector machines are widely used in medical domain. It has proved its efficiency on producing good diagnostic parameters. These results can be further improved by optimizing the hyperparameters of support vector machines. The proposed work is based on optimizing support vector machines with crow search algorithm. This optimized support vector machine classifier (CSA-SVM) is used for accurate diagnosis of Indian liver disease data. The various similar state of art algorithms are taken for comparison with proposed approach to prove its efficient. The performance of CSA-SVM is found to be outstanding among all other approaches in terms of all metrics taken for comparison. It has yielded the classification accuracy of 99.49%.

Keywords: Crow search algorithm, liver disease, sequential minimal optimization, support vector machine.

Received on 27 March 2020, accepted on 25 April 2020, published on 29 April 2020

doi: 10.4108/_____

*Corresponding author. Email: mail4kanniga@gmail.com

1. Introduction

Liver is the largest organ in the body. Its main functions include digestion, remove toxins, fights infection, balance hormones and secrete bile juice. There are various liver diseases which are caused due to virus infection, excess amount drugs, poisoning, alcohol, obesity and many other factors. These causes liver failure which significantly damage the body as it leads to improper body functions. This is a life-threatening condition. Some common liver problems include hepatitis, fatty liver disease, liver cancer etc. There are many tests to diagnose liver dysfunction, liver biopsy, viral hepatitis tests, comprehensive metabolic panel, transient elastography etc. The initial stage of liver diseases are mostly unable to diagnose, as the liver functions normally even with partial infections. This creates a challenging task for doctors for accurate prediction at early stage. Early detection and treatment leads to healing of liver rather than leading to critical conditions.

Many machine learning algorithms such as artificial neural networks, decision trees, support vector machine (SVM) and many others are used in the literature for liver data classification. A few recent works are discussed below and their classification results are tabulated below. The classification algorithms such as Naïve Bayes (NB), J48, Random tree (RT), K-Star are implemented using WEKA tool [1]. In [2], various algorithms namely Logistic regression, SVM, RT, Bagging techniques are compared for classification accuracy. Multi layer feed forward deep neural network (MLFFDNN) trained with back-propagation network (BPN) is used in [3]. XGBoost algorithm is used to predict the liver disease data collected from Andhra Pradesh, India. In this L1 and L2 regularization technique is used to improve efficiency [4]. The class imbalance in ILPD is handled using synthetic minority oversampling technique. Then the classification performance is evaluated for both balanced and unbalanced dataset using K-Nearest neighbor (KNN) and SVM [5]. Particle swarm optimization (PSO) is combined

with SVM for feature selection and applied for classifying liver data [6].

The liver disease diagnosis done using SVM has found to produce good results. This algorithm works still more efficient when it is combined with heuristic and nature-inspired meta-heuristic optimization algorithms (MHOAs). In [7], the modification of kernel and optimal set of SVM hyperparameters that are found using optimization methods such as random search, grid search and the Nelder–Mead method, has improved the classification accuracy of DNA sequence recognition problem. The learning vector quantization neural network algorithm and the Fisher-SVM coupling algorithm are applied for prediction of hypertension risk in steel workers. The efficiency of this combination is proved for varying sample size [8].

MHOAs jointly work with SVM for tasks such as parameter tuning and feature selection. The two key hyper parameters namely penalty parameter and kernel function width are mostly tuned for better efficiency in many works. A few includes, the MHOAs such as Ant colony optimization and PSO [9], Fruitfly optimization algorithm [10], accelerated PSO [11], Multi-verse optimizer approach [12], Simulated annealing [13] were adopted to find optimal set of parameters for SVM. To improve the SVM classification accuracy in high-dimensional datasets, the feature selection technique is applied with the help of MHOAs such as Grasshopper optimization algorithm [14] and Firefly algorithm [15] is used to train all the parameters of SVM. Many such MHOAs are used with SVM for specific applications. But still these algorithms are found to have some limitations. Most of the times the accurate results are not produced. Thus a robust algorithm that promises high diagnostic accuracy in early prediction is needed to solve the above mentioned issues.

In this work, Crow search optimization algorithm (CSA) [19] is firstly combined with SVM linear kernel to optimize its lagrangian values in order to improve the diagnostic efficiency of liver disease dataset. CSA is chosen among other MHOAs as it contains simple and efficient optimization steps. It also maintains good balance between exploration and exploitation. As it has only two tuning parameters, it is simple to apply and fast. It is also noted that it has proved its efficiency in many similar applications. The calculation of alpha and bias value is the critical task during the training of SVM. Many mathematical optimization algorithms like Quadratic programming, Least squares, SMO etc., have been used. Thus in this paper, the usual procedure of optimizing SVM lagrange values using SMO during training is discussed in steps. Then the details of CSA for optimizing these lagrange values in the place of SMO is illustrated. It is observed that the optimization steps of CSA-SVM is very simple and efficient.

