Received August 24, 2020, accepted September 4, 2020, date of publication September 14, 2020,
date of current version September 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3024111

Unsupervised Functional Link Artificial Neural Networks for Cluster Analysis

BHABANI SHANKAR PRASAD MISHRA[1], OM PANDEY[1], SATCHIDANANDA DEHURI[2], (Member, IEEE), AND SUNG-BAE CHO[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]Soft Computing Laboratory, Department of Computer Science, Yonsei University, Seodaemun-gu 120-749, South Korea

Corresponding author: Sung-Bae Cho (sbcho@yonsei.ac.kr)

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korean Government (MSIT), Artificial Intelligence Graduate School Program, Yonsei University, under Grant 2020-0-01361.

ABSTRACT In this paper, we propose a novel method of cluster analysis called unsupervised functional link artificial neural networks (UFLANNs), which inherit the best characteristics of functional link artificial neural networks and self-organizing feature maps (SOFMs). UFLANNs adopt three types of basis functions such as Chebyshev, Legendre orthogonal polynomials, and power series for mapping the input data into a new feature space with higher dimensions, where the objects are clustered based on the principle of competitive learning of SOFMs. The effectiveness of this algorithm has been tested with various artificial and real-life datasets including remote sensing images. A thorough comparison with other popular clustering algorithms shows that the proposed method is promising in revealing clusters from many complex datasets.

INDEX TERMS Cluster analysis, competitive learning, FLANN, SOFM.

I. INTRODUCTION

An unsupervised learning system evolves to extract potential characteristics or regularities from underlying data, without being stated what outputs or class labels associated with the given feature vectors are desired [1], [2]. In other words, the learning system perceives and categorizes persistent input vectors without any feedback from the environment in particular an external supervisor or critic. Thus this type of learning is popularly and frequently employed for data clustering [3]–[6], feature extraction [7], and similarity detection [8]. In a nutshell, this paper focuses on developing an unsupervised learning of data clustering.

Data clustering is a fundamental data analysis tool in the area of data mining [9], [10], pattern recognition [11], [12], [41], image analysis [47], [48], feature extraction [13], [14], vector quantization [15], [16], image segmentation [17], [18], function approximation [19], [20], dimensionality reduction [49], [50] and big data analysis [21], [22]. It is a process of revealing inherent structures present in a set of objects based on proximity measure and competitive learning. Let us define the problem of partitional clustering through a metric space.

A metric space is a tuple (X, d), where X is a set and $d : XxX \rightarrow [0, \infty)$ is a metric and satisfy the axioms like (i) $d(x, y) = 0$ (ii) $d(x, y) = d(y, x)$ and (iii) $d(x, z) \leq d(x, y) + d(y, z)$. A set $P \subseteq X$ is defined together with a parameter 'k'. The main aim is to find 'k' points s. t., $k \in C \subseteq P$ so that maximum distance of a point in P to the closest point in C is optimized (in this case minimized). In other words, minimize the cost $r_{\infty}^{c}(P) = Max_{p \in P}\, d(p, C)$. Formally, clustering with 'k' means problem is to find a set C of k points such that $r_{\infty}^{C}(P)$ is minimized i.e., $r_{\infty}^{opt}(P, k) = \min_{c,|c|=k} r_{\infty}^{C}(P)$.

That is every unit in a cluster is in distance at most $r_{\infty}^{C}(P)$ from it's respective centre of gravity (mean) and clustering with k center is treated as an intractable problem. As of now, we do not have polynomial-time exact algorithms for solving intractable problems. Therefore, it is an urgent need to give attention to approximation algorithms. This is one motivation for developing an UFLANN.

In neural networks, unsupervised learning attempts to learn for responding to different feature vectors with different parts of the network structure [4], [42]. With no available information in connection to the desired outputs, unsupervised learning in artificial neural networks update weights vector only based on given input vectors and their connectedness [5].

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

The competitive learning network is a very popular approach in various areas to achieve this type of unsupervised learning of data clustering [42], [43]. All input neurons 'i' are connected to all output neurons 'j' with weights w_{ij}. The number of inputs is the size of the input vector, while the number of outputs is equivalent to the number of clusters that the data points are to be divided into. A cluster center's position is specified by the weight vector associated with the connections to the corresponding output neuron. The output neuron with the largest activation must be selected for further processing point what is implied by ''competitive'' or ''winner take all'' [1], [34]. The Euclidean distance is used as a dissimilarity measure for competitive learning [43]. The weights of the output neuron with the lowest activation are updated using equation (1).

$$\mathrm{w_k}\,(\mathrm{t}+1) = \mathrm{w_k}\,(\mathrm{t}) + \eta\,(\mathrm{x}\,(\mathrm{t}) - \mathrm{w_k}\,(\mathrm{t})) \tag{1}$$

where w_k, x, t, and η are denoted as weight vector corresponding to a k^{th} neuron of the output layer, input vector, iteration number, and learning rate, respectively.

A competitive learning network performs the data clustering on the given input instances. When the process is completed, the input data is divided into disjoint clusters (in the case of hard clustering) such that similarities between sample points in the same cluster are larger than those in different clusters. A main constraint of competitive learning is that the random initialization of weight vectors may be far away from any input vectors and, in the sequel, it never gets updated. Such type of situation can be prevented by initializing the weight vectors to instances from the input data itself, thereby ensuring that all of the weight vectors get updated when all the input samples are presented. An alternative approach would be to update the weight vectors of both the winning and losing neurons, but use a significantly smaller learning rate η for the losers; this is commonly referred to as ''leaky-learning''.

It is highly desired to change dynamically the learning rate η in the weight update formula of equation (1). A setting of an initial large value of η explores the search space widely; later on, a progressively smaller value refines the weights vector. The operation is very similar to the temperature cooling strategy of simulated annealing [44].

Competitive learning lacks the ability to add new clusters when it is deemed to be necessary. Furthermore, if the learning rate η is a constant, competitive learning does not guarantee stability in revealing clusters; the winning unit responds to particular patterns may continue changing during this training. On the other hand, η, if decrementing with time, may become too small to update cluster centers when new data of a different probabilistic nature are presented. Carpenter and Grossberg [23] referred to such an occurrence as the ''stability-plasticity dilemma'', which is common in designing a machine learning system. Adaptive Resonance Theory (ART) introduced by Grossberg proposes a solution to the above dilemma. Based on ART, Carpenter and Grossberg, proposed a series of similar networks, including ART1 [24], ART2 [24], ART3 [24], [25] and ARTMAP [24], [26].

On the other hand, if the output neurons of a competitive learning network are arranged geometrically (such as in a one-dimensional array or two-dimensional arrays), then we can update the weight vectors of the winners as well as the neighbouring losers. Such a capability corresponds to the notion of Kohonen feature maps [27], [28]. The above unsupervised neural networks have suffered from many difficulties like slow learning rate, trapping in local optimality, not scale well for patterns with a large number of elements, etc. Hence our second motivations ignite us to cope-up with these problems. It may be an alternative to enhance the original representation right from the start, in a linearly independent manner, so that hyper-planes for separation might be learned more readily [47], [48]. One way of enhancing the initial representation at a pattern is to describe it in a space of increased dimensions through functional links. It has also been observed that supervised FLANN has achieved remarkable success in many areas of pattern classification [29]–[31]. Therefore, if we can combine the best attributes of supervised FLANN and competitive learning networks like SOFM [32] for uncovering clusters then some of the problems of traditional clustering approaches including SOFMs can be easily addressed.

Further, in the process of searching an efficient, robust, and scalable clustering algorithm, recently owing to the development of deep learning, a set of new algorithms have been added under the umbrella of deep clustering e.g., Autoencoder [52], Generative Adversarial Network (GAN), [57] Variational Autoencoder (VAE) [58], Deep Subspace Clustering [56], and a few more presented in [51], [53], [54]. From the perspective of network architecture a detailed survey of deep clustering can be found in [55]. Although a series of developments have been made in the area of deep clustering but its computational complexity in the absence of a very powerful computing and visualizing unit opens up avenues for development of new clustering algorithms and this makes the third motivation of this journey for developing UFLANN.

