IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

IEEE SYSTEMS, MAN AND CYBERNETICS SOCIETY SECTION

Initial submission 19 August 2022, final submission received 1 August 2023, accepted 17 August 2023, date of publication 28 August 2023, date of current version 7 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3309557

RESEARCH ARTICLE

Genetically Optimized UFLANN for Uncovering Clusters

HIMANSHU DUTTA[1], SAURABH BILGAIYAN[1], BHABANI SHANKAR PRASAD MISHRA[1], SATCHIDANANDA DEHURI[2], (Member, IEEE), AND ASHISH GHOSH[3], (Senior Member, IEEE)

[1]School of Computer Engineering, KIIT Deemed to be University, Bhubaneswar 751024, India
[2]Department of Information and Communication Technology, Fakir Mohan University, Balasore 756019, India
[3]Machine Intelligence Unit, Indian Statistical Institute, Kolkata 700108, India

Corresponding author: Satchidananda Dehuri (satchi.lapa@gmail.com)

The work of Satchidananda Dehuri was supported in part by the Science and Engineering Research Board (SERB) under the Teachers' Associateship for Research Excellence (TARE) Fellowship; and in part by SERB, Government of India, through TARE Fellowship, (2021–2024), under Grant TAR/2021/000065.

ABSTRACT In this work, we present a novel clustering approach which is inheriting the best characteristics of Unsupervised Functional Link Artificial Neural Network (UFLANN) and Genetic Algorithms (GAs) for uncovering clusters embedded in dataset represented through $(X)_{Nxd}$, where X consists of N data points with d-dimensions. With an aim to realize natural clusters in a linear space UFLANN mapped the input vectors from lower dimension to higher dimension with a greater hope to achieve linearity in higher dimensional space. As a result, UFLANN introduces the problem of curse of dimensionality in the given datasets. However, it has been realized that the problems like sparse data and distance concentration associated with curse of dimensionality cast this problem to again a very complex problem. Hence to address some of the issues of curse of dimensionality, we have used GAs for selecting optimal number of features in the higher dimension for UFLANN to discover clusters embedded in the dataset. The proposed approach herein after named as GAUFLANN has been experimentally evaluated by using the metrics like (i) Davies-Bouldin Index, ii) Silhouette Score, and iii) Completeness score on different synthetic and real datasets. Our experimental study confirms that GAUFLANN is evidently scoring better in DB-index, Silhouette score, and Completeness score than the clustering methods like K-means, Hierarchical-Agglomerative (Average Linkage), and UFLANN across the datasets like Circles, Moons, Iris, and CORD-19.

INDEX TERMS Clustering, UFLANN, genetic algorithms, cluster validity, feature selection, SOFM, FLANN.

I. INTRODUCTION

Clustering is one of the fundamental problems in the area of pattern recognition, machine learning, and big data analysis and has been a center of focus of researchers of the said areas [1]. The reason is that it is unsupervised in nature i.e., its structural characteristics are unknown unless some domain knowledge in advance exists [2]. The objective of clustering is to determine a finite set of clusters to describe a dataset by maximizing the homogeneity inside a cluster and minimizing the heterogeneity between the clusters. In other words, samples belonging to a cluster are very similar to each other than samples that belong to different clusters. The distance metrics are used for measuring the degree of similarity among data points in such a way that more dissimilar data points have lower similarity values. There are many similarity metrices that are used in data clustering depending on their suitability in different areas of analysis [3]. In addition to its special importance in pattern recognition and machine learning, it is also a center of attraction in big data analysis. In big data [4], the 5 V's like volume, value, variety, velocity, and veracity makes the classical algorithms of data clustering unsuitable to use directly. Therefore, this effort has been made with a hope to cope up with some of the issues of big data while uncovering clusters embedded thereon.

The associate editor coordinating the review of this manuscript and approving it for publication was Xiaojie Su.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

Further, clustering problem can also be known as a member of NP-hard group [5]. This has created the scope for developing approximation algorithms including not only the simple heuristic functions but also the use of general-purpose nature inspired algorithms. Further, examining whether or not the underlying data has the clustering tendency and contains natural clusters are two vital activities of clustering task. Over decades many algorithms have been developed to address the said issues [6]. Sinaga and Yang [7] proposed a novel unsupervised k-means (U-k-means) clustering algorithm which is automatically discovering the optimal number of clusters without giving attention on initialization and parameter selection. Liu et. al. [8] developed an improvement over non-dominated sorting genetic algorithm III using genetic k-means clustering to achieve better convergence. Further, Wang et. al. [9] proposed a k-means clustering approach which handles incomplete data intrinsically. Maulik and Bandyopadhyay [10], developed a GA-clustering method to search for the cluster centers which minimizes the clustering metric. They found that floating point representation was faster, consistent and provided a higher degree of precision. Maheshwar et al. [11] in 2015 proposed a firefly based genetic algorithm for data clustering, where the initial population is selected from a pool of population on the basis of firefly algorithms. Nguyen and Kuo [12] proposed a two stage method named partition and merge based fuzzy genetic clustering algorithm for categorical data clustering. Shinozaki [13] introduced unsupervised competitive learning, which only requires forward propagating signals for CNNs. They evaluated the proposed method on image discrimination tasks using the MNIST, CIFAR-10 and Image Net datasets, and they achieved better result in comparison to other biologically motivated algorithms. Merikhi and Soleymani [14] presents a novel framework for automatic data clustering using nature inspired optimization algorithms. Maronna et al. [15] provide a detailed description of various approaches to clustering along with their applications. Fahad et al. [16] and Mahdi et al. [17] present detailed survey of application of clustering algorithms in Big Data. Further, Bruse et al. [18] present a use case of clustering algorithms in the field of medical research. Ezugwu et al. [19] presents a comprehensive analysis and survey of state of the art clustering algorithms.

Recent research activities in clustering have recognized neural networks are a promising alternative to different conventional clustering algorithms [2]. Artificial Neural Networks(ANNs) are capable of generating complex mapping between the input and the output space, and hence these networks can form randomly complex nonlinear decision boundaries. In [6], authors have shown a direction that FLANN shall not be restricted its scope to function approximation and classification only it can also be extended for pattern clustering like self organizing map [2] to discover clusters hidden in datasets. However, the architectural complexity of UFLANN proposed in [6] is directly proportional to the number of features and the functions in hand for expansion of the given feature values. For reducing the architectural complexity, like UFLANN we map given input vectors to higher dimensional space and then unlike UFLANN we uncover hidden clusters by considering relevant features of higher dimensional space through hybridization of GA and UFLANN. The steps from selection to learning are accomplished by hybridization of UFLANN with GAs. The rest of the paper is organized as follows. In Section II, we have discussed background materials. Section III provides our GAUFLANN for revealing clusters. In Section IV we have presented the experimental studies and a comparative performance analysis over the datasets like Iris, Moon, Circles, and CORD-19. Section V presents the conclusions and future work.

II. BACKGROUND

In this section, we discuss the various concepts on which our work builds upon. The section is divided into two subsections, namely, the basic working principle of canonical Genetic Algorithms, followed by the description of the task of feature selection, and a brief description of UFLANN.