The organization of the paper is as follows, Section 1 gives the introduction, Section 2 provides the details of Indian liver disease dataset, concept of SVM, training of support vector machine parameters with Sequential minimal optimization (SMO), details of CSA, CSA-SVM methodology. Section 3 deals with experimental details, results and performance analysis. Finally, Section 4 concludes the proposed work

2. Materials and methods

2.1. Dataset details

The publicly available Indian liver patient dataset from University of California Irvine machine learning dataset repository [16] is used for this work. This data is collected from patients of north-east Andhra Pradesh, India. It contains 583 samples including 416 diseased liver samples and remaining 167 non-liver diseased samples. It data is tabulated with 10 input attributes and one output class attribute. The attribute details of the dataset is given in Table 1.

Table 1. Attribute details of dataset

Attribute no.	Attribute name	Attribute details
		Input attributes
1	Age	Age of the patient (all subjects greater than 89 are labelled 90)
2	Sex	Gender of the patient Female Male
3	Tot_Bil	Total Bilirubin
4	Dir_Bil	Direct Bilirubin
5	Alk_Phos	Alkaline Phosphotase
6	Alamine	Alamine Aminotransferase
7	Aspartate	Aspartate Aminotransferase
8	Tot_Prot	Total Protiens
9	Albumin	Albumin
10	A_G_Ratio	Albumin and Globulin Ratio
		Output attribute
11	Disease	Disease State (classified labeled by the medical experts) -1 = normal and 1= disease

2.2. Support vector machine

The SVM algorithm was firstly invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963. The SVM classifier is a kind of machine learning algorithm that attempts to find an optimal hyperplane with maximum margin [17]. This algorithm separates the linearly separable data samples into two classes. If the data is non-linearly separable, then SVM maps the data into high-dimensional feature space and performs the classification. The equation of the separating hyperplane is given by the Equation (1),

belongs to class 1(positive class) and 'n2' number of that belongs to class2(negative class). Each Xi contains m attributes(features), i.e., $X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,m}\}$. Yi is the actual output which may either take -1 for negative class and +1 for positive class. Let $Y_i = \{Y_1, Y_2, \ldots, Y_n\}$ represents the output class label of each record.

Step 3 Shifting the input data

For shifting the input data, first the shiftmean value is calculated. The shiftmean value is the negative of the mean of each column or attribute of the input data and it is calculated using the equation (9) as,

$$Shiftmean, \overline{\mu_j} = \left[\overline{\mu_1} = -\frac{\sum x_{i,1}}{n} \quad \overline{\mu_2} = -\frac{\sum x_{i,2}}{n} \ldots \quad \overline{\mu_m} = -\frac{\sum x_{i,m}}{n}\right], \quad i = 1,2 \ldots n, j = 1,2 \ldots m \tag{9}$$

The input data is shifted by adding the shiftmean value of each column with its corresponding column values. It is calculated using the equation (10) as,

$$Shiftdata, Sh_{ij} = \begin{bmatrix} \overline{\mu_1} + x_{1,1} & \cdots & \overline{\mu_m} + x_{1,m} \\ \vdots & \ddots & \vdots \\ \overline{\mu_1} + x_{n,1} & \cdots & \overline{\mu_m} + x_{n,m} \end{bmatrix} \quad i = 1,2,..n; j = 1,2,..m \tag{10}$$

This is to centre the data points at their mean. Shiftdata, sh is the shifted data matrix

Step 4 Scaling the input data

The scalefactor is calculated as one divided by the standard deviation of each column as per the equation (11) given below,

$$\sigma_j = \left[\sigma_1 = \frac{1}{\sqrt{\frac{\sum(x_{i,1} - \overline{\mu_1})^2}{(n-1)}}} \sigma_2 = \frac{1}{\sqrt{\frac{\sum(x_{i,2} - \overline{\mu_2})^2}{(n-1)}}} \cdots \sigma_m = \frac{1}{\sqrt{\frac{(x_{i,m} - \overline{\mu_m})^2}{(n-1)}}}\right] \quad i = 1,2 \ldots n; j = 1,2 \ldots m \tag{11}$$

The scalefactor of each column is multiplied with the shifted data matrix of its corresponding column using equation (12) as,

$$Scaledata, S_{ij} = \begin{bmatrix} \sigma_1 Sh_{1,1} & \cdots & \sigma_m Sh_{1,m} \\ \vdots & \ddots & \vdots \\ \sigma_1 Sh_{n,1} & \cdots & \sigma_m Sh_{n,m} \end{bmatrix} \quad i = 1,2 \ldots n; j = 1,2 \ldots m \tag{12}$$

The scaled data matrix is used for training the SVM classifer.