Like SOFM, UFLANN is also trained by the method of competition learning. It learns a weight vector configuration without being told explicitly of the existence of clusters at the input, then it is said to undergo a process of self-organized or unsupervised learning. This is to be contrasted to supervised learning like gradient based learning, delta rule, back-propagation, etc., [40], [41].

The advantages of our method in discovering natural clusters over SOFM can be realized easily because instead of grouping around until we find a suitable sequence of transformation, we enhance the original representation right from the start and the hyper-planes for separation have been learnt more rapidly. In our UFLANN, we enhance the initial representation of a pattern by describing it in a space of increased dimensions.

The rest of the paper is set out as follows. In Section II, we discuss the preliminary materials like FLANN and SOFM. Our contributed work is presented in Section III. The experimental details and conclusions are presented in

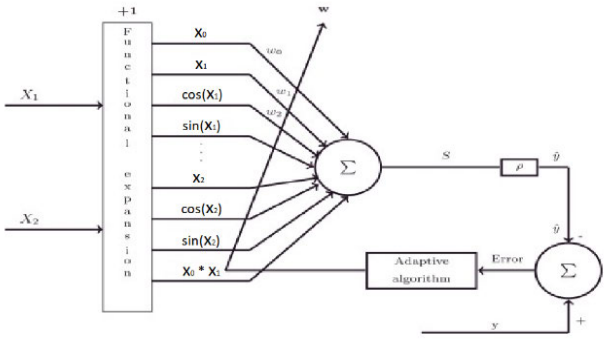


FIGURE 1. A FLANN with adaptive supervised learning.

Sections IV and V, respectively followed by a list of very useful references.

II. PRELIMINARIES

A. FLANNs

The widely used functional neural network was designed to be computationally more feasible and efficient that the multi-layer perceptron. This architecture consists of a single layer of neurons using polynomial functions to expand the feature space to increase variance and faster convergence allowing the algorithm to fit more complex functions. The introduced non-linearity at each of the neurons which makes it easier to back-propagate the gradient as in the general neural networks, increasing the neuron's layers leads to increasing computational complexity. Thus, the introduced nonlinearity whatever they maybe would help us to learn a better representation of the data.

A simple FLANN model [33] with a pattern of two features is shown in Figure 1. For this, in a single layer FLANN consisting of a two-dimensional input sample vector $\vec{x} = [x_1, x_2]^T$ is mapped to a higher dimensional space by functional expansion using trigonometric functions = [(x_1, sin x_1, sin $2x_1$, cos x_1, cos $2x_1$), (x_2, sin x_2, sin $2x_2$, cos x_2, cos $2x_2$), (x_1*x_2)]^T. The expanded feature space is mapped to output activation O_k using weight vector w_k, the linear sum for which can be calculated as:

$$S_k = \sum_{k=1}^{N} w_k x_k + \theta_k \tag{2}$$

θ_k, is the inductive bias to the output. FLANN obtains the solution for the weights iteratively comparing the acquired sum to the ground truth class labels O_k. The error in learning of FLANN is back-propagated by calculating cost functions as,

$$E = \frac{1}{2}[t_k - O_k]^2 = \frac{1}{2}\varepsilon_k^2 \tag{3}$$

$$O_k = f(S_k) \tag{4}$$

where t_k, is the desired output ε_k is the final error and f is a non-linear activation, usually a logistic function to squish values between a specific range.

The final weights are updated using the following policy,

$$w_k(New) = w_k(Old) + \Delta w_k \tag{5}$$

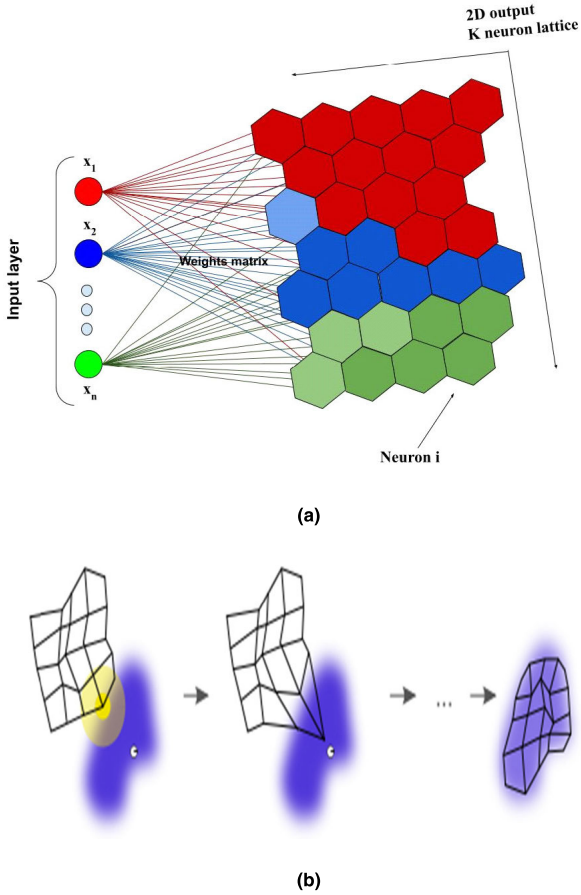


FIGURE 2. (a) Self-organizing feature maps architecture; (b) Topology of SOFM.

where, change is calculated as,

$$\Delta w_k(t) = \left[-\eta(t)\frac{\partial E}{\partial w_k}\right] \tag{6}$$

where, $-\eta(t)$ is the learning rate.

B. SOFMs

Kohonen's Self-organizing feature map (SOFMs) [32], [34] is an artificial neural network based on unsupervised learning. It works on a competitive learning algorithm which learns to fit the given datasets overtime and comes up with a learned representation of a map in the form of a 2D lattice to compress the vector space such that they best approximate the density distribution of the data as shown in Figure 2(b). They also simultaneously distribute the quantization prototypes on the rigid lattice by preserving their neighbourhood relations in the data space, a learned SOFM has relevant clustering information which can be extracted and used to find relationships among the data points, making them widely used for cluster extraction and data analysis tasks [35]–[37].

Let an input vector $x = [x_1, x_2, x_3 \ldots x_n]$, be mapped to each cell v in the rectangular/ hexagonal structure as shown in Figure 2(a). The mxn lattice of neurons is typically smaller than, the x. Each of the ith neuron comprises of weights w_i,

where, $w_i = [w_1, w_2, w_3 \ldots w_m]^T \in R^n$, the adaptation mapping criterion is based on the Euclidean distance norm i.e. $x, w_i \in R^2$. The minimum distance between x, and the weighted neurons is defined by w_c. The updation of weight vectors is inspired by biological neurons that affect spatial neighbouring cells using lateral feedback connections and interactions.

Thus, in the neighbourhood $\rho_c(w_c, k, t)$, around the updated cell c at a given time step t, the ripples of the revision of weights is propagated in a diminishing locality of radius. So, shrinking with time towards the end the radius gets constrained to the best-matching-unit (''winner neuron'') that gets updated, given by:

1. At each time t, present an input $x(t)$, and select the winner,

$$w_c(t) = arg\ min_{k \in m.n} x(t) - w_k(t) \tag{7}$$

2. Update the weights of the winner and its neighbors,

$$\Delta w_k(t) = \eta(t)\rho_c(w_c, t)[x(t) - w_v(t)] \tag{8}$$

Until the map converges.

Here, $\eta(t)$ is the suitable learning rate, $\rho_c(w_c, t)$ is the neighbourhood updating function with exponential decay.

The performance of a map for this optimization task is commonly measured by the mean squared error (MSE):

$$MSE = \frac{1}{K}\sum_1^K \min ||x_k - w_j||^2; \quad j \in \{1, \ldots, M\} \tag{9}$$

Here $|| \ldots ||$ is the Euclidian distance norm.