A. CANNONICAL GENETIC ALGORITHM AND FEATURE SELECTION

Genetic Algorithms (GAs) [20] are a class of meta-heuristic algorithms based on evolutionary principles such as natural selection, genetic mutation, etc. For a d-dimensional optimization problem, the algorithm initializes a set of solutions, termed as population. Each of the solution is a d-dimensional vector, which is sampled from a distribution function and scaled to fit the problem boundary. The population then goes through a process of evolution, in each iteration of which, the fitness of each solution is evaluated. Based on the fitness, pairs of individuals are selected to reproduce. Reproduction can take effect in multiple ways. One such method of reproduction is, binary crossover, in which, half of each parent solution vector is combined to form the offspring vector. Then the newly generated offspring vectors are mutated using some mutation strategy, for example, single cell mutation, in which a single value in the vector is replaced with a different value from the solution boundary. The generated offspring vectors are then added to the original population, and the individuals with least fitness value are eliminated. Over time this results in a better quality population, which in turn results in an optimized solution. The flow of canonical genetic algorithm has been depicted in Figure 1. The algorithm can run for a predetermined number of iterations, or till a certain criterion is met. GA has been widely adapted for different applications, such as image processing, data mining, computer games, graph optimization, network optimization, etc, [21], [22], [23], [24], [25]. One such application of interest to this work is feature selection.

Feature selection in the context of unsupervised approaches, is the process of finding a subset of features which contribute the most to the task of uncovering the

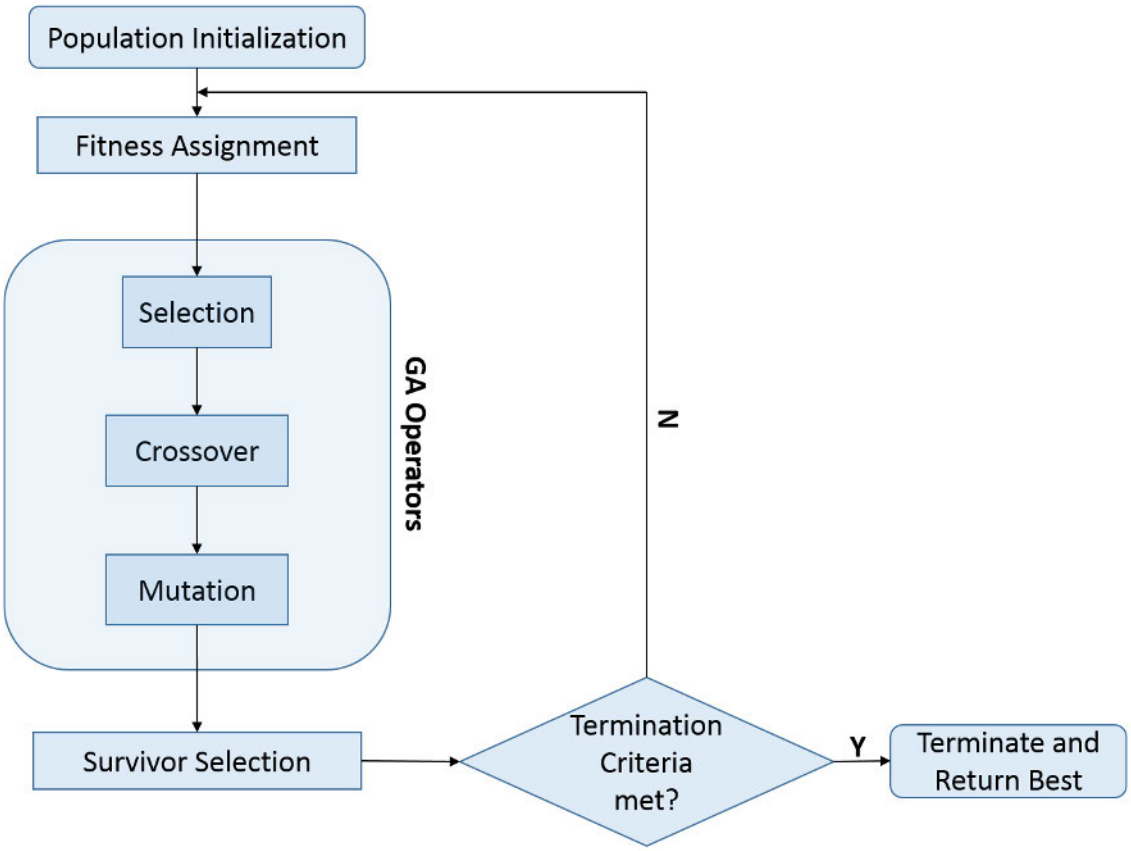

FIGURE 1. Flow of cannonical genetic algorithm.

clusters of data points. Based on the methodology used, there can be different classes for feature selection methods. For example, filter based methods consider various univariate statistics of individual features in order to eliminate the insignificant features. Some of the popular metrices includes, Correlation (as defined in (1) where X and Y are two of the features of the dataset), Entropy, Fischer Score, etc.

$$\rho_{X,Y} = \frac{cov(X, Y)}{\sigma_X \sigma_Y} \tag{1}$$

Generally, the contribution of features is evaluated as a subset of the feature set, rather than evaluating individual features. Another class of methods used for feature selection are called as wrapper-based methods. These methods consist of an algorithm to search the space of all possible subsets of features. Each subset is evaluated based on a certain criterion, for example, a dataset being clustered can be evaluated based on metrics like Completeness Score, Davies-Bouldin index, etc. Wrapper based methods are generally expensive since they require exhaustive search of the feature space, but they can be optimised by using a stochastic algorithm in place of exhaustive algorithms. The general flow of feature selection has been summarized in (2),(3), and (4). Consider a dataset X which consists of d features, when the dataset is passed through a feature selection method, FS, it outputs a subset of features which are most relevant to the specified task. If a feature selection algorithm explores all the possible feature subsets, it would result in an exponential complexity, i.e., $O(2^d)$.

$$X =< x_1, x_2, x_3, \ldots, x_n > \tag{2}$$

$$FS(X) =< x_2, x_5, x_9, \ldots > \tag{3}$$

$$|P(X)| = 2^d \tag{4}$$

Genetic Algorithm and its variants have been well utilized for the purpose of feature selection and optimization in the recent years. GA based feature selection relies on a well-defined metric, such as accuracy, precision, or model loss. In case of supervised algorithms used for regression and classification tasks, it is easier to define such metrics as opposed to unsupervised algorithms, used for clustering and knowledge discovery.

B. UNSUPERVISED FUNCTIONAL LINK ARTIFICIAL NEURAL NETWORK

Unsupervised Functional Link Artificial Neural Network combines the power of Functional Link Artificial Neural Networks and Self Organizing Feature Maps. FLANN maps the input feature set to a higher dimension feature space, along with introducing non-linearity to the input features.

It consists of a single layer of neurons, where each neuron is a polynomial function, which expands the feature space to increase the variance and facilitates faster convergence.