Step 5 Selection of Kernel function and calculation of the Kernel matrix

The kernel function maps the training data into kernel space. There are many kernel functions such as Linear kernel, Quadratic kernel, Polynomial kernel, Gaussian Radial Basis function and Multilayer Perceptron kernel. The kernel function is denoted by K(Si,Sj), where Si and Sj are the scaled input vectors. The calculation for Linear kernel is given in the equation (13),

$$K(S_i, S_j) = (\varphi(\vec{S_i}) . \varphi(\vec{S_j})) \tag{13}$$

The kernel matrix, Ki,jis calculated by the equation (14).

$$K_{i,j} = \begin{bmatrix} (\varphi(\vec{s_1}) . \varphi(\vec{s_1})) & \cdots & (\varphi(\vec{s_1}) . \varphi(\vec{s_n})) \\ \vdots & \ddots & \vdots \\ (\varphi(\vec{s_n}) . \varphi(\vec{s_1})) & \cdots & (\varphi(\vec{s_n}) . \varphi(\vec{s_n})) \end{bmatrix} \quad i,j = 1,2 \ldots n \tag{14}$$

The kernel matrix represents the similarities between the input vectors. It is a symmetric and positive semi-definite matrix. The x represents the input vector and K denotes the feature space vector got after the transformation. The kernel function maps the shifted and scaled training data 'S' into kernel space or the feature space vector. Linear kernel, Polynomial kernel, Gaussian Radial Basis function (RBF) are some popular kernel functions listed in Table 2.

Table 2. List of kernel functions

Kernel name	Kernel function
Linear Kernel	$K = S^T.S$
RBF kernel	$K = exp\left(-\frac{\|S - S^T\|^2}{2\omega^2}\right), \forall\ kernel\ width, \omega > 0$
Polynomial Kernel	$K = (1 + S.S^T)^d$, for any degree, d > 0

The purpose of the kernel matrix is to find out the similarities between the input vectors. It is a symmetric and positive semi-definite matrix.

Step 6 Retrieving the diagonal of the Kernel matrix

The diagonal of the kernel matrix is retrieved and given using equation (15) as,

$$K_{diag} = \begin{bmatrix} K_{1,1} \\ K_{2,2} \\ \vdots \\ K_{n,n} \end{bmatrix} \tag{15}$$

Step 7 Calculation of Box constraint values for classes

The Boxconstraint (c) is a value used in the training process to handle the trade-off between training error and complexity of the model. Further this penalty parameter is a boundary condition that decides the number of outliers accepted for the calculation of support vectors. It is of same length as the training data. It is always initialized as 1,[c = 1].It automatically rescales the samples if two groups are unbalanced. The box constraint for each class is calculated using the equation (16) and (17) as,

$$c_{1-} = \begin{cases} \frac{0.5\,cn}{n_1}, & if\ n_1 \neq n_2; \\ 0.5 & if\ n_1 = n_2. \end{cases} \quad (16)$$

$$c_{2-} = \begin{cases} \frac{0.5\,cn}{n_2}, & if\ n_1 \neq n_2; \\ 0.5 & if\ n_1 = n_2. \end{cases} \quad (17)$$

In general, smaller value of 'c' makes the classifier flat, larger value makes the training with less error and very larger values make the classifier to start overfitting. Hence an optimal c value is chosen to make the classifier retain its property of generalisation with less training error.

Step 8 Calculation of Alpha and Bias values using Sequential Minimal Optimization (SMO) algorithm

In this section, SMO is discussed to calculate the alpha and bias value. The following are the control parameters to be initialized for SMO algorithm.

In each iteration, the SMO algorithm chooses a pair of the Alpha values (α_1 and α_2) also known as the Lagrange multipliers and optimizes it by solving analytically, till convergence takes place. The existence of the equality constraints makes it impossible to optimize the variables individually which in turn only optimizes the alpha values. Likewise the Alpha values are calculated for all the datapoints, two at a time till the optimum values are obtained, based on the condition 1 that whether the maximum number of iterations reached or when the condition 2 ($(\alpha_1-\alpha_2)\leq$tolKKT) is satisfied. Then the bias or the threshold value is calculated using the equation (18) as,

$$bias = \frac{(\alpha_1-\alpha_2)}{2} \quad (18)$$

Each datapoint is associated with one alpha value which plays a vital role in qualifying the datapoints as the support vectors. The post condition is the alpha values should be greater than or equal to 0 and less than or equal to the Boxconstraint value, i.e., $0 \leq \alpha_i \leq c\ where\ i = 1,2 \ldots n$

Step 8.1 Initialization of training parameters

- Maximum number of iterations, maxIter = [1…5000]
- Tolerance by which the Karush-Kuhn-Tucker (KKT) conditions and stopping criteria are checked, tolKKT= 1.0000e-03
- Episilon(ϵ) = 2.2204e-16
- Tolerance by which support vectors are identified, svTol= square root of ϵ = 1.4901e-08
- KKTViolationLevel=[0,1], specify the fraction of variables allowed to violate the KKT conditions, it is set as 0
- Number of accepted KKT violations, acceptedKKTviolations = 0
- The initial alpha value of all datapoints are initialized with $\alpha^{old}(i)$ 0, i=1,2…n
- The object gradients of the objective function are initialized with $\Delta w = 1$
- The vectors, Box_{+1} and Box_{-1} of class1 and class2 respectively are initialized as,
- $Box_{+1}(i) = \begin{cases} c_1, if y_i = +1 \\ 0, if y_i = -1 \end{cases}$ and
- $Box_{-1}(i) = \begin{cases} -c_2, if y_i = -1 \\ 0, if y_i = +1 \end{cases}$ where $i = 1,2 \ldots n$
- The masking vectors for class1 and class2 are $I_{+1} and I_{-1}$ respectively, initialized as,

