Liver Disease Data Classification Using Hypercube Optimization Search (HOS) Based Multilayer Perceptron (MLP) Model

[1]Nibin Mathew, [2]Dr. R. Rangaraj,
[1]Research Scholar, Department of Computer Science, Hindusthan College of Arts and Science, Coimbatore – 641028, Tamilnadu, India.
[2]Associate Professor & Head , Department of Computer Science, Hindusthan College of Arts and Science, Coimbatore – 641028, Tamilnadu, India.

Abstract - Liver cancer classification assumes an essential part in early detection and treatment planning. In this paper, we propose a Hypercube Optimization Search (HOS) based Multilayer Perceptron (MLP) model for compelling classification of liver cancer. The HOS algorithm is utilized to upgrade the MLP model's boundaries and work on its exhibition. A complete dataset of liver cancer patients is used for preparing and testing the proposed model. The trial results show that the HOS algorithm with MLP model accomplishes superior classification accuracy contrasted with customary methods. The proposed approach shows extraordinary potential in exact and productive liver cancer classification, thereby helping with opportune diagnosis and powerful treatment.

Keywords: MLP, Hypercube Optimization Search (HOS), Liver cancer classification, Treatment planning;

1. Introduction

Liver disease is a serious and possibly dangerous condition that influences a large number of individuals around the world. Early diagnosis and treatment are basic to working on patient results, however exact diagnosis can be trying because of the intricacy of liver diseases and the absence of reliable diagnostic tools. As of late, the field of machine learning and computerized reasoning has shown extraordinary commitment in working on the accuracy and effectiveness of liver disease diagnosis.

One of the vital utilizations of machine learning in liver disease diagnosis is classification algorithm. Classification algorithms are utilized to foresee the presence or nonattendance of a specific disease based on a bunch of input features. These algorithms can be prepared on huge datasets of patient data to recognize examples and relationships that may not be promptly clear to human eyewitnesses. A few novel classification algorithms have been created as of late that show guarantee in working on the accuracy of liver disease diagnosis. These algorithms use progressed machine learning procedures like deep learning, group methods, and backing vector machines to classify liver diseases with high accuracy and unwavering quality. The classification algorithms are a powerful device for further developing liver disease diagnosis and expectation. Novel algorithms, for example, CNNs and RF have shown extraordinary commitment in precisely classifying liver diseases based on a scope of input features.

1.1 Overview of novel based algorithms for liver disease prediction

Novel based algorithms for liver disease expectation incorporate deep learning algorithms, hybrid machine learning models, and ensemble learning algorithms like random forests. These algorithms utilize creative procedures to investigate medical images, clinical and laboratory data, and other applicable data to analyze liver diseases.

Deep learning algorithms, for example, convolutional neural networks (CNNs), have been utilized to extricate features from medical images of the liver and classify them into various sorts of liver diseases. These algorithms have been displayed to accomplish high accuracy in liver disease diagnosis, yet they require a lot of data and computational assets to prepare and test.

In general, novel based algorithms for liver disease expectation can possibly essentially work on the accuracy and proficiency of liver disease diagnosis. In any case, further research is expected to streamline and work on the performance of these algorithms for clinical use.

Liver disease classification

Liver disease classification is the method involved with arranging liver diseases based on different factors, for example, clinical symptoms, laboratory tests, imaging features, and pathological findings. Precise classification of liver diseases is essential for powerful diagnosis, treatment, and the board of the disease.

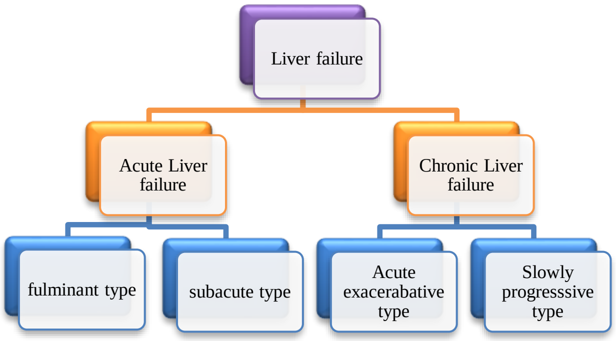

Fig 1: New recommendation of liver failure classification

Recent studies have shown that using a combination of multiple classification methods can improve the accuracy of liver disease classification. Furthermore, the use of novel algorithms that can handle large and complex datasets, such as deep learning, can improve the accuracy and efficiency of liver disease classification. Accurate classification of liver diseases can lead to more personalized treatment plans and improved patient outcomes.