Self Organizing Feature Maps is a competitive-learning based algorithm, which, for a given set of input data points, generates a learned representation in the form of a 2D lattice, or map. The generated map is a latent representation of the input data points. The representation along with preserving the input features, also preserves their topology in an d-dimensional vector space, where d is the number of input features, and hence encompasses the information regarding the neighborhood of each data point. This information can then be used to form clusters in the latent vector space. SOFMs have been widely used in various applications, such as, [26] and [27].

UFLANN utilizes both these methodologies in a two-stage algorithm, which at first maps the input features to a higher dimensional space, and hence increases the variability and introduces non-linearity, then uses the feature vectors thus obtained to generate the feature map using SOFM. The map is then used to reveal clusters from the data points in the latent vector space.

Although UFLANN algorithm results in well formed clusters for a given input dataset, it is compute intensive and requires proper configuration to achieve optimal results. The different features generated using FLANN, aren't always relevant to the task of clustering, some of which rather hamper the performance of the algorithm. The choice of proper polynomial functions is also critical to the performance of the UFLANN algorithm. One other important aspect of any clustering algorithm, is to discover the optimal number of clusters, which again is a compute intensive task.

III. GENETICALLY OPTIMIZED UNSUPERVISED FUNCTIONAL LINK ARTIFICIAL NEURAL NETWORKS (GAUFLANN)

In this section, we first formally define the problem of feature selection to reduce the curse of dimensionality introduced by the use of UFLANN, and then describe the solution, and the algorithm GAUFLANN, addressing the various aspects of the problem.

A. PROBLEM FORMULATION

Although UFLANN results in well-formed clusters for a given dataset, it is a computationally intensive algorithm. Moreover, it requires quantum of effort to fine tune large number of weight values to find the optimal configuration. As a result of expansion of input features using FLANN, the feature set gets more complex by inviting problems like sparse data and distance concentration, as it is projected to a higher dimensional feature space with an expectation of linearity between clusters in transformed space. FLANN consists of multiple polynomial functions, which take the input feature vector and expand each feature using the functions. Consider the input feature vector X (5), which consists of two features x_1 and x_2.

$$X =< x_1, x_2 > \tag{5}$$

Also consider that the UFLANN network is using the following polynomial functions: Chebyshev, Legendre, and Power function. Each of the functions will expand every feature till the third polynomial as shown in (6).

$$\begin{aligned} X_{out} =< & Ch0(x_1), Ch1(x_1), Ch2(x_1), \\ & Ch0(x_2), Ch1(x_2), Ch2(x_2), \\ & L0(x_1), L1(x_1), L2(x_1), \\ & L0(x_2), L1(x_2), L2(x_2), \\ & P0(x_1), P1(x_1), P2(x_1), \\ & P0(x_2), P1(x_2), P2(x_2) > \end{aligned} \tag{6}$$

Hence the 2-dimensional vector is mapped to an 18-dimensional output vector. Real world datasets comprise of many more features, as opposed to just two, which would result in even larger dimensionality of the output vector.

The features of a dataset are not always linearly separable, and hence for clustering algorithms such as K-means, or Hierarchical clustering, it becomes difficult to uncover the clusters present in such datasets. By the use of FLANN, UFLANN address non-linearity in the given dataset on the higher dimensional space effectively, as well as manages to discover compound features with the help of different polynomial functions. Although FLANN maps the input feature vector to a higher dimensional feature vector, not all the generated features contribute to projection of input data points to the latent space, or cluster discovery. The redundant features in the feature vector, at the very least produce noise in the generated feature map, resulting in improperly formed clusters, and increased algorithmic complexity.

Further for a feature map with well-formed clusters, it is crucial to determine the number of clusters to form along with the topology and size of the generated feature map, as running parameters for the UFLANN algorithm. This requires multiple iterations of the algorithm to be run to find the optimal number of clusters. UFLANN in most real-world scenarios becomes infeasible to run, or results in sub-optimal clustering performance. In the following section we propose a framework to reduce the overall complexity of the UFLANN algorithm, and at the same time improve its performance.

B. PROPOSED SOLUTION

For any dataset clusters are patterns that originate from the high local density of data points which are separated from each other by low density regions. As proposed in UFLANN, there are two phases which result in revelation of these clusters. The first phase is the mapping of the input vectors to a higher dimension feature space. This is accomplished using various polynomial functions. Mishra et al. [6] made use of three recurrent polynomial function, namely Chebyshev (7), (8) and (9); Legendre (10), (11) and (12); and Power

series (13) and (14).

$$Ch_0(X) = 1 \tag{7}$$
$$Ch_1(X) = x \tag{8}$$
$$Ch_{n+1}(X) = 2 * x * Ch_n(X) - Ch_{n-1}(X), n >= 1 \tag{9}$$
$$L_0(X) = 1 \tag{10}$$
$$L_1(X) = x \tag{11}$$
$$L_{n+1}(X) = \frac{(2n-1) * x * L_{n-1} - (n-1) * L_{n-2}}{n} \tag{12}$$
$$P_0(X) = 1 \tag{13}$$
$$P_{n+1}(X) = x * P_n(X) \tag{14}$$

Although these functions manage to map individual features to higher dimension feature space, but they don't make use of the relationship between the features. It has been well observed that the interaction between the input features can result in generation of new features. The same idea is also used in the case of data imputation, by leveraging the more prominent features to impute the missing features. Hence we propose, in addition to using polynomial functions which map input features individually to higher dimensional feature space, to also make use of functions, that take into account multiple features at once. In this work, we have chosen to use Three Hump Camel (15) and Beale functions (16).

$$f(x_1, x_2) = 2 * x_1^2 - 1.05 * x_1^4 + \frac{x_1^6}{6} + x1 * x2 + x_2^2 \tag{15}$$
$$f(x_1, x_2) = (1.5 - x_1 + x_1 * x_2)^2 + (2.25 - x_1 + x_1 * x_2^2)^2 + (2.625 - x_1 + x_1 * x_2^3)^2 \tag{16}$$

Both the functions allow for the interaction between the different features, while also introducing non-linearity in the expanded feature space. Although the functions are used with only two input parameters, it can be extrapolated to take include more features. With N/2 features as input for a dataset with N input features yielding, (N|(N/2)) expanded features in the higher dimensional feature space.

The second phase of UFLANN makes use of SOFMs, to find the clusters formed by the various data points. The algorithm first initializes the weight vectors using the PCA of the expanded input features, then in each iteration it performs the following operations:

1) Find the distance between each data point and weight vectors, and select the output neuron with highest similarity (where similarity is defined as the distance between the data point and the weight vector) as the winner.
2) Calculate the neighbourhood impact on the topology, around the winner neuron, using a neighbourhood function. The Gaussian function is used as the neighbourhood function (17),

$$\rho(w_c, i) = e^{-\frac{\|x_i - x_c\|^2}{2*\sigma^2}} \tag{17}$$

where σ denotes the monotonically decreasing radius hyperparameter of the algorithm, and x_i and x_c denote the position of the i^{th} weight vector and the winner weight vector respectively.
3) Update each of the weight vectors in the topology using the weight update formula (18),

$$w_{new} = w_{old} + \Delta w * \rho(w_c, i) \tag{18}$$

where, w_{new}, w_{old}, and Δw, denote the new weight vector, old weight vector and shift in the weight vector respectively.