$$I_{+1}(i) = \begin{cases} 1, if y_i = +1 \\ 0, if y_i = -1 \end{cases} \quad \text{and}$$

$$I_{-1}(i) = \begin{cases} 1, if y_i = -1 \\ 0, if y_i = +1 \end{cases}$$

Step 8.2 Calculation of first Alpha value

The first alpha value α1 is calculated using the equation (19) as,

$$\alpha_1 = max(y_i \Delta w_i I_{+1}(i)), \quad i = 1,2 \ldots n \quad (19)$$

The index value of α_1 is stored in id_1

Step 8.3 Calculation of second Alpha value using Maximum gain method

The Maximum gain method is used for finding the second alpha value (α2) and its index (id_2) using α_1 and id_1 values. For that, mask values are calculated using the equation(20) as,

$$mask(i) = I_{-1}(i) P_i \text{ where } P = \begin{cases} 1, & y_i \Delta w_i < \alpha_1 \\ 0, & otherwise \end{cases}, \ i = 1,2 \ldots n \quad (20)$$

Now calculate the gain value using the equation (21),

$$gainNumerator(i) = (y_i \Delta w_i - 1)^2 mask(i)$$

$$gainDenominator(i) = -4\, k_{id1,1} + 2k_{id_2}(i,i) + 2k_{id_2}(id1)$$

$$gain = max\left(\frac{gainNumerator(i)}{gainDenominator(i)}\right), \quad (21)$$

where i=1,2,…n, gainNumerator gainDenominator≠ 0

The index value of gain value is the index of the second alpha value (id_2). The α2 is calculated in the equation (22),

$$\alpha_2 = y_{id2} \Delta w_{id2}, \quad i = 1,2 \ldots n \quad (22)$$

The alpha values are updated based on stopping condition.

Step 8.4 Checking for the stopping conditions and calculation of Bias value

The condition 1 is to check whether the maximum number of iterations (maxIter) is reached to check for stopping of

optimization process. If the condition 2 is true then the bias value is calculated using the equation (10). If the condition fails, then the Lagrange multipliers α1 and α2 are updated till it the convergence occurs. The alpha values are updated before the next iteration. The alpha calculation is stopped if any one or both the conditions are satisfied, which ever be the earliest.

Step 8.5 Updating the Alpha values based on clip limits

The bound constraints, $0 \le \alpha_i \le c_1, c_2$ causes the Lagrange multiplier (LM) to lie within a box, while the Linear equality constraints $\sum_{i=1}^{n} y_i \alpha_i = 0$ makes the LM to lie on the diagonal line segment. The ends of diagonal line is computed with the help of LM. This corresponds to the right orientation of the Hyperplane. The clip limits are calculated using the equation (23),

$$L = \begin{cases} \max(0, \alpha_2 - \alpha_1), & if y_{in1} = y_{in2} \\ \max(0, \alpha_2 + \alpha_1 - c_1), & if y_{in1} \neq y_{in2} \end{cases} \quad \text{and}$$
$$H = \begin{cases} \min(c_2, c_1 + \alpha_2 - \alpha_1), & if y_{in1} = y_{in2} \\ \min(c_2, \alpha_2 - \alpha_1), & if y_{in1} \neq y_{in2} \end{cases} \quad (23)$$

The calculation of the second derivative of the objective function, (η) along the diagonal line is given in the equation (24),

$$\eta = k_{in1,in1} + k_{in2,in2} - 2k_{in1,in2} \quad (24)$$

When $\eta > \varepsilon$, then lambda, λ value calculated using equation (25) as,

$$\lambda = -y_{in1} \Delta w_{in1} + y_{in2} \Delta w_{in2} \quad (25)$$

The new second alpha value, α_2^{new} is calculated first using the equation (26),

$$\alpha_2^{new} = \alpha^{old}(id2) + \frac{y_{in2}}{\eta} \quad (26)$$

Next the constrained minimum is found by clipping the unconstrained minimum to the ends of the line segment i.e., the $\alpha_2^{new} values is clipped$ using the equation (27),

$$\alpha_2^{new,clipped} = \begin{cases} H if \alpha_2^{new} \ge H; \\ \alpha_2^{new} if L < \alpha_2^{new} < H; \\ L if \alpha_2^{new} \le L. \end{cases} \quad (27)$$

Now the new first alpha value, α_1^{new} is calculated first using the equation (28),

$$\alpha_1^{new} = \alpha^{old}(id1) + y_{in1} y_{in2} \left(\alpha^{old}(id2) - \alpha_2^{new,clipped} \right) \quad (28)$$

The α_1^{new} is clipped using the equation (21),

$$\alpha_1^{new,clipped} = \begin{cases} 0, & if \alpha_1^{new} < \varepsilon; \\ c_1, & if \alpha_1^{new} > \varepsilon; \\ \alpha_1^{new}, & otherwise \end{cases} \quad (29)$$

These alpha values from the equation (27) and (28) are updated in the global array α^{old}.