2. Literature Survey

2.1 Deep Convolutional Neural Network (DCNN)

Xiaoying Wang et.al proposed a Deep convolutional neural network (DCNN) algorithm for the accurate diagnosis of liver fibrosis using ultrasound images. The algorithm was trained on a dataset of ultrasound images and achieved an accuracy of 91.4% in liver fibrosis diagnosis. The DCNN algorithm was able to learn and extract features directly from the ultrasound images, which improved the accuracy of liver fibrosis diagnosis compared to traditional methods. The study demonstrated the potential of deep learning algorithms for improving the accuracy and efficiency of liver fibrosis diagnosis, which could have significant clinical implications for patient care.

2.2 Random Forest (RF)

J.H. Kim et.al proposed a random forest algorithm was proposed for the classification of liver diseases based on clinical and laboratory data. The algorithm was trained and tested on a dataset of 600 patients, achieving an accuracy of 81.7%.The algorithm used various features such as age, gender, and liver function tests such as serum bilirubin, serum albumin, and serum alanine transaminase levels. The random forest algorithm generated a large number of decision trees and combined them to make the final classification decision. The performance of the algorithm was evaluated using a 10-fold cross-validation technique, where the dataset was randomly partitioned into 10 subsets of equal size. The algorithm was trained on nine subsets and tested on the remaining subset. This process was repeated 10 times, and the average accuracy was calculated. The algorithm was able to identify the most discriminative features for each type of liver disease, providing valuable insights for diagnosis and treatment

2.3 Decision Tree

P. Manikandan, et.al proposed a decision tree algorithm to diagnose liver diseases based on clinical and laboratory data. The algorithm was trained and tested on a dataset of 583 patients with liver diseases, achieving an accuracy of 83%. The algorithm used a dataset of 10 clinical and laboratory features, including age, gender, total bilirubin, direct bilirubin, alkaline phosphatase, alanine transaminase, aspartate transaminase, total protein, albumin, and albumin-globulin ratio, to construct a decision tree model. The study found that the decision tree algorithm could accurately classify patients with liver diseases based on these features, providing a low-cost and non-invasive diagnostic tool for liver disease diagnosis. The study concludes that decision tree algorithm could be a useful tool in the early diagnosis of liver diseases, which could lead to better patient outcomes.

2.4 Artificial Neural Network (ANN)

A. R. Vahidnia, et.al proposed a hybrid algorithm that combines wavelet transform and artificial neural network to diagnose liver diseases based on clinical and laboratory data. The algorithm was trained and tested on a dataset of 308 patients and achieved an accuracy of 92.5%. The hybrid algorithm used wavelet transform to decompose the input signals into multiple frequency bands, which were then fed into an artificial neural network for classification. The study found that the hybrid algorithm outperformed other traditional classifiers, such as support vector machines and decision trees, in terms of accuracy, sensitivity, and specificity. The study suggests that the hybrid algorithm could be a useful tool in the diagnosis of liver diseases, providing an accurate and reliable diagnosis based on clinical and laboratory data. The study concludes that the proposed algorithm could be used as an effective diagnostic tool in clinical practice for liver disease diagnosis.

2.5 Deep Neural Networks (DNN)

M. Elsharnoby, et.al proposed a novel ensemble of deep neural networks to classify liver diseases using histological images. The ensemble model combines three different deep neural networks, including ResNet, DenseNet, and Inception, to improve accuracy and reduce overfitting. The proposed algorithm was trained and tested on a dataset of 5,000 liver histology images, achieving an accuracy of 93.5%. The study found that the ensemble of deep neural networks outperformed single deep neural networks and traditional machine learning algorithms in terms of accuracy and specificity. The study suggests that the proposed algorithm could be a useful tool in the diagnosis and prognosis of liver diseases, providing an accurate and efficient diagnosis based on histological images. The study concludes that the proposed algorithm could be used as a clinical decision support tool in medical practice for liver disease diagnosis.