At the end of the iterations the weight vectors thus obtained, form the feature topology map of the dataset, each weight vector representing a single cluster center. The UFLANN algorithm has been summarized in Algorithm 1.

Algorithm 1 UFLANN

```
Input  : DATASET[ ], max_learning_rate, max_sigma
Output: weights[ ]
map_size ← Initialize map_size to (no. of weight
vector × size of weight vector)
weights[ ] ← Initialize weights as a vector of map_size
randomly by sampling normal distribution
for step ← 1 to num_iterations do
    learning_rate ←
    (1 − step/num_iterations) × max_learning_rate
    sigma ← (1 − step/num_iterations) × max_sigma
    for vec ∈ DATASET[ ] do
        vec_expanded ← FLANN-EXPANSION(vec)
        weight_idx ← Find weight vector with
        minimum distance fron vec_expanded
        update_radius ←
        NEIGHBOURHOOD-FUNC(weights[ ],
        weight_idx, sigma)
        weights[ ] ←
        WEIGHT-UPDATE-FUNC(weights[ ],
        update_radius, vec_expanded, learning_rate)
    end for
end for
```

We propose a third phase in addition to the two proposed by Mishra et al. [6], as the optimization phase. This phase exists before the other phases, and makes use of meta-heuristic algorithms as a modular addition to further optimize the results as well as computational cost of UFLANN algorithm. In this work, we have chosen Genetic Algorithm as the choice of optimization algorithm. The proposed algorithm optimises three parameter of the UFLANN algorithm:

1) **Input feature**: Each input feature after FLANN expansion, can either be allowed to be used as a feature by the SOFM phase or masked. This can be indicated using a vector consisting of 1s and 0s, where 1 represents an allowed feature, and 0 represents a masked feature.
2) **Map Size**: The output of the SOFM phase is the feature map, which maps the expanded input vectors to a latent representation, each weight vector of which represents

a cluster. Finding the optimal map size is necessary in order to properly find the clusters in the input space.

3) **Number of Clusters**: Each input vector is mapped to one of the specified cluster, but it takes multiple iterations to find the optimal number of clusters for a set of input vectors, and randomly performing this task can take even longer.

Algorithm 2 GAUFLANN

Input : UFLANN-FITNESS-FUNC, DATASET[ ]
Output: Best Fitness Individual
population_size ← (num_individuals, (num_features + map_dimensions + 1))
population[ ] ← Initialise a population randomly from normal distribution
fitness[ ] ← Initialize an empty array of size (num_individuals, 1)
for *step* ← 1 **to** *num_iterations* **do**
  for *individual* ∈ *population*[ ] **do**
    fitness[*individual.index*] ← UFLANN-FITNESS-FUNC(*individual*, DATASET[ ])
  end for
  for $i \leftarrow 1$ **to** $\frac{num\_individuals}{2}$ **do**
    parent1, parent2 ← SELECTION-FUNC(population[ ], fitness[ ])
    child ← CROSSOVER-FUNC(parent1, parent2)
    child ← MUTATION-FUNC(child)
    Insert the child in the population
  end for
end for

Algorithm 2 summarises the process of GAUFLANN. The algorithm first initialises a population of individuals, where each individual consists of randomly initialized vectors for the three parameters. For the expanded input features, a random vector of size, num_features is initialised, with 0s and 1s, as discussed above, for the map size, two random integers (H, W) in a specified range are generated describing the height and width of the map respectively, and for the number of clusters, a single value is initialized at random in a specified range. GAUFLANN uses, UFLANN as a modular fitness function, which takes input the dataset, and returns a fitness score depending on the task. In this work, we use Davies-Bouldin index (DBI) as the fitness metric. The reason for choosing Davies-Bouldin index as the fitness metric is twofold. Firstly, it is defined as a function of only the cluster labels of the data points, and hence both, label and unlabelled datasets can be optimized and evaluated, and second, it examines the cluster quality based on the coherence between the data points, which is the same philosophy followed by UFLANN algorithm. At each step of the algorithm, the fitness of each individual in the population is evaluated on the UFLANN-FITNESS-FUNC. After the fitness of all the individuals is evaluated, for num_individuals / 2 iterations two individuals from the population are selected as parents, using a selection strategy, and are mated using a crossover strategy to generate a new offspring. The generated offspring is then mutated using a mutation strategy. In this work, we use k-way tournament selection, two point crossover, and multi-point mutation as the respective strategies for selection, crossover and mutation. The algorithm returns the optimal individual from the population. The individual consists of the feature mask, the map dimensions and the optimal number of clusters.

GAUFLANN, by optimizing the feature mask ensures that only the features which have relatively higher contribution towards the unravelling of the clusters in the SOFM map are allowed. The reduced number of features, also results in the reduction in the dimensionality of weight vectors initialized in the ULFANN process, which significantly reduces the computational complexity. Further, by optimizing the map size for the SOFM phase, GAUFLANN ensures a proper topology, which results in better cluster discovery. Since datasets can be distributed in varying ways, initializing a proper topology ensures that the neighbourhood surrounding each weight vector is optimal for the competition with the specific weight vector for each input vector. Finally, the UFLANN algorithm results in a better cluster discovery, when proper number of clusters are specified.

IV. EXPERIMENT AND RESULTS

In this section, we discuss the experimental setup for evaluation of GAUFLANN. In order to evaluate the performance of the proposed approach, we perform numerous experiments on multiple synthetic and real world datasets, and evaluate the performance of the proposed approach in contrast with different well-established methods, on metrices, including Silhouette Score and Completeness Score.

A. EXPERIMENTAL SETUP

For the extensive evaluation of the proposed approach, multiple real world and synthetic datasets are used. The synthetic datasets which have been considered for comparison are: Circles and Moons. Both of these are standard binary datasets used for evaluation and visualization of clustering and classification algorithms. For the purpose of conducting experiments, we have considered 1000 points in each dataset. As can be seen in Figures 2 and 3, both the dataset consist of interleaving data points, which makes it difficult for simple distance based clustering algorithms to properly categorise the data points into the right clusters. Further two real world datasets have also been used. Iris [28] dataset consists of four features and three target classes. Distribution shown in Figure 4. For evaluation of the proposed approach in a real world use-case, we consider CORD-19 dataset [29], which consists of 500,000 scholarly articles relating to COVID-19, consisting of 200,000 full length texts. The dataset can be used to extract information of various kinds, but for the sake of comparison we focus on clustering the articles based on the full text content and title, because of which the dataset gets restricted to only 200,000 data points. We preprocess

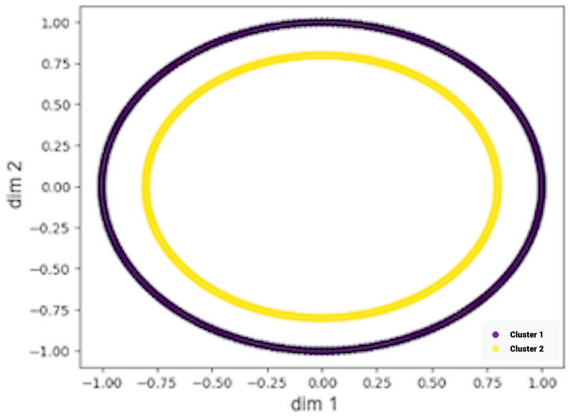

FIGURE 2. Circles dataset.