Step 8.6 Updating the training parameters

The relevant training parameters such as $\Delta w, I_{+1} and I_{-1}$ of the (19) and (20) are updated using the equation (30).

$$\Delta w_i = \Delta w_i - k_{i,in1} y_i y_{in1} \left(\alpha_1^{new,clipped} - \alpha_{in1}^{old} \right) - k_{i,in2} y_i y_{in2} \left(\alpha_2^{new} \right.$$

$$I_{+1}(id1) = \begin{cases} 1, & if y_{in1} \alpha_1^{new,clipped} < Box_{+1}(id1) - svTol \\ 0, & otherwise \end{cases}$$
$$I_{+1}(id2) = \begin{cases} 1, & if y_{in2} \alpha_2^{new,clipped} < Box_{+1}(id2) - svTol \\ 0, & otherwise \end{cases}$$
$$I_{-1}(id1) = \begin{cases} 1, & if y_{in1} \alpha_1^{new,clipped} > Box_{-1}(id1) + svTol \\ 0, & otherwise \end{cases}$$
$$I_{-1}(id2) = \begin{cases} 1, & if y_{in2} \alpha_2^{new,clipped} > Box_{-1}(id2) + svTol \\ 0, & otherwise \end{cases}$$

where i = 1,2 …n. (30)

Proceed with the next iteration till the condition1 and 2 gets satisfied.

Step 9 Evaluation of the value of the objective function in Equation (7)

Step 10 Calculation of the Support vectors

The training datapoints are qualified as support vectors of size q with the help of alpha values updated at the end of the training process using the equation (31) as,

$$svindices(j) = index(alphas(j) > \sqrt{\varepsilon})$$
$$support\_vectors(j) = K_{svindices(j)}, \ j = 1,2 \ldots q \quad (31)$$

The kernel matrix with svindices are the support vectors

Step 11 Discrimination of alpha values

The alpha values, α^{old} are classified based on its class labels as α^{bar} using the equation (32),

$$\alpha^{bar}(j) = y(svindices(j)) \alpha^{old}(svindices(j)), \ j=1,2\ldots q \quad (32)$$

This is the final alpha values used for testing process.
Testing phase

Step 12 Load the test data

The test dataZi={Z1,Z2,….Zn} with m attributes as that of training data is loaded for testing the SVM classifier.

Step 13 Shift the test data

The test data is shifted using the equation (33) from the shiftmean calculated from equation (9),

$$Shiftdata_{test}, sh_{test_{i,j}} = \begin{bmatrix} \overline{\mu_1} + z_{1,1} & \cdots & \overline{\mu_m} + z_{1,m} \\ \vdots & \ddots & \vdots \\ \overline{\mu_1} + z_{n,1} & \cdots & \overline{\mu_m} + z_{n,m} \end{bmatrix}$$

$$i = 1,2,..n; j = 1,2,..m \quad (33)$$

Step 14 Scale the test data
The shifted test data is scaled using the equation (34) with the scaling factor derived in equation (11),

$$Scaledata_{test}, S_{test_{i,j}} = \begin{bmatrix} \sigma_1 sh_{test_{1,1}} & \cdots & \sigma_m sh_{test_{1,m}} \\ \vdots & \ddots & \vdots \\ \sigma_1 sh_{test_{n,1}} & \cdots & \sigma_m sh_{test_{n,m}} \end{bmatrix}$$

$$i = 1,2 \ldots n; j = 1,2 \ldots m \quad (34)$$

Step 15 Classification of the test data
The classification function is evaluated and the sign of it denotes the classification of data into class1 and class2. The classification function is given using the equation (35),

$$Output(j) = sign\left(\sum_{j=1}^{n} \alpha_j^{bar} K_{m,z} + bias\right) \quad (35)$$

where $K_{m,z}$ is the kernel matrix which gives the similarity or distance between the support vectors and the testing data.If the sign of the output of particular test data is positive, then it belongs to class 1 and else if it is negative, then it belongs to class 2.

2.4. Crow search algorithm

CSA is the most recently developed algorithm by Alireza Askarzadeh in the year 2016 [19]. It is inspired based on intelligent stealing behaviour of clever bird crow. The crows hide extra food in hiding places and retrieve when needed. A crow follows the other that has better food sourceinorder to steal it. From its own stealing experience, it also tries to avoid being a future victim. These behavioural characteristics of crows are simulated as metaheuristic optimization algorithm. The flock of crows forms the population (N). Each crow X_i, [i=1, 2,…N] is considered as search agent, the environment as search space, the hiding places as certain positions which corresponds to feasible solution, the fitness function is based on food source quality where best food source is global best solution.