The algorithm can be briefly described as below:

1. Each neuron's weights are initialized either by randomly or by using some prior domain knowledge.
2. An input vector is selected at random from the set of training data.
3. Every neuron is examined to calculate which one's weights are with proximity of the input vector. The winning node is commonly known as the ''**Best Matching Unit** (BMU)''.
4. Then the neighbourhood of the BMU is computed. The amount of neighbours gradually decreases over time.
5. The weight of the winning neuron is rewarded with becoming more like the sample vector. The neighbours also become more like the sample vector. The closer a node is to the BMU, the more its weights get updated and the farther away the neighbour is from the BMU, the less it learns.
6. Repeat steps 2 to 6 until a stopping criterion is reached.

III. UNSUPERVISED FUNCTIONAL LINK NEURAL NETWORKS

Recall that a number of neural network models for unsupervised learning particularly clustering have been proposed so far [5]. In pattern recognition problem the pattern vector tend to form clusters, a center for each cluster, and therefore the first step for a typical unsupervised learning technique is the estimate of these clusters. The clusters are usually separated by regions of low pattern density. We present here a new method of unsupervised learning in higher-order neural networks known as unsupervised FLANNs. The newly proposed method reveals clusters through two major phases. In first phase we map the given input vectors in to higher dimensions. The details of mapping lower to higher dimensions of the input vectors are given below.

In this work, we have chosen three recurrences in respect of Chebyshev polynomial, Legendre polynomial, and power series for generating the first three polynomials for each category.

The recurrence relation for the Chebyshev polynomials is defined as:

$$Ch_0(x) = 1 \tag{10}$$

$$Ch_1(x) = x \tag{11}$$

$$Ch_{n+1}(x) = 2.x.Ch_n(x) - Ch_{n-1}(x), \quad n \geq 1 \tag{12}$$

In the Chebyshev approximation the average error can be large but the maximum error is minimized. The Chebyshev approximations of a function are said to be min-max approximations of the function.

The Legendre polynomial forms an $L^2([-1, 1])$ - orthogonal set of polynomials and is also good choices for approximation. The following recurrence relation can generate the Legendre polynomials.

$$L_0(x) = 1 \tag{13}$$

$$L_1(x) = x \tag{14}$$

$$L_n(x) = \frac{1}{n}\{(2n-1)\,x.L_{n-1} - (n-1)L_{n-2}\} \tag{15}$$

Similarly, the recurrence relation in respect of power series can be generated as follows:

$$P_0(x) = 1 \tag{16}$$

$$P_{n+1}(x) = xP_n(x) \tag{17}$$

In all the above cases, we have considered the first three polynomials for functional expansion. The reason is that the Principal Component Analysis (PCA) of more added values from other higher-order polynomials, contributes very less in increasing the accuracy or right shape of clusters. Alongside, if we consider more number of polynomials for functional expansion in the method then it leads to higher computational cost.

Now it is to be noted that each of the above polynomial have their merits and demerits, hence to address some of the demerits of the said polynomials, we have taken all three polynomials for functional expansion coherently. For example, let the input vector $\vec{x} = <x_1, x_2>$. Then we map this feature vector to a higher dimension by using these three polynomials combined as follows.

$$\vec{x} = <x_1, x_2>$$

$$\vec{x}_f = <Ch_0(x_1), Ch_1(x_1), Ch_2(x_1), Ch_0(x_2), Ch_1(x_2), Ch_2(x_2), L_0(x_1), L_1(x_1), L_2(x_1), L_0(x_2), L_1(x_2), L_2(x_2), P_0(x_1), P_1(x_1), P_2(x_1), P_0(x_2), P_1(x_2), P_2(x_2)>$$

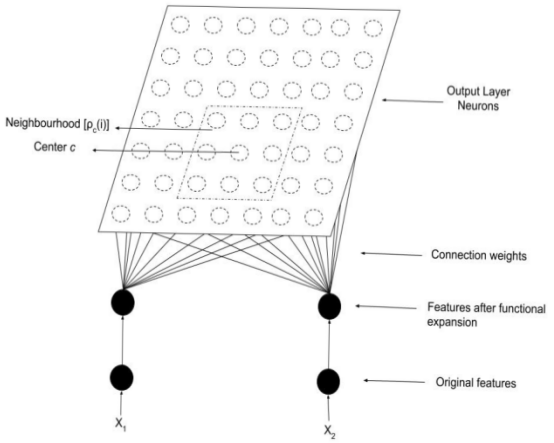


FIGURE 3. Unsupervised FLANN with 2 inputs and 18 features often functional expansion and 49 output neurons at the output layer.

That is a two-dimensional vector is mapped to an 18-dimensional vector. In general if we have a feature vector of n-dimensions, then the new dimension of the feature space will be $3 \times (n \times 3)$.

In the second phase of our proposed method a competitive learning networks also known as Kohonen's feature maps or topology preserving maps are used for revealing the natural clusters. Figure 3 depicts the architecture of the proposed method.

A step-by-step description of the proposed unsupervised FLANN to discover clusters is as follows:

1] The given set of input vectors mapped to a set of vectors with higher dimensions using the first three polynomials from each category like Chebyshev polynomials, Legendre polynomials, and power series. $\vec{x} = \vec{x}_f$, where $\vec{x}$ is the input vector and $\vec{x}_f$ is the vector obtained after functional expansion.

2] The connection weights of the network are initialized using the PCA of the input data obtained after functional expansion.

3] Select the winning output neuron as the one with the highest similarity measure between all weight vectors $\vec{w}_i$ and the vector $\vec{x}_f$. If the Euclidean distance metric is chosen as the dissimilarity measure, then the winning neuron 'c' satisfies the equation given below.

$$\|\vec{x}_f - \vec{w}_c\| = \min_i \|\vec{x}_f - \vec{w}_i\| \tag{18}$$

where, the index 'c' refers to the winning neuron.

4] Let Nc denote a set of index corresponding to a neighbourhood around 'c'. The weights of the winner and its neighbouring units are then updated by: $\Delta\vec{w}_i = \eta(i)\,(\vec{x}_f - \vec{w}_i)$, $i \in Nc$, where $\eta(i)$ is a small positive learning rate. Instead of defining the neighbourhood of a winning unit we use here a neighbourhood function like Gaussian function $\rho_c(w_c, i)$ around a winning unit 'c'. The Gaussian function is defined around 'c' is as follows:

$$\rho_c(w_c, i) = e^{-\left(\frac{-||\rho_i - \rho_c||^2}{2\sigma^2(i)}\right)} \tag{19}$$

where, ρ_i and ρ_c are the position of the output units 'i' and 'c', respectively, and $\sigma(i)$ reflects the scope of the neighbourhood radius which is a monotonically or exponentially decreasing function of time. By using the neighbourhood function, the update formula can be rewritten as:

$$\Delta w_i = \eta\,(i)\,\rho_c\,(w_c, i)\,(\vec{x}_f - \vec{w}_i) \tag{20}$$

where, i is the index for all the output units.

To achieve a better convergence, the learning rate $\eta(i)$ and size of the neighbourhood (or $\sigma(i)$) should be decreased gradually with each time.

5] Finally, the set of output neurons will have the learned representations of the clustered data. So, new data can be now passed through the network and mapped to the consecutive neurons for classification.

IV. EXPERIMENTAL DETAILS

To investigate the advantages of UFLANN in comparison to other unsupervised learning approaches, numerous experimental scenarios were considered. Both real-world and synthetic datasets were considered to compare against the learning tasks and multiple metrics were employed to study thoroughly the benefits of the proposed methodology.