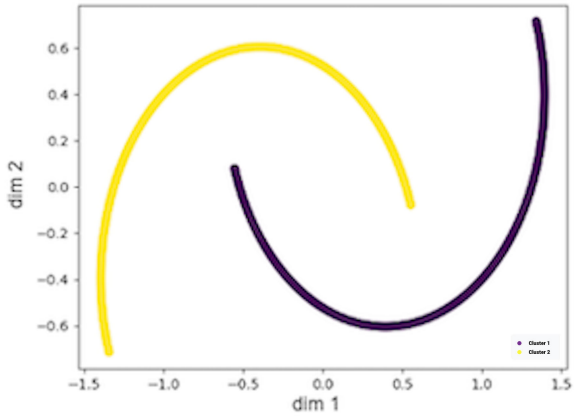

FIGURE 3. Moons dataset.

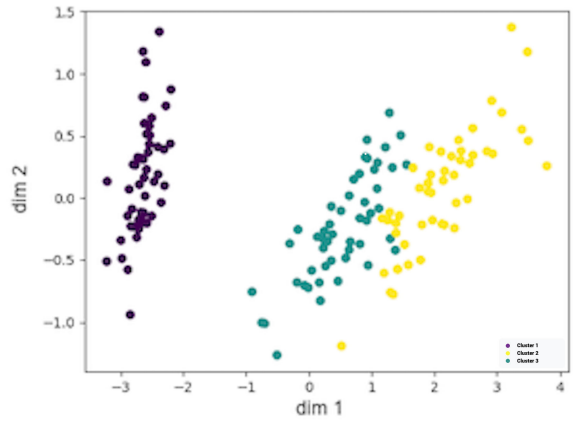

FIGURE 4. Iris dataset.

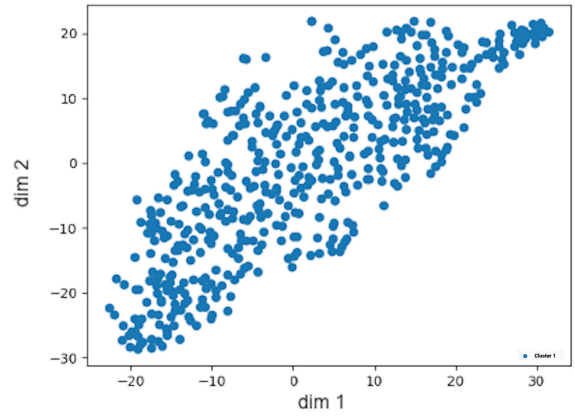

FIGURE 5. CORD-19 dataset.

the dataset and extract the feature vectors by tokenizing and embedding extraction for each data point individually. The dataset thus obtained consists of 200,000 data points each consisting of 30 features. Distribution shown in Figure 5. We consider only 30 features considering the feature expansion by FLANN.

The datasets are evaluated on the basis of three commonly used clustering metrics. Firstly we use Davis-Bouldin Score (DBScore) [30], which we also use as the optimisation metric for GAUFLANN. DBScore is an internal metric, as it only uses the cluster labels to evaluate the cluster quality. The minimum DBScore value is 0. A DBScore towards 0 indicates higher coherence between the data points of same cluster, while higher DBScore indicates poor cluster coherence. Secondly we use Silhouette Coefficient (19), which ranges from -1 to 1. A Silhouette score of 1 indicates that the clusters are well apart and can be clearly distinguished from each other, 0 signifies that the clusters are indifferent, or the distance between the clusters is not significant, hence lower Silhouette score indicates improper cluster assignment. Lastly, completeness score (20) has been used, which is an external metric, and requires the assigned cluster labels as well as true cluster labels in order to assess the proper assignment of clusters in contrast to the true labels. A Completeness score of 1 represents perfectly labelled predicted. Since the clusters are assigned to the data points arbitrarily, a direct comparison between the true labels and the predicted cluster becomes infeasible, and hence metrices such as accuracy score cannot be used to evaluate the performance of the clustering algorithm. Using Completeness score provides with a similar insight as by accuracy score.

$$S_c = \frac{(b-a)}{max(a,b)} \tag{19}$$

$$C = 1 - \frac{H(y_{pred}|y_{true})}{H(y_{pred})} \tag{20}$$

To properly evaluate and establish proper benchmark, the proposed approach has been compared with commonly used clustering approaches, KMeans clustering (KM), Average Linkage Hierarchical Clustering (HC), and UFLANN itself. Each experiment has been repeated over 10 times, and the reported metrices are the average value of the runs. The plots shown are of the overall best performance over the different runs. All the experiments are conducted on an Apple M1, 16GB, and MacOS based system. The source code for the algorithms as well as experiments are written in Python 3.10.

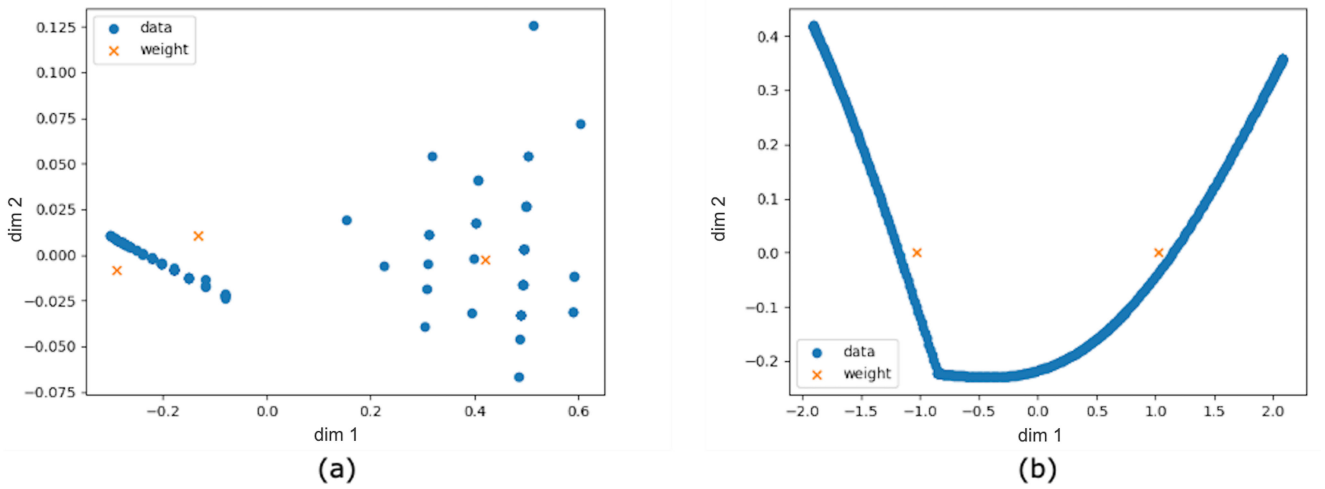

FIGURE 6. Weight vs data (a) Iris (b) Moon.