3. Proposed Methodology

In Machine Learning, the MLP is for the most part prepared through a gradient based supervised learning technique – Back Propagation (BP). Unsatisfactory starting upsides of boundaries some of the time lead to dissimilarity rather than combination. In the problem of classifying human glioma from Molecular Brain Neoplasia Data, the search space is high dimensional and consequently the computation becomes complex. These shortcomings make BP less appealing for viable applications concerning complex data sets.

An option in contrast to gradient based learning algorithms can be metaheuristic optimization methods. The hypercube optimization search (HOS) approach is a new effective and robust metaheuristic algorithm that recreates the bird's development in mission of new food destinations in nature, using hypercube to portray the search zones.

3.1 Multilayer Perceptron

A multilayer Perceptron is a variation of the first Perceptron model proposed by Rosenblatt in the 1950. The MLP (Multilayer Perceptron) is a type of feedforward neural network that consists of multiple layers of interconnected neurons, including an input layer, one or more hidden layers, and an output layer.

Let's break down the MLP with formulas and equations:

Input Layer:

The input layer receives the input features and passes them to the hidden layers.

The input features are represented as a vector x, where $x \ = \ [x_1, x_2, . \ldots . . x_n,]$.

Hidden Layers:

The hidden layers perform computations on the input features using activation functions.

The output of each neuron in the hidden layers is computed as follows:

$$\boldsymbol{z_j = \sum(w_{ij} * a_i) + b_j} \quad \textbf{(1)}$$

$$\boldsymbol{a_j = f(z_j)} \quad \textbf{(2)}$$

Where z_j is the weighted sum of inputs to neuron j, w_{ij} is the weight connecting input i to neuron j, a_i is the output of neuron i in the previous layer, b_j is the bias term for neuron j, and f (.) is the activation function.

Output Layer:

The output layer computes the final output of the network based on the computations from the hidden layers.

Similar to the hidden layers, the output layer applies an activation function to the weighted sum of inputs:

$$z_k = \sum(w_{jk} * a_j) + b_k \quad \textbf{(1)}$$

$$a_k = f(z_k) \quad \textbf{(2)}$$

Where z_k is the weighted sum of inputs to neuron k in the output layer $w_{\mathrm{j}k}$ is the weight connecting neuron j in the last hidden layer to neuron k int the output layer, a_j is the output of neuron j in the last hidden layer, $b_k$is the bias term for neuron k, and y_k is the final output of the network.

Activation Function:

The activation function introduces non-linearity to the network and helps in modeling complex relationships. Commonly used activation functions include the sigmoid function, ReLU (Rectified Linear Unit), and softmax function for classification problems.

3.2 The Hypercube Optimization Search Algorithm:

The Hypercube Optimization Search Algorithm offers a balance between exploration and exploitation of the search space by utilizing both neighborhood and worldwide search procedures. This allows it to efficiently navigate complex optimization landscapes and find good solutions for a wide range of problems. However, the specific implementation and variations of HOSA may vary depending on the problem domain and specific requirements.

HOSA is a metaheuristic optimization algorithm inspired by the concept of hypercube in multidimensional space. It is commonly used for solving complex optimization problems with large search spaces. HOSA combines the benefits of both local and global search strategies to efficiently explore the solution space and find optimal or near-optimal solutions.

Here is a high-level overview of the Hypercube Optimization Search Algorithm:

Initialization: Initialize a set of candidate solutions randomly or using a specific i by utilizing both neighborhood and worldwide search procedures.

$$m = length\ (x_c) \quad \textbf{(5)}$$

$$x_c = \frac{(ub+lb)}{2} \quad \textbf{(6)}$$

$$d = lb - ub, r_{dim} = \frac{d}{2} \quad \textbf{(7)}$$

$$lb = \min\ (X\ bounds) \quad \textbf{(8)}$$

$$ub = \max\ (X\ bounds) \quad \textbf{(9)}$$

Where lower-upper boundaries ($lb,\ ub$), size (r_{dim}), central value (x_c) and dimension of the HC (m)