A. DATASET DESCRIPTION

This section broadly describes the datasets that were used. In terms of the real-world datasets, the basic Iris dataset [38], [39] was used (referred to as Dataset 1), where the input features – sepal length, sepal width, petal length, and petal width served as the input to the unsupervised algorithms. The class division served into three main classes–Setosa, Versicolour, Virginica. The dataset was divided into two mutually exclusive parts- 90% for training and 10% for testing purposes. Though training and testing terms in the case of unsupervised learning is confusing but here we have considered 90% for class discovery and then 10% for testing.

Figure 4 shows the synthetic data (Dataset 2 and Dataset 3) for which the considered network had $m = 25$ neurons applied over $L = 750$ points on the 2D plane. Figure.4(a) was designed to cover the edge cases for types of data made use of analyzing effectively if neighbourhood function was optimizing to the exponential decay in $\eta(t)$ & $\eta(v, k, t)$ for (t)th epoch(that is, $t = 1, 2, 3 \ldots . t_{max}$; $t_{max} = 10000$). It was developed to represent concentric circles, using:

$$\theta = 2\pi * Rand\,(i)\,, \quad i \in (0, 1)\ \&\ Rand \in R^n \tag{21}$$

For which,

$$h + \cos(\theta) * r, \quad k + \sin(\theta) * r \tag{22}$$

This generates the individual data points, where h and k are the centers and r is the radius of the concentric circles. Similarly in Figure 4(b), it was made to generate half-moons using line spacing on data points.

For outer circle:

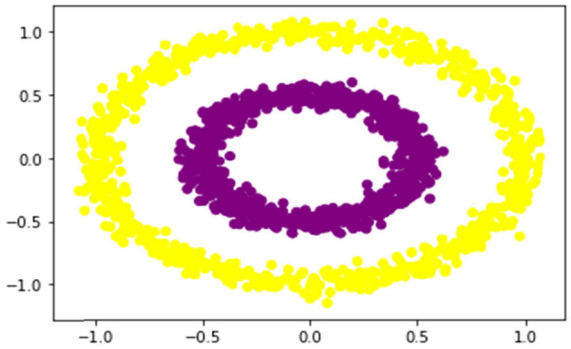
$$x = \cos(\text{random}(0, \pi/2)) \tag{23}$$

$$y = \sin(\text{random}((0, \pi/2)) \tag{24}$$

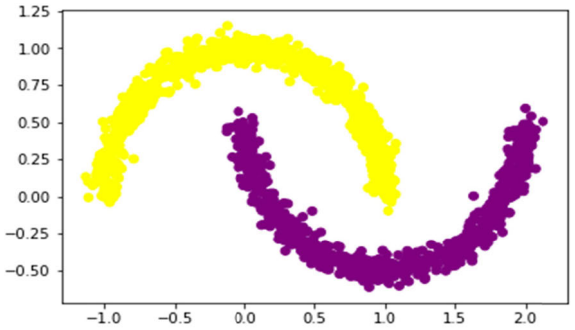
For inner circle:

$$\text{Inner}_x = 1 - x \tag{25}$$

$$\text{Inner}_y = 1 - y - 0.5 \tag{26}$$



(a)



(b)

FIGURE 4. (a) Synthetic - Dataset 2; (b) Synthetic - Dataset 3.

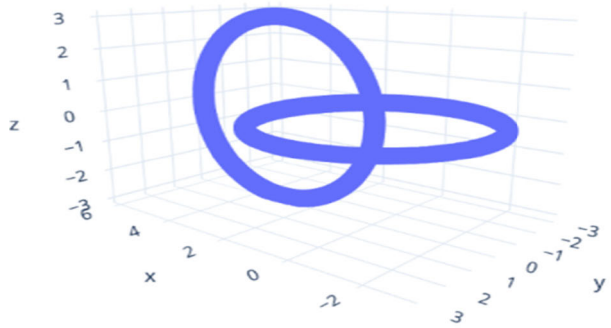


FIGURE 5. Chain-link Dataset (Custom 1).

For Figures 5 and 6 also known as the famous spatial Chain-link datasets were generated using the same methodology as in equations (18) and (19), (L = 2000 points on the 3D plane, as offered in Fundamental Clustering Problems Suit [4]) but the axes are changed and in the Figure 6 Gaussian noise is added to investigate the effect on UFLANN's results. The learning algorithm had $m = 49$ neurons applied over the data in both cases. A data of a total of 112 experiments was averaged to provide for the current noise in Figure 6. This dataset is preferred as a benchmark to justify unsupervised clustering because according to Pearson's coefficient data is non-linearly-separable, so its PCA projection to a vectorial sub-space is impossible without data loss ($\sigma_1 = 33.95, \sigma_2 = 33.46, \sigma_3 = 32.57$).

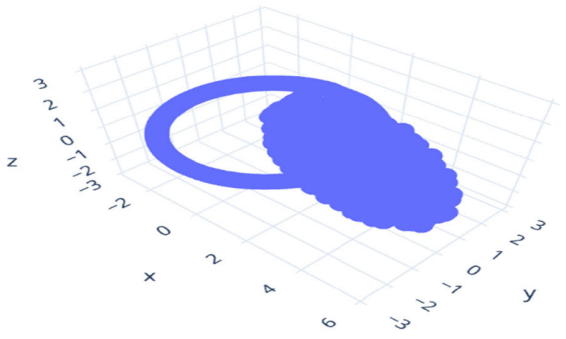


FIGURE 6. Chain-link with increased variance (Custom 2).

A few satellite images were scraped off from the internet to visualize and compare the clustering power of the provided approach, in terms of colour clustering for segmentation tasks. This has been mentioned in the latter part of the paper.

In addition to the above datasets, we have applied our proposed method over two well known datasets MNIST and CIFAR10 that are very popular in the field of clustering. MNIST [45]: A collection of 32 X 32 black and white pictures denoting handwritten digits (0-9), with $\approx$ 60000 images with labeled annotations. This dataset is the extension of the NIST dataset.

CIFAR-10 [46]: This consists of another 60000, 32×32 colour images in 10 classes, with 6000 images per class. These images belong to 10 real-life objects like airplane, cat, deer, etc. The dataset is divided into five training and one testing batch. The test batch contains 1,000 randomly selected images from each class. The training set contains the rest of the images in a random order.

B. EVALUATION METRICS

The performance of a clustering algorithm depends on the type of dataset that we are using and the type of improvement in performance that is to be achieved. For our research, we consider the common clustering metrics that have been used to compare the performance of such algorithms.

Firstly, we use the standardized accuracy evaluation metric because the synthetic/real data we generated/ scraped had predefined classes that could be used to treat it as a standard classification problem, and penalizing the metric for wrongly generated predictions. Thus, the first metric is unsupervised clustering accuracy (ACC):

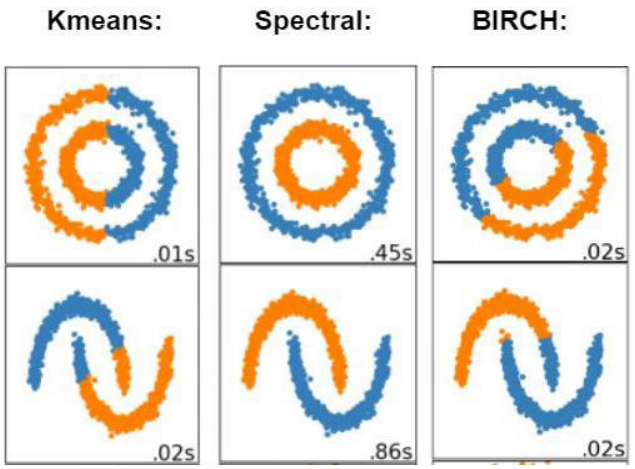
$$ACC = max_m \sum_{i=1}^{n} \frac{1\{y_i = map(c_i)\}}{n} \tag{27}$$

where, y_i is the ground-truth label, c_i is the cluster assignment generated by the algorithm, and the *map* is a mapping function that ranges over all possible one-to-one mappings between assignments and labels.