For KMeans, and Hierarchal Clustering, the Scikit-Learn package [31] is used as a standard.

B. EXPERIMENTAL RESULTS

To properly evaluate the performance of GAUFLANN against other standard approaches, we focus on mainly two observational results, along with the comparison based on the evaluation metrics themselves. To highlight the performance of GAUFLANN in terms of cluster centres, we first plot the Weight vs Data plot. UFLANN makes use of SOFM, which generates a condensed representation of the input vectors in the form of weight vectors, which are organized in a certain topology. Weight vs Data plot compares the distribution of the input features against the weight vectors, rather than the generated topology map, since the weight vectors act as the cluster centres when revealing the clusters in the actual data points. As can be seen in Figure 6, GAUFLANN produces the optimal number of clusters for both the Iris and Moons dataset. Since both the datasets are labelled, we can compare the number of clusters produced with the actual number of target classes in the datasets.

Finding the optimal number of clusters is essential as in UFLANN, it depends on the empirical knowledge about the dataset itself. GAUFLANN on the other hand, starts with a rough range of cluster count, and stochastically explores the search space to reach the optimal number of clusters based on the fitness metric. The modularity of the proposed approach allows for multiple clustering metrices to be factored in, using a weighted mean of the metrices as well. Further, Table 1 represents the performance of GAUFLANN against, K-means, Hierarchical Clustering, and UFLANN, on the basis of DBIndex, Silhouette Score, and Completeness score. For most of the experiments the proposed approach outperforms the compared algorithms. As can be observed, for all the different datasets GAUFLANN shows the best performance in terms of DBIndex, as a lower DBIndex indicates a better cluster coherence. Similarly, for all the datasets, GAUFLANN shows better Silhouette score compared to the other approaches. For the comparison of Completeness score, GAUFLANN shows a comparatively better performance as compared with other approaches for all the datasets, except Moons. For the Moons dataset, UFLANN shows slightly better performance overall.

TABLE 1. Comparision of different approaches.

Dataset	Approach	DBIndex	Silhouette Score	Completeness Score
Cirlces	GAUFLANN	0.38	0.97	0.89
	UFLANN	0.64	0.87	0.74
	K-means	0.72	0.45	0.42
	Hierarchical	0.66	0.31	0.65
Moons	GAUFLANN	0.15	0.96	0.88
	UFLANN	0.37	0.77	0.91
	K-means	0.87	0.92	0.51
	Hierarchical	0.54	0.76	0.46
Iris	GAUFLANN	0.07	0.80	0.79
	UFLANN	0.25	0.76	0.54
	K-means	0.66	0.55	0.46
	Hierarchical	0.19	0.28	0.49
CORD-19	GAUFLANN	0.11	0.85	-
	UFLANN	0.28	0.73	-
	K-means	0.31	0.23	-
	Hierarchical	0.82	0.85	-

The results thus presented prove that GAUFLANN results in a better optimization when compared with UFLANN and overall, results in a more sound clustering of the data points. Further we compare the clustering ability of the different algorithms. For all the experiments, GAUFLANN shows the an overall better cluster formation for the data points. Since the datasets themselves are multi-dimensional, we perform PCA transform on the datasets, to map them to a 2-dimensional vector space, in order to better visualize the clusters. Figure 7, shows the formed clusters for the Iris dataset by GAUFLANN compared with UFLANN. Since the Iris dataset has the target labels, we can compare the formed clusters, indicated by the color of the data points, with

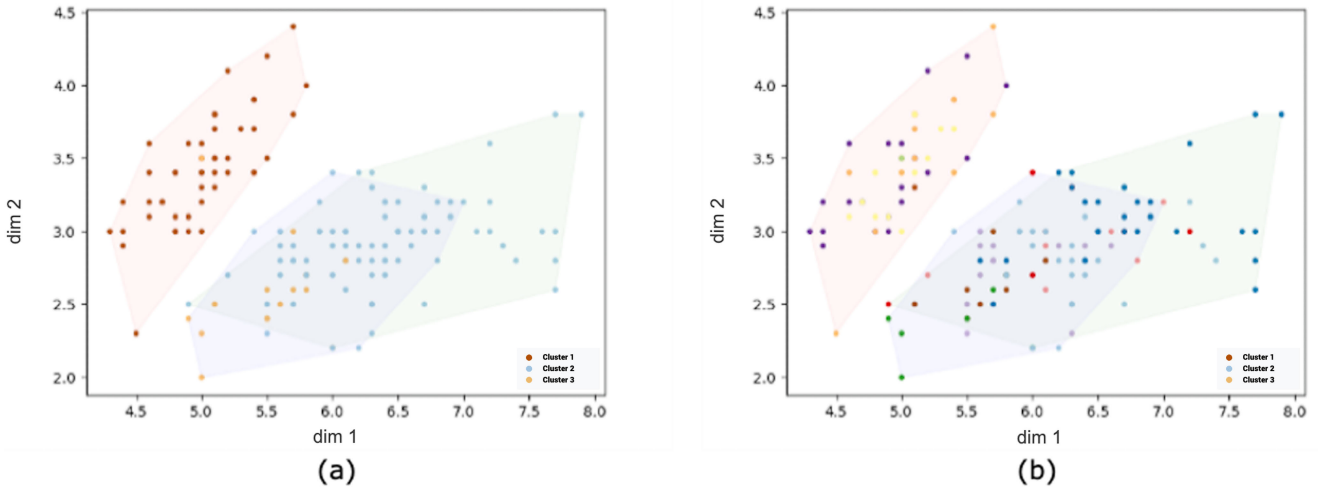

FIGURE 7. Clusters formation for Iris Dataset (a) GAUFLANN (b) UFLANN.

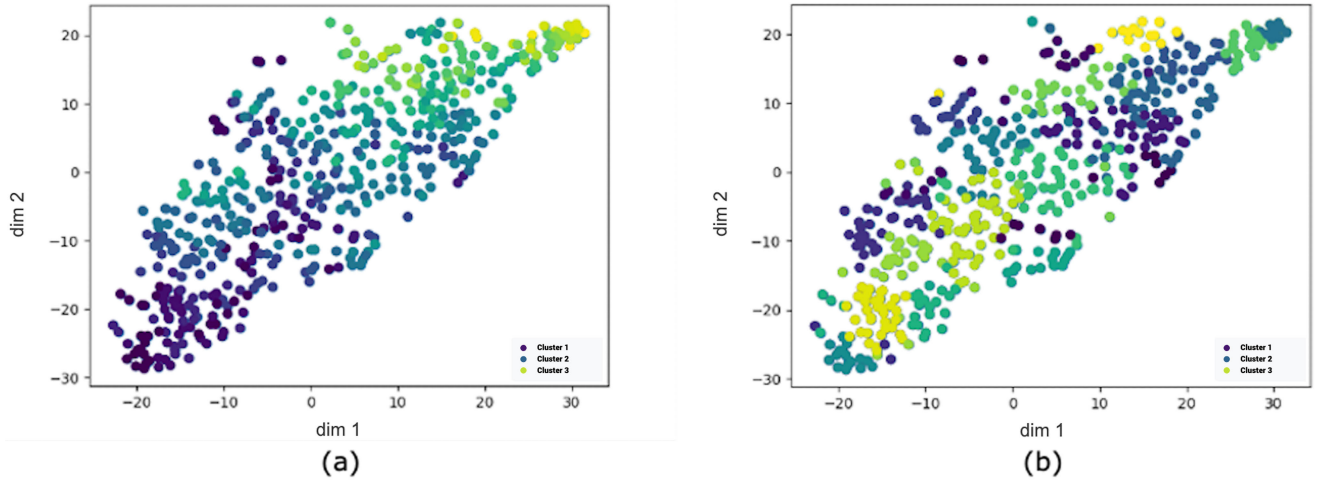

FIGURE 8. Clusters formation for CORD-19 Dataset (a) GAUFLANN (b) UFLANN.