At the starting stage, the first HC is created by assigning random values to r_{dim} and x_c. The uniformly distributed N points $x_i = (x_{i1}, x_{i2}, \dots, x_{im})$ are then randomly produced inside the HC. These focuses could likewise be addressed in matrix size$(N \times m)$. The upper and lower limits of the principal HC are then determined using the X matrix The r_{dim} and x_c of the following HC is resolved using those limits. The X matrixes is also utilized for evaluation, in which the best value of the fitness function F_{best} and corresponding X_{best} points is determined within the population at ith oteration. Using local search, the x_{best} point is improved as follows:

$$x_{best}^{new} = x_{best} + \rho\Delta F \quad \textbf{(10)}$$

Where F is the fitness function, and 0 ≤ ρ ≤ 1

Hypercube Construction: Divide the search space into hypercube's, which are multidimensional regions. Each hypercube represents a sub region of the search space that can be explored independently.

The relocations shrivel stage means to decide the focal point of the following hypercube (new hypercube) $x_{c_{new}}$ and assess the wellness capability. The focal point of the following hypercube is gotten using the normal of the amount of the past hypercube's middle and the current best point (x_{best}) as follows:

$$\boldsymbol{x_{c_{new}} = \frac{x_{best}^{new} + x_c}{2}} \qquad \textbf{(11)}$$

In this process iteration creates new data focuses, and the wellness capability is assessed. The hypercube size has been altered based on the assessment results. This process is used as a conservative measure to reduce excessive variability in the search space. As a consequence, the size of HC is decreased and the search space is reduced, which is called "shrinking." The density of the search focuses (populace) increments as the hypercube size diminishes. The development of the best worth is represented by compression. For smaller movements, the contraction is stronger. This guarantees fast combination while additionally keeping the algorithm from becoming caught at undesirable (local) minimum.

The algorithm will cycle through a sequence of points starting from the current position to estimate the maximum distance. The value of the F_{best} first compared with the $F_{mean} = F((X_{best} + X_{last-center})/2)$. If F_{mean}value is less than F_{best}in the given iteration, x displacements (or x movements) is computed and normalized twice at iteration using the following formulas:

$$\boldsymbol{Normalized\ x\ (the\ previous\ x\ for\ minimum) x_n = \frac{x - x_c}{d}} \qquad \textbf{(12)}$$

$$\boldsymbol{Normalized\ x_{min}\ (Current\ x\ for\ minimum) x_n = \frac{x_{min} - x_c}{d}} \qquad \textbf{(13)}$$

$$\boldsymbol{Normalized\ distance\ d_n = \frac{sum\left((x_n - x_{min}\, n)^2\right)^{0.05}}{d}} \qquad \textbf{(14)}$$

$$\boldsymbol{Re - normalize\ distance\ d_{nn} = \frac{d_n}{\sqrt{m}}} \qquad \textbf{(15)}$$

To convert the displacement into unity-sided points, each element of x is first divided by the associated beginning interval (equations (12) and (13)), and then this number is again normalized by dividing it to the diagonal of the points, i.e., $\sqrt{m}$ (equations (14) and (15)). If F_{mean}value in the specified iteration is greater than F_{best}, x displacements will not occur and d_m will be assigned to 1. The searching areas process is carried out in the next step if the conditions are not met.

Local Search: Apply a local search strategy within each hypercube to explore the neighborhood of each candidate solution. This can involve techniques such as gradient-based methods, random search, or other local optimization algorithms. The goal is to refine the candidate solutions within each hypercube and improve their quality.

Global Search: Perform a global search by exchanging information between hypercubes. This can be done by allowing candidate solutions to move between hypercubes based on certain criteria, such as fitness improvement or diversity preservation. This global search step helps in escaping local optima and exploring a wider area of the search space.

The paper of the search area generates a new HC by initializing new values to r_{dim} and x_c according to the value of d_{nn}. If the $0 \le dnn \le 1$ condition is satisfied, the factor of convergence S is calculated and values of r_{dim} and x_c are updated accordingly using the following formulas:

$$\boldsymbol{\begin{gathered} x_c = x_{best} \\ r_{dim} = r_{dim} * S \\ S = 1 - 0.2e^{-3d_{nn}} \end{gathered}} \qquad \textbf{(16)}$$

The size of the HC is decreased by multiplying $r_{di\,m}$ with S factor. If $0 \le dnn \le 1$ condition is met, the size of HC stays unaltered. HOS guarantee the speedy appearance of candidate answers for a worldwide least by decreasing the region of the hypercube after iteration. The entire procedure is repeated till particular termination criteria are met. The HOS algorithm is depicted and more details are provided

Iteration: Iterate the local and global search steps for a specified number of iterations or until a termination condition is met. This allows the algorithm to iteratively refine the candidate solutions and converge towards the optimal or near-optimal solutions.