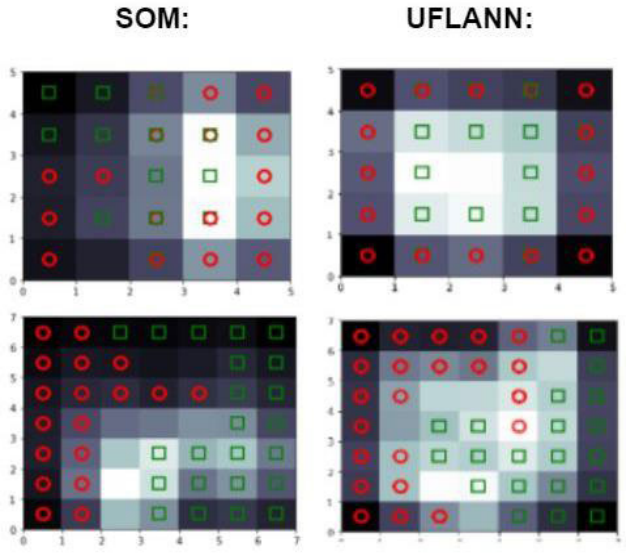
Secondly, we use the Completeness-Score, which can be considered as Completeness metric of a cluster labelling when the ground truth label is given, a clustering result satisfies completeness if all the elements of a given cluster

TABLE 1. Comparison of completeness score of algorithms (in %).

Dataset	Kmeans	SOFM	Spectral	BIRCH	UFLANN
Dataset1	83.6	92.6	999	93.4	**93.4**
Dataset2	26.9	88	100	26	**92.3**
Dataset3	24	100	100	35	**100**



(a)



(b)

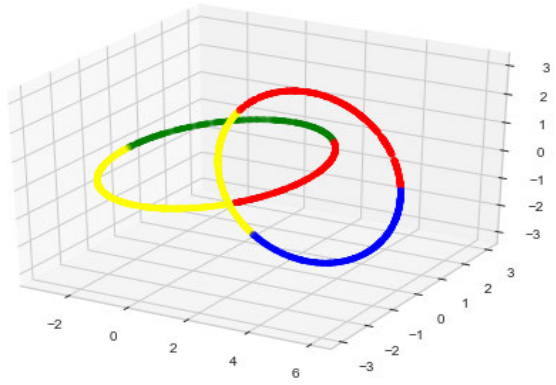
FIGURE 7. (a) Experiments on various datasets; (b) Comparison of SOFM and UFLANN.

are data points that are members of the same class. This can be regarded as a useful metric because it is independent of the absolute values of the labels: the permutation of the class or cluster label values won't change the score value in any way.

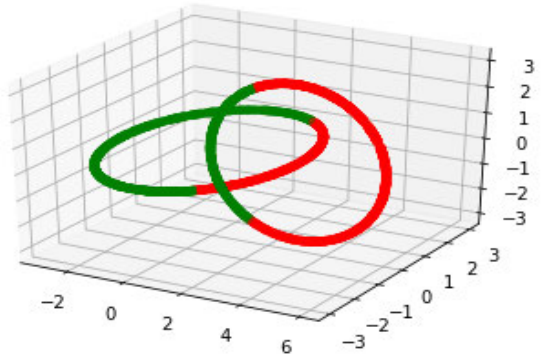
Thirdly, we used the Adjusted Rand Index, a form of the Rand index that is adjusted for the chance grouping of elements. The calculation of Rand Index mostly relies on the calculation of class-based contingency matrix thus for a set of N elements where $S = \{S_0, S_1, S_2 \ldots\ldots S_n\}$ with partitions

TABLE 2. Comparison of accuracy of algorithms (in %).

Dataset	Kmeans	SOFM	Spectral	BIRCH	UFLANN
Dataset1	92	98	66.67	12	**98**
Dataset2	36	98	100	36	**99.3**
Dataset3	34	100	100	20	**100**



(a)



(b)

FIGURE 8. (a) K-means clustering on Chain-Link; (b) Mini-Batch K-means clustering on Chain-Link.

$A = \{A_0, A_1, A_2 \ldots\ldots A_r\}$ and $B = \{B_0, B_1, B_2 \ldots\ldots B_s\}$ ought to be containing r and s number of subsets respectively.

Thus for two such subsets, the Rand Index is:

$$R = \frac{(tp + tn)}{(tp + fp + fn + tn)} \tag{28}$$

tp - True positive; tn- True Negative; fp- False Positives; fn - False Negatives.

But, the Adjusted Rand Index is the corrected-for-chance version of the Rand index, the baseline uses the expected similarity of all pairwise comparisons between clustering specified by a random model, usually, Rand Index yields values between 0 and +1 but adjusted Rand Index can give negative results for unexpected values.

We also use the Silhouette Coefficient which is mostly used for data where the ground truth labels are not known, but we have not included the results as it did not seem to show the considerable difference for different clustering which is evident as it is calculated using the mean intra-cluster distance

TABLE 3. Comparison of silhouette coefficient of algorithms (in %).

Dataset	Kmeans	SOFM	Agglomerative	DBSCAN	UFLANN
Dataset2	-0.005	1.0	0.002	1.0	**1.0**
Dataset3	0.26	1.0	0.493	1.0	**1.0**
Custom1	0.1	1.0	0.489	1.0	**1.0**
Custom2	0.27	0.92	0.419	0.22	**1.0**

TABLE 4. Accuracy of clustering on MNIST and CIFAR-10 dataset (in %).

Dataset	Kmeans	Agglomerative	DBSCAN	SOFM	UFLANN
CIFAR-10	9.4	10.6	8.58	36.3	**44.2**
MNIST	3.5	18.5	19.2	91.6	**93.2**

TABLE 5. Completeness score of clustering on MNIST and CIFAR-10 dataset (in %).

Dataset	Kmeans	Agglomerative	DBSCAN	SOFM	UFLANN
CIFAR-10	6.8	8.2	57.2	12.6	**13.4**
MNIST	50.4	57.6	67.2	81.7	**86.4**

a and the mean nearest-cluster distance b for each sample. Thus, Silhouette Coefficient for a sample is:

$$Sc = \frac{(b-a)}{max(a,b)} \tag{29}$$

In our work, we recognize that the use of popular clustering metrics for famous concentric circles problems would prove incompetent.

C. ENVIRONMENTAL PARAMETERS

The training for the proposed methodology was carried out on an Intel Core i5, 7th Gen, 8.00GB RAM, and a Microsoft Windows 10 Home Edition based personal computer. The programming was carried out in Python 3.6, in Anaconda 3 development environment with Spyder. The datasets were divided into 90-10 training is to testing ratio for all data. The hyper-parameters used during experimentation have been specified in Section D. To validate our method on the MNIST [45] and CIFAR10 dataset [46], the input is flattened in to 784 units. Each row works as a feature vector and taken as an input to clustering algorithm.