the actual target labels, indicated by the bounding polygons. It can clearly be observed that the clusters formed my GAUFLANN show more similarity with the actual target labels.

Further, Figure 8, depicts the comparison between the clusters formed for the CORD-19 dataset, by UFLANN and GAUFLANN. Similar to the Iris dataset, GAUFLANN shows better cluster coherence for the CORD-19 dataset as well. The proposed approach uncovers four clusters in the dataset, while data points in each cluster evidently higher coherence within than with data points from other clusters.

Overall, GAUFLANN optimizes the performance over the regular UFLANN, by optimizing the number of clusters, as well as the selecting the optimal features generated using FLANN, along with optimizing the overall performance based on the various clustering metrices. The results show a superior performance overall, in terms of the clustering metrices, and the actual clusters formed using the various approaches on the different datasets.

V. CONCLUSION AND FUTURE WORK

The UFLANN algorithm proposed for the task of cluster extraction, introduces the problem termed as curse of dimensionality. The increase in dimensionality of the data, leads to problems like sparse data and distance concentration, along with performance penalties. This work proposed a novel approach, GAUFLANN, which utilizes the ability of Genetic Algorithms to select an optimal feature subset, resulting in better cluster formation, and makes use of UFLANN to discover the clusters themselves. We perform extensive comparative study of the proposed approach with well-known clustering algorithms, such as K-means, and Agglomerative Clustering (average linkage), along with comparison with

UFLANN as well. The proposed approach outperforms the other approaches when compared on the basis of DBIndex, Silhouette score and completeness Score. The approach is further tested on real-world CORD-19 dataset, showing better cluster extraction. Further work can include a comparative study of different optimization algorithms in place of canonical Genetic Algorithm, and evaluation of different expansion functions used in UFLANN.

REFERENCES

[1] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999.

[2] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, nos. 1–3, pp. 1–6, Nov. 1998.

[3] J. C. Bezdek and N. R. Pal, "Some new indexes of cluster validity," *IEEE Trans. Syst., Man Cybern., B*, vol. 28, no. 3, pp. 301–315, Jun. 1998.

[4] E. C.-K. Tsao, J. C. Bezdek, and N. R. Pal, "Fuzzy Kohonen clustering networks," *Pattern Recognit.*, vol. 27, no. 5, pp. 757–764, May 1994.

[5] E. R. Hruschka, R. J. Campello, and A. A. Freitas, "A survey of evolutionary algorithms for clustering," *IEEE Trans. Syst., Man, Cybern., C*, vol. 39, no. 2, pp. 133–155, Mar. 2009.

[6] B. S. P. Mishra, O. Pandey, S. Dehuri, and S.-B. Cho, "Unsupervised functional link artificial neural networks for cluster analysis," *IEEE Access*, vol. 8, pp. 169215–169228, 2020.

[7] K. P. Sinaga and M.-S. Yang, "Unsupervised K-means clustering algorithm," *IEEE Access*, vol. 8, pp. 80716–80727, 2020.

[8] Q. Liu, X. Liu, J. Wu, and Y. Li, "An improved NSGA-III algorithm using genetic K-means clustering algorithm," *IEEE Access*, vol. 7, pp. 185239–185249, 2019.

[9] S. Wang, M. Li, N. Hu, E. Zhu, J. Hu, X. Liu, and J. Yin, "K-means clustering with incomplete data," *IEEE Access*, vol. 7, pp. 69162–69171, 2019.

[10] U. Maulik and S. Bandyopadhyay, "Genetic algorithm-based clustering technique," *Pattern Recognit.*, vol. 33, no. 9, pp. 1455–1465, Sep. 2000.

[11] K. Kaushik and V. Arora, "A hybrid data clustering using firefly algorithm based improved genetic algorithm," *Proc. Comput. Sci.*, vol. 58, pp. 249–256, Jan. 2015.

[12] T. P. Q. Nguyen and R. J. Kuo, "Partition-and-merge based fuzzy genetic clustering algorithm for categorical data," *Appl. Soft Comput.*, vol. 75, pp. 254–264, Feb. 2019.

[13] T. Shinozaki, "Biologically motivated learning method for deep neural networks using hierarchical competitive learning," *Neural Netw.*, vol. 144, pp. 271–278, Dec. 2021.

[14] B. Merikhi and M. R. Soleymani, "Automatic data clustering framework using nature-inspired binary optimization algorithms," *IEEE Access*, vol. 9, pp. 93703–93722, 2021.

[15] R. Maronna, C. C. Aggarwal, and C. K. Reddy, "Data clustering: Algorithms and applications," *Stat. Papers*, vol. 57, no. 2, pp. 565–566, Jan. 2015.

[16] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras, "A survey of clustering algorithms for big data: Taxonomy and empirical analysis," *IEEE Trans. Emerg. Topics Comput.*, vol. 2, no. 3, pp. 267–279, Sep. 2014.

[17] M. A. Mahdi, K. M. Hosny, and I. Elhenawy, "Scalable clustering algorithms for big data: A review," *IEEE Access*, vol. 9, pp. 80015–80027, 2021.

[18] J. L. Bruse, M. A. Zuluaga, A. Khushnood, K. McLeod, H. N. Ntsinjana, T.-Y. Hsia, M. Sermesant, X. Pennec, A. M. Taylor, and S. Schievano, "Detecting clinically meaningful shape clusters in medical image data: Metrics analysis for hierarchical clustering applied to healthy and pathological aortic arches," *IEEE Trans. Biomed. Eng.*, vol. 64, no. 10, pp. 2373–2383, Oct. 2017.

[19] A. E. Ezugwu, A. M. Ikotun, O. O. Oyelade, L. Abualigah, J. O. Agushaka, C. I. Eke, and A. A. Akinyelu, "A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects," *Eng. Appl. Artif. Intell.*, vol. 110, Apr. 2022, Art. no. 104743.

[20] J. H. Holland, "Genetic algorithms," *Sci. Amer.*, vol. 267, no. 1, pp. 66–72, Jul. 1992.

[21] C. Han, L. Wang, Z. Zhang, J. Xie, and Z. Xing, "A multi-objective genetic algorithm based on fitting and interpolation," *IEEE Access*, vol. 6, pp. 22920–22929, 2018.