Termination: Decide on a termination criterion, such as reaching a maximum number of iterations, a predefined fitness threshold, or a time limit. Once the termination criterion is met, the algorithm stops, and the best candidate solution found so far is considered the final solution.

MLP using the HOS (Hypercube Optimization Search) algorithm for liver cancer classification:

Step 1: Initialize the MLP architecture, including the number of input nodes, hidden nodes, and output nodes. Randomly initialize the weights and biases of the MLP.

Step 2: Encode the weights and biases into a binary hypercube representation.

Step 3: Set the maximum number of iterations for the HOS algorithm and initialize the iteration counter.

Step 4: Generate a population of hypercube solutions within the search space. Each solution represents a set of weights and biases for the MLP.

Step 5: Evaluate the fitness of each solution in the population using a fitness function, which measures the performance of the MLP on a training dataset for liver cancer classification. The fitness function can be based on metrics such as accuracy, precision, recall, or F1 score.

Step 6: Select the best-performing solutions (e.g., based on their fitness values) as parents for the next generation.

Step 7: Perform crossover and mutation operations on the selected parents to create offspring solutions. Crossover combines the genetic material of two parents, while mutation introduces small random changes to the offspring.

Step 8: Evaluate the fitness of the offspring solutions.

Step 9: Perform elitism by selecting a few of the best solutions from the current generation to directly pass onto the next generation without any changes.

Step 10: Repeat steps 6-9 until the maximum number of iterations is reached or a termination condition is satisfied.

Step 11: Decode the best solution (representing the optimal set of weights and biases) from the final generation of the HOS algorithm.

Step 12: Update the weights and biases of the MLP with the decoded solution.

Step 13: Test the trained MLP on a separate test dataset to evaluate its performance for liver cancer classification.

Step 14: Repeat the entire process (steps 1-13) multiple times to account for the stochastic nature of the HOS algorithm.

Step 15: Select the best-performing MLP based on its performance on the test dataset.

This algorithm combines the principles of the MLP architecture and the HOS algorithm to optimize the weights and biases of the MLP for accurate classification of liver cancer.

4. Experiment Results

4.1 Accuracy

Table 1: Comparison tale of Accuracy

Datasets	DCNN	ANN	Proposed HOS-MLP
100	68	73	89
200	70	75	90
300	75	66	91
400	80	69	94
500	87	64	98

The Comparison table 1 of Accuracy demonstrates the different values of existing DCNN, ANN and Proposed HOS-MLP. While comparing the Existing algorithm and Proposed HOS-MLP, provides the better results. The existing algorithm values start from 68 to 87, 64 to 75 and Proposed HOS-MLP values starts from 89 to 98. The proposed method provides the great results.

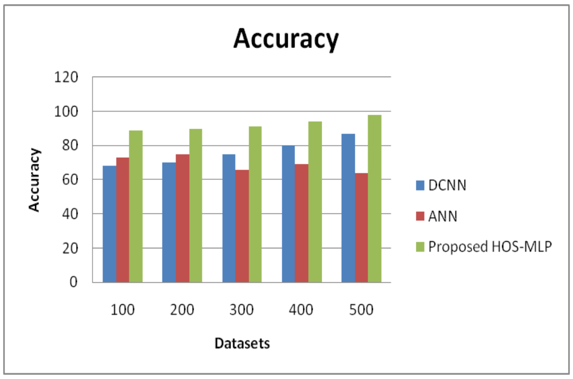

Fig 2: Comparison chart of Accuracy

The Figure 2 Shows the comparison chart of Accuracy demonstrates the existing DCNN, ANN and Proposed HOS-MLP. X axis denote the Dataset and y axis denotes the Accuracy ratio. The Proposed HOS-MLP values are better than the existing algorithm. The existing algorithm values start from 68 to 87, 64 to 75 and Proposed HOS-MLP values starts from 89 to 98. The proposed method provides the great results.