D. RESULTS AND DISCUSSION

The performance of the proposed unsupervised FLANN as already discussed in Section III is compared with various competitive baselines and methods to prove the efficiency as tabulated in Table 1. Again the proposed method was also applied over the well-known data set MNIST and CIFAR10, the accuracy and the completeness score is presented in the Table 4 and 5. Figure 7 shows the plot for the ''averaged experiments''. This is because BIRCH and K-means perform well only on linearly separable data. Moreover, Figure 7(b) the additional 2D lattice maps provide insights into the data density structure of particular clusters in our approach. As we

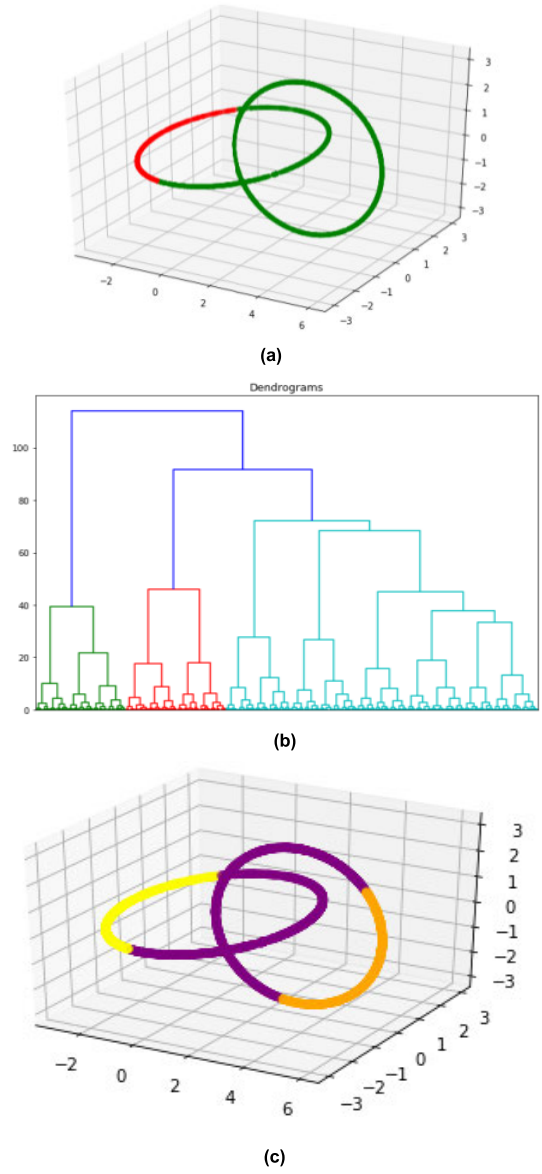


FIGURE 9. (a) Clustering with 2 clusters; (b) Dendograms on data; (c) Clusters according to dendograms.

can see, UFLANN fairs over all the other approaches in terms of data representation by efficiently segregating the data. This shows the formation of two categories as far as the number of detected clusters (mean $\pm$ std. deviation) $= (2.0 \pm 0.0)$, which is a perfect match as represented by the symbols. Figure 8(a), (b) is the averaged result of 55 experiments done, on Figure 5, provides clear evidence on the presence of 4 and 2 cluster categories for K-means and Mini-Batch K-means respectively. The performance of clustering algorithms varies because of the random initialization of the starting point. Thus, to evaluate how sensitive the unsupervised FLANN is to the initialization, UFLANN has been executed 50-80 times on each dataset described in subsection A of IV. Table 2 helps us to compare the accuracy of different clustering algorithms against the same set of datasets. Similarly, in Table 3 we

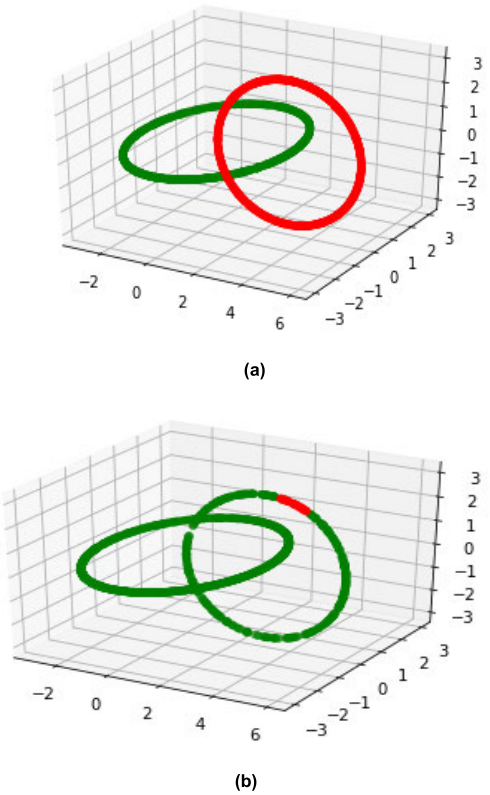


FIGURE 10. (a) Clustering with equal density; (b) Clustering reducing density of one cluster.

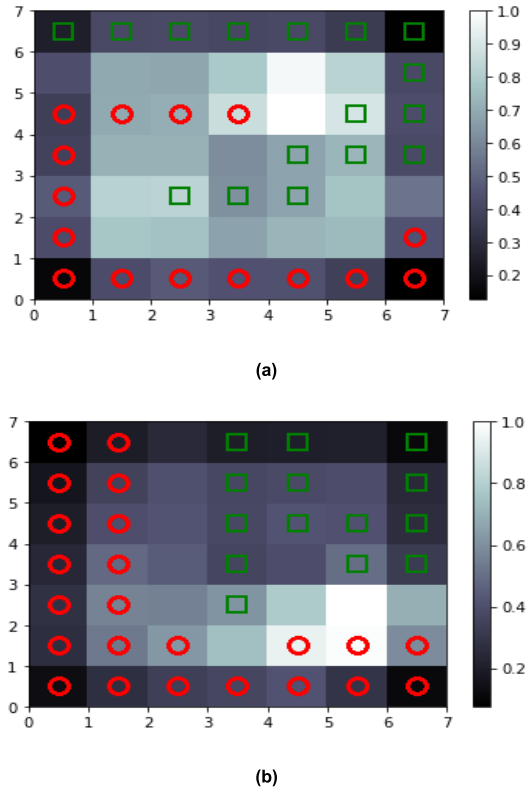


FIGURE 11. (a) Application of SOFM; (b) Proposed method.

compare the silhouette coefficient of various datasets against the set of algorithms considered in this work. The Iris dataset (Dataset 1) has not been considered in the table as it had been considered earlier and our approach showed a fairly competitive baseline w.r.t it. We have replaced the competitive baselines with Agglomerative and DBSCAN clustering algorithms as they proved to be more useful in our experiments. In both cases, our algorithm fairs over the other baselines.

Howsoever, most competitive baselines come very close to the proposed methodology. But, further investigation into these algorithms revealed flaws that were exposed during our experiments - making them fail. Further in this section of the paper, we have discussed how minor changes in the used datasets, breakdown of the competitive baselines, and how our algorithm significantly would outperform the considered baselines in each case.

By applying our proposed method on the MNIST data set we got high accuracy, which is tabulated in Table 4. As the MNIST dataset consist of digits with black back ground, hence it made easy for UFLANN to identify the underlying function representing the digit. Again UFLANN gives better result in comparison to all other algorithms as the input provided to the UFLANN is passing through the Functional Expansion Network which represents the input in the form of different polynomial functions like legendry polynomial, Chebyshev and power series. While calculating completeness score as shown in Table 4 it is clear that the proposed method could able to classify the dataset properly which proves that there is no imbalance or bias towards a certain cluster. So, it can be derived that the proposed model has learned the underlying relationship between the data properly and act as a good classifier for each of the cluster.

Fig. 9 describes the use of Hierarchical Clustering using agglomerative technique with the Euclidian affinity and linkage ward. The drawback of this approach is the algorithm itself utilizing-$\mathbf{min}\left\{\boldsymbol{d}\left(\boldsymbol{a},\boldsymbol{b}\right):\boldsymbol{a}\in\boldsymbol{A},\boldsymbol{b}\in\boldsymbol{B}\right\},\boldsymbol{A},\boldsymbol{B}\rightarrow\boldsymbol{obsv}$. i.e. greedy approach for calculating core cluster center for intra-cluster variance rather than expanding or learning vector subspace or explicitly-differentiable features. Experiments were carried out using different values of threshold $\mathrm{d_{coef}} = 0.1, 0.2, 0.3, 0.4 \ldots .2.0$, and $k = 0, 1 \ldots 4$ clusters under observation to see how the algorithm behaves. This leads to unusual clustering patterns as visualized in dendograms in Figure 9(b).