A d-dimensional environment is assumed, that is each crow is considered as d-dimensional vector. The position of crow i at iteration iter is given as $X^{i,iter}$ where iter=1,2,…max_iter.

$$X^{i,iter} = \left[X_1^{i,iter}, X_2^{i,iter} \ldots X_d^{i,iter}\right] \quad (36)$$

Each crow is associated with memory to memorize the information of its hiding places. The memory of i^{th} crow in iteration iter is given as $m^{i,iter}$. This is considered as best position achieved so far, based on fitness value calculated at each iteration. The crows have the habit of following the other to find their hiding places to steal food. Based on their behavioural strategy, two cases are formulated to update their position. Assume crow i follows crow j,

- Case 1: If crow j does not know that it is followed by crow i, it reaches its hiding place which is also reached by crow i. Hence position is updated for crow i. The new position of crow i is calculated as,

$$X^{i,iter+1} = X^{i,iter} + r_i * fl^{i,iter}(m^{j,iter} - X^{i,iter}) \quad (37)$$

where r_i is random number and $fl^{i,iter}$ denotes flight length of crow i at iteration iter.

- Case 2: If crow j notices that it is followed by crow i, then it tries to fool crow i by reaching some other location randomly. Now the new position of crow i is updated with random value. The case 1 and case 2 depends on value of awareness probability (AP).

2.5. CSA-SVM Methodology

The procedure of CSA-SVM is given below,

Step 1 Initialize population size (N), Maximum iteration (Max_iter), awareness probability (AP), flight length (fl).
Step 2 Initialize the crow population using random values (position of crows).
Step 3 Initialize the memory of crows. For first iteration, its initial positions are considered as memory.
Step 4 The quality of position of each crow is evaluated using the fitness function (Equation 7)
Step 5 The new position of each crow is generated based on two cases, case 1 and case 2.
Step 6 Feasibility of new positions are checked. If it is found better than current, then position update takes place else current one is saved.
Step 7 Fitness of each crow for new position is calculated.
Step 8 The memory of each crow is updated by comparing the new fitness value with memorized one. It is updated with the better one.
Step 9 Check for stopping criterion, that reaching maximum iteration.
Step 10 Steps 5-9 are repeated till max_iter is reached.
Step 11 The optimal or best solution is achieved.

3. Experimental details and discussions

The most widely used algorithms such Genetic algorithm (GA) [20], Multi-verse optimizer (MVO) [21], Firefly algorithm (FA) [22] and PSO [23] are used to optimize

SVM and applied for classification of liver dataset. The results are compared with proposed CSA-SVM. The proposed CSA-SVM and all other comparison SVM hybrids such as MVO-SVM, GA-SVM, FA-SVM and PSO-SVM algorithms are developed in MATLAB R2015b installed in machine with Intel core i5 processor of speed of 2.7 GHz and 4 GB RAM. The details of other MHOAs such as GA, MVO, FA and PSO that are taken for comparison are out of scope for this work and are not further discussed here. As it is necessary to reveal the values that are set for the control parameters of comparison MHOAs used in the experiments, the details regarding it are given in Table 3. All the experiments are conducted using ten-fold cross validation method and averages of results are tabulated in Table 4.

Table 3. Setting of control parameters

Algorithm	Parameter	Value
CSA	flight length (fl) awareness probability (AP)	2.5 0.05
MVO	Min wormhole existence ratio Max wormhole existence ratio	0.2 1
FA	light absorption coefficient attractiveness	1 1
PSO	Acceleration constants (C1,C2) Inertia weight (w)	(2, 2) 0.8
GA	Selection Cross over probability Mutation probability	Roulette wheel 1 0.01

The performance measures such as sensitivity, specificity, precision and accuracy are used to scale the performance along with standard deviation (SD) [24]. The results from the experiments clearly shows that CSA-SVM has the best diagnostic capability than all other hybrid SVMs. It has produced accuracy, specificity, sensitivity and precision of 99.49±0.12, 98.80±0.33, 99.76±0.21 and 99.52±0.51 respectively.

Table 4. Performance comparison of SVM classifiers

Approach	Sensitivity ± SD(%)	Specificity ± SD (%)	Precision± SD (%)	Accuracy± SD (%)
CSA-SVM	99.76±0.21	98.80±0.33	99.52±0.51	99.49±0.12
MVO-SVM	96.15±1.87	92.22±2.11	96.85±0.69	95.03±2.01
GA-SVM	93.99±1.99	95.81±1.32	98.24±1.86	94.51±1.93
FA-SVM	89.66±3.41	79.04±2.52	91.42±2.21	86.62±2.64
PSO-SVM	84.13±2.43	88.02±2.94	94.59±1.57	85.25±2.28

The performance of MVO-SVM and GA-SVM are found to be a little closer. MVO-SVM has the second highest in sensitivity and accuracy as 96.15±1.87 and 95.03±2.01 respectively. GA-SVM is next highest to CSA-SVM in specificity and precision values as 95.81±1.32 and 98.24±1.86 respectively. FA-SVM is competitive with PSO-SVM in diagnostic accuracy. PSO-SVM has yielded least performance in terms of accuracy and sensitivity as 85.25±2.28 and 84.13±2.43 respectively, and this shows that it has not correctly predicted the most positive samples. FA-SVM is least in terms of specificity and precision as 79.04±2.52 and 91.42±2.21 respectively, and from this it is found that it has given the least discrimination ability toward the negative samples. The results prove that CSA-SVM has produced outstanding performance than all classifiers used for comparison. This is plotted in a graph with performance metrics in X-axis versus scaling (in percentage) in Y-axis. This details are shown using Figure 2.