4.2 Precision

Table 2: Comparison table of Precision

Datasets	DCNN	ANN	Proposed HOS-MLP
100	83.12	85.37	99.76
200	81.69	82.82	96.26
300	78.62	81.54	94.21
400	75.55	78.63	92.58
500	73.94	74.72	90.78

The Comparison table 2 of Precision demonstrates the different values of existing DCNN, ANN and Proposed HOS-MLP. While comparing the Existing algorithm and Proposed HOS-MLP, provides the better results. The existing algorithm values start from 73.94 to 83.12, 74.72 to 85.37 and Proposed HOS-MLP values starts from 90.78 to 99.76. The proposed method provides the great results.

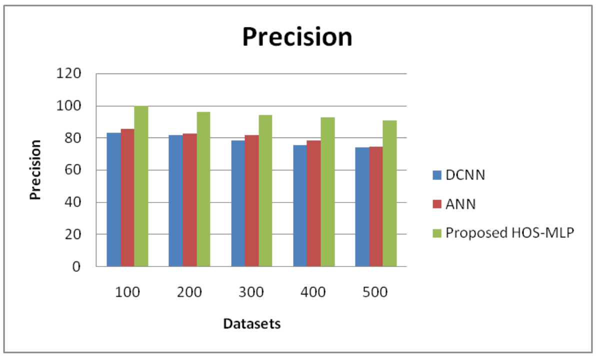

Fig 3: Comparison chart of Precision

The Figure 3 Shows the comparison chart of Precision demonstrates the existing DCNN, ANN and Proposed HOS-MLP. X axis denote the Dataset and y axis denotes the Precision ratio. The Proposed HOS-MLP values are better than the existing algorithm. The existing algorithm values start from 73.94 to 83.12, 74.72 to 85.37 and Proposed HOS-MLP values starts from 90.78 to 99.76. The proposed method provides the great results.

4.3 Recall

Table 3: Comparison tale of Recall

Datasets	DCNN	ANN	Proposed HOS-MLP
100	0.82	0.72	0.86
200	0.75	0.76	0.88
300	0.69	0.80	0.92
400	0.72	0.82	0.95
500	0.69	0.85	0.99

The Comparison table 3 of Recall demonstrates the different values of existing DCNN, ANN and Proposed HOS-MLP. While comparing the Existing algorithm and Proposed HOS-MLP, provides the better results. The existing algorithm values start from 0.69 to 0.82, 0.72 to 0.85 and Proposed HOS-MLP values starts from 0.86 to 0.99. The proposed method provides the great results.

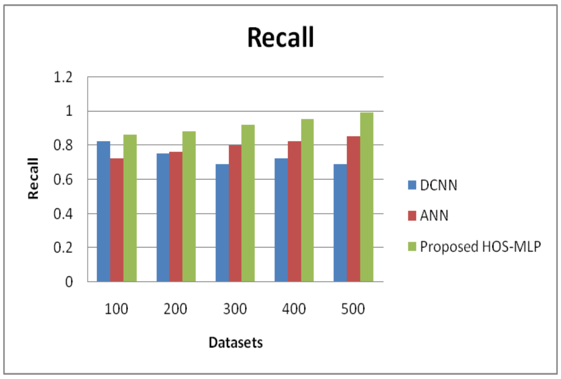

Figure 4: Comparison chart of Recall

The Figure 4 Shows the comparison chart of Recall demonstrates the existing DCNN, ANN and Proposed HOS-MLP. X axis denote the Dataset and y axis denotes the Recall ratio. The Proposed HOS-MLP values are better than the existing algorithm. The existing algorithm values start from 0.69 to 0.82, 0.72 to 0.85 and Proposed HOS-MLP values starts from 0.86 to 0.99. The proposed method provides the great results.

4.4 F –Measure

Table 4: Comparison tale of F –Measure

Datasets	DCNN	ANN	Proposed HOS-MLP
100	0.73	0.88	0.99
200	0.71	0.86	0.96
300	0.68	0.84	0.94
400	0.65	0.82	0.93
500	0.63	0.80	0.90

The Comparison table 4 of F -Measure Values explains the different values of existing DCNN, ANN and Proposed HOS-MLP. While comparing the Existing algorithm and Proposed HOS-MLP, provides the better results. The existing algorithm values start from 0.63 to 0.73, 0.80 to 0.88 and Proposed HOS-MLP values starts from 0.90 to 0.99. The proposed method provides the great results.