Thus, the clustering categories were to be amongst, (4.0 ± 2.0) as observed. Density-based clustering was analyzed as a competitive baseline due to the efficiency in near-perfect classification as shown in Figure. 10. Through various observations, it can be analyzed that the concept of the algorithm heavily banks on recursively finding all its density connected points and assigning them to the same cluster as the core point. Thus, if the cluster capacity of 1^{st} cluster was reduced to less than 1/2 of other, then the algorithm fails as in Figure 10(b). We experimented on the neighbourhood of the data points $eps = 0.1, 0.11, 0.12 \ldots\ldots 0.3$ and the

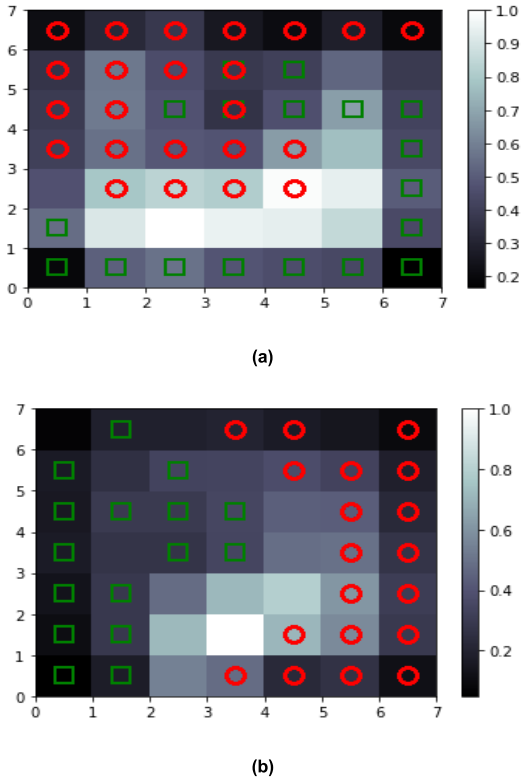


FIGURE 12. (a) Clustering using SOFM; (b) Proposed method.

multiple values of Gaussian noise the average result of which was taken after analysis. Finally, the results of our proposed methodology and the comparison with the basic SOFM are seen in Figure 11. The 2D grid of neurons by using a partial BP algorithm on the current neuron and the nearby neighbourhood of neurons, thus we see that the clustering is much more efficient, as seen in Figure 11(b). The results provided here are the best case are derived from 62 out of the 85 trials carried out (73%), furthering our study into the non-linearity of representation and performance over the previous dataset we experimented by trying to scatter the points of a specific cluster over a more generalized area i.e. adding noise for allowing expanding the vector subspace which would affect such that the distance metric of features between the clusters. Finally, we can see the advantages of UFLANN in Figure 12 where it out performs the basic SOFM algorithm as the expansion of the functional link allow the learning neurons to better map the data from Figure 6 as seen in Figure 12 (b). This approach was tested with multiple hyper-parameters $\eta(\mathrm{i}) \in 0.1, 0.2 \ldots 0.5$ and $\rho_c(w_c, i) \in 1.0, 1.1, 1.2 \ldots .2.0$ the provided results are the generalized versions of the obtained results.

Clustering on Image Data: The technique of reduction of image size by losing out on "less-important" information like colours is a concept introduced earlier. This technique generally utilizes unsupervised algorithms to learn information and can be used to segment satellite images by performing colour segmentation on images. As we proposed unsupervised FLANN which belongs to the plethora of such

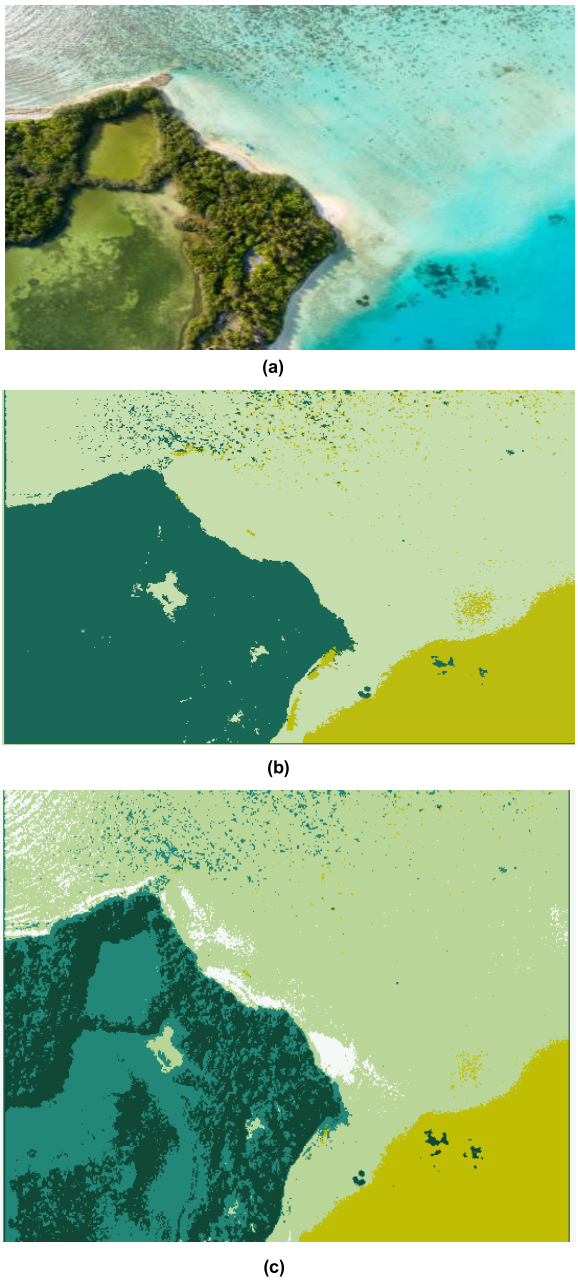
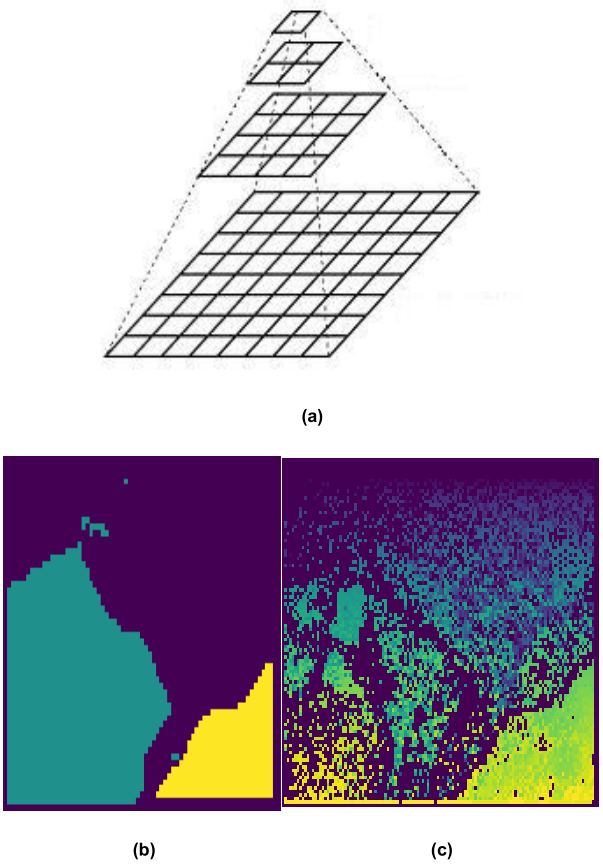


FIGURE 13. (a) Given image; (b) Clustering k = 3; (c) Clustering k = 5.

algorithms by experimental analysis we apply the proposed methodology to images and compare it with previous efforts to do the same.

Figure 13 gives the vague conception of the clustering methodology of K-means, the most common pitfall in this is the specification of clusters beforehand which does not give the algorithm enough dynamicity to adapt to more complex data. The images are generally converted to HSV format for giving ease to clustering but this may lead to important feature loss in terms of small images with similar colors. Figure 14 shows the application of hierarchical clustering on such data, since the algorithm requires the consideration of all points in the linkage table for complex data like images the number of features are exponentially greater thus the image



(a)

(b) (c)

FIGURE 14. (a) The process of down-sampling via Gaussian kernel); (b) Down-sampled image with hierarchical clustering; (c) Density-based agglomerative clustering on the image.