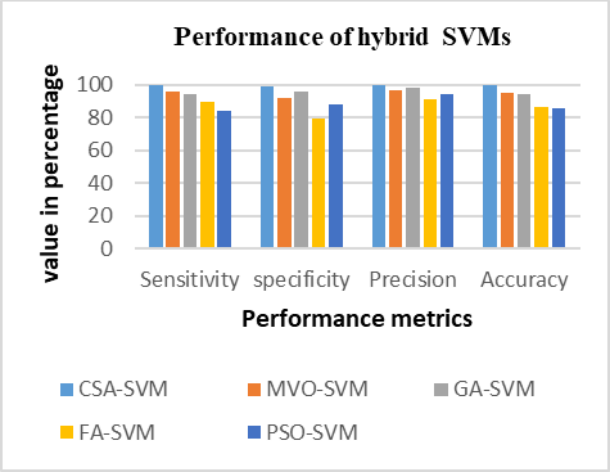

Figure 2. Performance chart of hybrid SVMs

Table 5. Performance comparison of works in literature on Liver disease diagnosis

Works in literature	Approach	Performance (Accuracy)
[1]	NB	60.6%
	K-star	67.2%
	J-48	71.2%
	RT	74.2%
[2]	Logistic regression	73.5%
	SVM	70.94%
	RT	66.66%
	Adaboost	74.35%
	Bagging	72.64%

[3]	MLFFDNN	98%
[4]	XGBoost algorithm	99%
[5]	SVM	73.96%
	K-NN	74.67%
[6]	PSO-SVM	94.42%
[25]	NeuroSVM (SVM + ANN)	98.83%
[26]	Boosted C5.0	93.75%
	CHAID algorithm	65%
[27]	Naïve Bayes(NB)	53.90%
	Decision trees	69.40%
	Multi-layer perceptron (MLP)	67.90%
	k-Nearest Neighbor	65.30%
[28]	J-48	68.78%
	MLP	68.26%
	Random Forest	70.30%
	Bayesian network	67.24%
[29]	Bagging	69.30%
	IBK	64.49%
	J-48	68.78%
	J-Rip	66.38%
	MP	68.95%
	NB	55.75%
Proposed method	CSA-SVM	99.49%

The several works on Liver disease data using various algorithms in literature along with proposed CSA-SVM are tabulated in Table 5 based on accuracy produced by them. This also shows that CSA-SVM has produced better result when compared with others.

4. Conclusion

The optimization of SVM parameters with SMO is dealt. In this work, the lagrange values of support vector machines are optimized using the crow search algorithm. This optimized CSA-SVM classifier applied for the efficient diagnosis of Liver disease. It is noticed that the procedure to optimize SVM with CSA is simpler than with that of SMO. The experiments are carried out using ten-fold cross validation method. Many similar SVM hybrids are taken for comparing the efficiency of CSA-SVM. It is experimentally found that CSA-SVM has good discrimination ability on the liver disease data in terms of performance metrics such as sensitivity, specificity, precision and accuracy. Also the results of various algorithms that are used for liver disease diagnosis in literature are also compared. The overall classification accuracy produced by CSA-SVM is 99.49% which is the highest value. Finally, it is found that CSA-SVM has produced outstanding results than that of other approaches in liver disease data diagnosis. This approach can also be recommended to be used for other disease diagnosis. It is proved that it can help the medical domain in earlier accurate diagnosis of diseases based on the results produced in this proposed work.

References

[1] Muthuselvan S, Rajapraksh S, Somasundaram K, Karthik K, “Classification of Liver Patient Dataset Using Machine Learning Algorithms,” International Journal of Engineering & Technology, vol. 7, 2018, pp. 323-326.

[2] Idris K, Bhoite S, “Applications of Machine Learning for Prediction of Liver Disease,” International Journal of Computer Applications Technology and Research, vol. 8, 2019, pp. 394-396.

[3] Murty S V, Kumar R K, “Enhanced classifier accuracy in liver disease diagnosis using a novel multi layer feed forward deep neural network,” International Journal of Recent Technology and Engineering, vol. 8, 2019, pp. 1392-1400.

[4] Murty S V, Kumar R K, “Accurate Liver Disease Prediction with Extreme Gradient Boosting,” International Journal of Engineering and Advanced Technology, vol.8, 2019, pp. 2288-2295.

[5] Kumar P, Thakur R S, “Early Detection of the Liver Disorder from Imbalance Liver Function Test Datasets,” International Journal of Innovative Technology and Exploring Engineering, vol. 8, 2019, pp. 179-186.