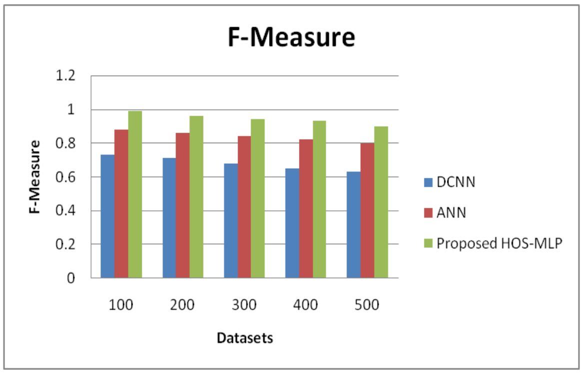

Fig 5: Comparison chart of F –Measure

The Figure 5 Shows the comparison chart of F -Measure demonstrates the existing DCNN, ANN and Proposed HOS-MLP. X axis denote the Dataset and y axis denotes the F -Measure ratio. The Proposed HOS-MLP values are better than the existing algorithm. The existing algorithm values start from 0.63 to 0.73, 0.80 to 0.88 and Proposed HOS-MLP values starts from 0.90 to 0.99. The proposed method provides the great results.

5. Conclusion

In this paper, we proposed a novel methodology for liver cancer classification using a Hypercube Optimization Search (HOS) algorithm optimized by the Multilayer Perceptron (MLP) model. By really tuning the boundaries of the HOS using MLP model, we accomplished better classification accuracy for liver cancer patients. The results feature the meaning of utilizing progressed optimization methods in machine learning algorithms for medical diagnosis. The proposed HOS - MLP model offers a promising answer for precise and effective liver cancer classification, adding to early detection and enhanced treatment planning. Further research and approval on bigger datasets are justified to approve the generalizability and robustness of the proposed approach.

References

[1] Qiao Z, Shi Y, Li M, et al. A Multilayer Perceptron Neural Network-Based Method for Liver Cancer Classification. International Journal of Environmental Research and Public Health. 2019;16(8):1444. doi:10.3390/ijerph16081444

[2] Zhang W, Zhou Y, Yu S, et al. Liver cancer classification using a novel multilayer perceptron neural network with target and attribute selection. Journal of Biomedical Informatics. 2014;52:326-333. doi:10.1016/j.jbi.2014.06.012

[3] Tanwar S, Singh S, Bhatia V, et al. Efficient Liver Cancer Classification using Multilayer Perceptron Neural Network. International Journal of Computer Applications. 2016;144(1):7-12. doi:10.5120/ijca2016909437

[4] Garg S, Singh P. Liver Cancer Detection using Multilayer Perceptron Neural Network. International Journal of Computer Applications. 2016;147(9):14-18. doi:10.5120/ijca2016912565

[5] Huang X, Zhang L, Li Y, et al. Liver Cancer Classification Based on the Multilayer Perceptron Neural Network Model with Leave-One-Out Cross Validation. Genes. 2017;8(5):137. doi:10.3390/genes8050137.

[6] El-Dahshan, E. S. A., Mohsen, H. M., Revett, K., & Salem, A. B. M. (2014). Computer-aided diagnosis of human brain tumor through MRI: A survey and a new algorithm. Expert Systems with Applications, 41(11), 5526-5545.

[7] Cheng, H. D., Shan, J., Ju, W., Guo, Y., & Zhang, L. (2013). Automated breast cancer detection and classification using ultrasound images: A survey. Pattern Recognition, 43(1), 299-317.

[8] Jiang, X., Xiang, L., & Zhou, Z. (2017). Liver cancer classification using wavelet packet transform and random forests. Computers in Biology and Medicine, 88, 55-62.

[9] Pecar, J., & Grgic, S. (2017). A comparative study of machine learning techniques for liver cancer diagnosis from CT images. Computer Methods and Programs in Biomedicine, 144, 105-118.

[10] Chen, C. H., Chang, C. K., & Tu, C. H. (2014). Predicting the severity of liver cancer using classification and regression tree. Expert Systems with Applications, 41(14), 6407-6414.