[22] J. Meena, M. Kumar, and M. Vardhan, "Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint," *IEEE Access*, vol. 4, pp. 5065–5082, 2016.

[23] B. B. Munyazikwiye, H. R. Karimi, and K. G. Robbersmyr, "Optimization of vehicle-to-vehicle frontal crash model based on measured data using genetic algorithm," *IEEE Access*, vol. 5, pp. 3131–3138, 2017.

[24] X.-Y. Liu, Y. Liang, S. Wang, Z.-Y. Yang, and H.-S. Ye, "A hybrid genetic algorithm with wrapper-embedded approaches for feature selection," *IEEE Access*, vol. 6, pp. 22863–22874, 2018.

[25] M. B. Bashir and A. Nadeem, "Improved genetic algorithm to reduce mutation testing cost," *IEEE Access*, vol. 5, pp. 3657–3674, 2017.

[26] S. Aly and S. Almotairi, "Deep convolutional self-organizing map network for robust handwritten digit recognition," *IEEE Access*, vol. 8, pp. 107035–107045, 2020.

[27] D. Guamán, S. Delgado, and J. Pérez, "Classifying model-view-controller software applications using self-organizing maps," *IEEE Access*, vol. 9, pp. 45201–45229, 2021.

[28] *UCI Machine Learning Repository*. Accessed: Sep. 1, 2023. [Online]. Available: https://archive.ics.uci.edu/citation

[29] L. Lu Wang, K. Lo, Y. Chandrasekhar, R. Reas, J. Yang, D. Burdick, D. Eide, K. Funk, Y. Katsis, R. Kinney, and Y. Li, "CORD-19: The COVID-19 open research dataset," 2020, *arXiv:2004.10706*.

[30] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979.

[31] G. Varoquaux, L. Buitinck, G. Louppe, O. Grisel, F. Pedregosa, and A. Mueller, "Scikit-learn," *GetMobile, Mobile Comput. Commun.*, vol. 19, no. 1, pp. 29–33, 2015.

HIMANSHU DUTTA is currently pursuing the bachelor's degree in computer science and system engineering with KIIT Deemed to be University, India. He has been working in the field of machine learning and deep learning, for the past three years, with experience in computer vision and NLP. Along with working in a research environment, he has previously worked on different internships. He has published one research paper and is currently working on two others.

SAURABH BILGAIYAN received the B.E. degree in information technology from BIRT, Bhopal, India, in 2012, and the master's and Ph.D. degrees in computer science and engineering from KIIT Deemed to be University, Bhubaneswar, India, in 2014 and 2018, respectively. He has been an Assistant Professor with KIIT Deemed to be University, since 2016. He has published more than 44 research papers in various reputed international journals, conferences, and edited books. He has reviewed various manuscripts in more than 25 international conferences and journals, including *Soft Computing* (Springer), IEEE Access, *IETE Journal of Research* (Taylor and Francis), *Future Generation Computer Systems* (Elsevier), and *Concurrency and Computation: Practice and Experience* (Wiley). His research interests include soft computing, software engineering, cloud computing, image processing, and machine learning.

BHABANI SHANKAR PRASAD MISHRA was born in Talcher, Odisha, India, in 1981. He received the B.Tech. degree in computer science and engineering from the Biju Patnaik University of Technology, Odisha, in 2003, the M.Tech. degree in computer science and engineering from KIIT University, Bhubaneswar, Odisha, in 2005, the Ph.D. degree in computer science from Fakir Mohan University, Balasore, Odisha, in 2011, and the Ph.D. degree from the Soft Computing Laboratory, Yonsei University, South Korea, in 2013. He is currently a Professor with the School of Computer Engineering, KIIT University. He has published more than 80 research papers in reputed journals and conferences, and has edited more than five books of current importance. He is a member of different technical bodies ISTE, CSI, and IET. Under his guidance, two Ph.D. scholars are already been awarded. He was a recipient of the Gold Medal and the Silver Medal, during the M.Tech., for the Best Post Graduate from University.

SATCHIDANANDA DEHURI (Member, IEEE) received the M.Tech. and Ph.D. degrees in computer science from Utkal University, Vani Vihar, Odisha, in 2001 and 2006, respectively. He has been a Professor with the Department of Computer Science (Erstwhile Department of Information and Communication Technology), Fakir Mohan University, Balasore, Odisha, India, since 2013. Prior to this appointment, for a short stint (i.e., from October 2012 to May 2014), he was an Associate Professor with the Department of Systems Engineering, Ajou University, South Korea. He visited the Soft Computing Laboratory, Yonsei University, Seoul, South Korea, as a BOYSCAST Fellow, under the BOYSCAST Fellowship Program of DST, Government of India, in 2008. His research interests include multi-objective optimization, machine learning, and data science. In 2010, he received the Young Scientist Award in Engineering and Technology for the year 2008 from the Odisha Vigyan Academy, Department of Science and Technology, Government of Odisha. In 2021, he received the Teachers Associateship and Research Excellence (TARE) Fellowship from SERB, DST, Government of India, for three years to carry out intensive research on Higher Order Neural Networks for Big Data Analysis at host Institute, ISI Kolkata and Parent Institute, Fakir Mohan University, Balasore.

ASHISH GHOSH (Senior Member, IEEE) starting as an electronic engineer, from 1983 to 1987, he moved to computer science during the master's, from 1987 to 1989, and finally to artificial intelligence (AI) during the Ph.D., in 1993. He completed the postdoctoral research in Japan, from 1995 to 1996, and then he joined ISI, as a Faculty Member, in 1997. He is contributing immensely to the field of AI, machine learning, image/video analysis and data science. He has already published about 250 research articles on these subjects and has an H-index of 40. He is a member of the founding team that established the National Center for Soft Computing Research, Indian Statistical Institute, Kolkata, in 2004, with funding from the Department of Science and Technology, Government of India. Since 2010, he has been acting as the In-Charge of the Center that has been recognized as an Associate Institute of ISI. He served as the Head of the well-known Machine Intelligence Unit, ISI, from 2013 to 2014. He is a Full Professor. He is currently leading the Data Science Research Consortium Project, as the Theme Coordinator, to manage a number of projects assigned to different reputed Indian Institutes under ICPS Program of DST. He is a fellow of the West Bengal Academy of Science and Technology.

He is a member of the Research Advisory Committee of many Academic organizations, and SEBI (Mumbai, India), and was nominated to the Academic Council of Banasthali Vidyapith, as an Eminent Educationist. For his pioneering contribution in computational intelligence and image analysis, he received the Young Scientist Award from the Indian Science Congress, in 1992; the Young Scientist Medal from INSA, in 1995; and the Young Associateship of Indian Academy of Sciences, in 1997. For his excellent service to IEEE, he received the IEEE-GRSS Regional Leader Award, in 2019. He is an Associate Editor of several international journals, such as *IET Journal of Computer Vision*, *CAAI Transactions on Intelligence Technology* (published from IET), *Sadhana*, and *Journal of Banking and Financial Technology*; and a Series Editor of *Communications in Computer and Information Science* (Springer-Nature).