[6] Joloudari J H, Saadatfar H, Dehzangi A, Shamshirband S, “Computer aided decision-making for predicting liver disease using PSO-based optimized SVM with feature selection,” Informatics in medicine unlocked, vol. 17, 2019, 100255

[7] Damaševičius R, “Optimization of SVM parameters for recognition of regulatory DNA sequences,” TOP, vol. 18, 2010, pp. 339-353.

[8] Wu J-H, Wei W, Zhang L, Wang J, Damaševičius R, Li J, Wang H-D, Wang G-L, Zhang X, Yuan J-X, Woźniak M, “Risk assessment of hypertension in steel workers based on LVQ and fisher-SVM deep excavation,” IEEE Access, vol. 7, 2019, pp. 23109-23119

[9] Blondin J, Saad A, "Metaheuristic techniques for Support Vector Machine model selection," *2010 10th International Conference on Hybrid Intelligent Systems*, IEEE, Atlanta, GA, 2010, pp. 197-200.

[10] Huang H , Feng X, Zhou S , Jiang J, Chen H, Li Y, Li C, “A new fruit fly optimization algorithm enhanced support vector machine for diagnosis of breast cancer based on high-level features,” BMC Bioinformatics, vol. 20, 2019, (Suppl 8):290

[11] Yang X S, Deb S, Fong S, “Accelerated Particle Swarm Optimization and Support Vector Machine for Business Optimization and Applications,” in: Networked Digital Technologies (NDT2011), Communications in Computer and Information Science, Vol. 136, Springer, pp. 53-66 (2011).

[12] Faris H, Hassonah M A, Al-Zoubi A M, Mirjalili S, Aljarah I, “A multi-verse optimizer approach for feature selection and optimizing SVM parameters based on a

robust system architecture," Neural Comput&Applic vol. 30, 2018, pp. 2355–2369.
[13] Sartakhti J S, Afrabandpey H, Saraee M H, "Simulated annealing least squares twin support vector machine (SA-LSTSVM) for pattern classification", Soft Computing - A Fusion of Foundations, Methodologies and Applications, 21 (15) , 2016, pp. 4361-4373.
[14] Aljarah I, Al-Zoubi A M, Faris H, Hassonah M A, Mirjalili S, Saadeh H," Simultaneous Feature Selection and Support Vector Machine Optimization Using the Grasshopper Optimization Algorithm," Cognitive Computation, June 2018, Vol. 10, pp 478–495.
[15] Chao C-F, Horng M-H, "The Construction of Support Vector Machine Classifier Using the Firefly Algorithm," Hindawi Publishing Corporation Computational Intelligence and Neuroscience Volume 2015, Article ID 212719, 8 pages.
[16] Dua D, Graff C, UCI Machine Learning Repository, 2019 Irvine, CA: University of California, School of Information and Computer Science, [http://archive.ics.uci.edu/ml].
[17] Corinna C, Vladimir V, "Support Vector Networks," Machine Learning, vol. 20,1995, pp. 273-297.
[18] Platt J, Fast Training of Support Vector Machines using Sequential Minimal Optimization, in Advances in Kernel Methods – Support Vector Learning, MIT Press, 1998.
[19] Askarzadeh A, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm", Computers and Structures, vol. 169, 2016, pp. 1–12.
[20] Holland J H, Adaptation in Natural and Artificial Systems, University of Michigan Press, USA, 1975.
[21] Mirjalili S, Mirjalili S M, Hatamlou A, "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization," Neural Comput&Applic, vol. 27, 2016, pp. 495–513.
[22] Yang X-S, "Firefly algorithms for multimodal optimization," in Proceedings of the 5th International Conference on Stochastic Algorithms: Foundation and Applications (SAGA '09), Sapporo, Japan, October 2009, vol. 5792 of Lecture Notes in Computer Sciences, pp. 169–178, Springer, Berlin, Germany, 2009.
[23] Kennedy J, Eberhart R, "Particle swarm optimization," in Proc. IEEE Int. Conf. on Neural Network, Perth, Australia, 1995, pp. 1942–1948
[24] Han J, Kamber M, "Data mining concepts and techniques", 2nd ed., Morgan Kaufmann Publishers Inc. San Francisco. CA. USA, 2001.
[25] Kalyan N, Sridhar A, "NeuroSVM: A Graphical User Interface for Identification of Liver Patients," International Journal of Computer Science and Information Technologies, Vol. 5, 2014, pp. 8280-8284.
[26] Abdar M, Moghadam M Z, Das R, Ting I-H, "Performance analysis of Classification Algorithms on early detection of Liver disease," Expert Systems With Applications, vol.67,2017, pp. 239-251.
[27] Hoon J, Seoungcheon K, Jinhong K, "Decision Factors on Effective Liver Patient Data Prediction," International Journal of Bio-Science and Bio-Technology, vol. 6, 2014, pp. 167-178.
[28] Gulia A, Vohra R, Rani P, "Liver Patient Classification Using Intelligent Techniques," International Journal of Computer Science and Information Technologies, vol.5,2014, pp. 5110-5115.
[29] Ramana B V, Boddu R S K, "Performance Comparison of Classification Algorithms on Medical Datasets," *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, 2019, pp. 140-145.