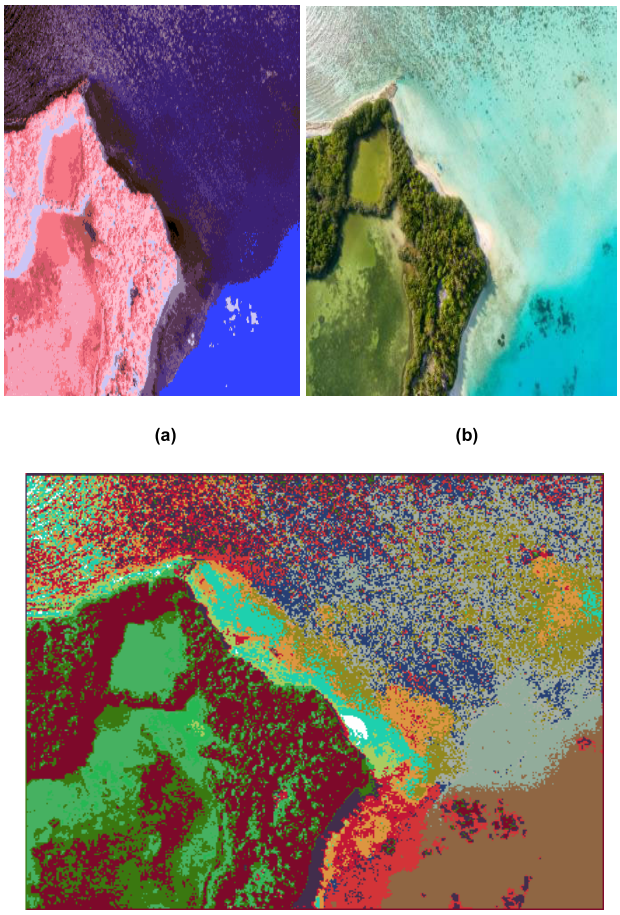
has to be down sampled by using a Gaussian kernel i.e.

$$\frac{1}{16}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Yet, this method also leads to reducing the quality of the image and hence a reduction in the overall quality of image leading to loss of features. Density-based clustering as used earlier as a baseline is generally not commonly used for such tasks as the pixels clutter around regions of high pixel concentration as seen in Figure 14 (b).

For images we improved our algorithm such that the data to the algorithm in the form of batches which also is a hyper parameter effectively making use of the T-based clustering [4] so that data points are very near to each other do not get cluster in different clusters.

In Figure 15, using our proposed analogy, we see that the pixel values of the image increase overall and hence giving a more enhance image because the distances between neighbouring pixels values increase exponentially but this also has its pitfalls that it increases the Gaussian noise thus we mean-normalize the data as in Figure 15(c), Figure 16(c) which wouldn't cause a great decrease in performance



(a) (b)

(c)

FIGURE 15. (a) Input image; (b) Image using SOFM; (c) Proposed method.

because it was a linear transformation but it would allow us to learn more features suppress the unnecessary small features which are getting added to increase the unnecessary number of clusters which are present. The increase in the distance between different features which would effectively allow us to segment the image based on the color representations. We also saw that increasing the number of neurons allows us to learn more complex colors and improve distinguishing between the color representations of the images. UFLANN will especially aid the situation of low lighting or dim photography as it expands the feature space and then contracts bleak color representations which become separable this is also illustrated in Figure 16.

E. TIME COMPLEXITY ANALYSIS

In this section, we get into the details of UFLANN, and a brief analysis of our algorithm in comparison with the benchmark algorithms in consideration. The time complexity of the SOM is considered to be O(NC) i.e. N is the input vector size and C is the number of document presentation cycles. So subsequently UFLANN in worst case performs a given classification in O(k.n.m) time complexity for a given sample where "k" is the number of polynomial expansions and n, m are the sizes of the 2D lattice, where the input vector

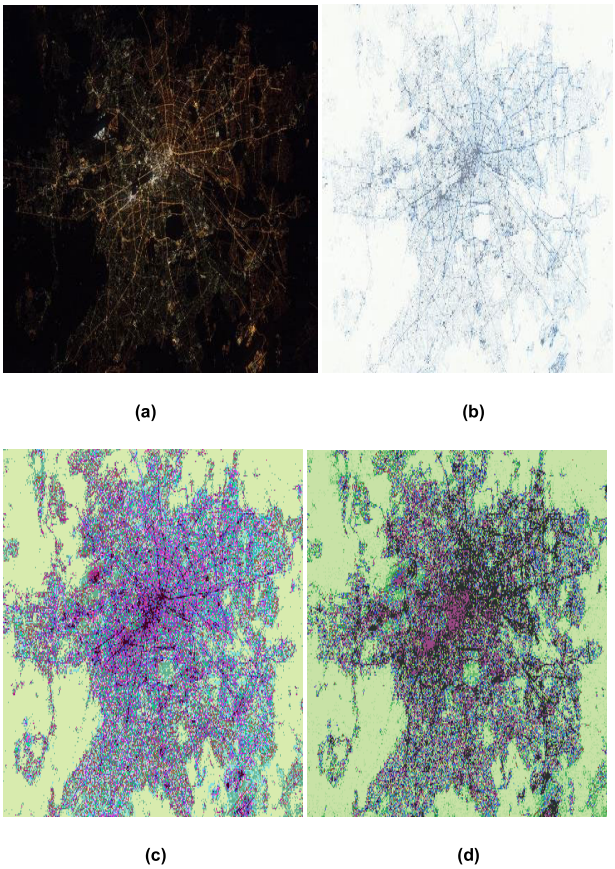


FIGURE 16. A comparison between (a) Input image, (b) Image generated using SOFM (c) Proposed method + Mean-normalization (d) Proposed method.

is compared to each of the weights to find the BMU. The average time taken by the algorithm is 0.026s for inference, compare that to the SOFM algorithm that takes 0.021s, and the sole FLANN algorithm that takes 0.015s for non-image data. The slight increase in time can compensate for the gain in accuracy achieved by the network. When compared with other benchmark algorithms for image datasets, it can achieve inference faster than most algorithms except simple ones like k-means which again is due to time-accuracy trade- off.

V. CONCLUSION

The proposed approach UFLANN has designed by inheriting the best attributes of FLANN and the competitive learning mechanism of SOFM. The experimental design of the proposed approach throws the light that UFLANN is better in accuracy than other algorithms taken in this paper for comparison. The results obtained show that this methodology can be used as a complete solution to clustering with acceptable accuracy for both numerical and image datasets. The main issues of colour image segmentation is being systematically addressed in this paper including perceptual uniformity in colour representation, colour reduction by mean normalization and clustering in unsupervised segmentation. The UFLANN algorithm shall serve very useful in photo sensitive base colour image segmentation in low-lighting conditions. The approach shows good results and has a straight-forward application in the vision domain. In nutshell, we provide a novel unsupervised learning based on FLANN that can find its application in a variety of fields like remote sensing image processing, earthquake studies, outliers detection, insurance, marketing, etc.

REFERENCES

[1] A. J. Richardson, C. Risien, and F. A. Shillington, "Using self-organizing maps to identify patterns in satellite imagery," *Prog. Oceanogr.*, vol. 59, nos. 2–3, pp. 223–239, Oct. 2003, doi: 10.1016/j.pocean.2003.07.006.

[2] M. Garofalakis, J. Gehrke, and R. Rastogi, *Data Stream Management Processing High-Speed Data Streams*. Berlin, Germany: Springer, 2016.

[3] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, 1996, pp. 226–231.

[4] K.-L. Du, "Clustering: A neural network approach," *Neural Netw.*, vol. 23, no. 1, pp. 89–107, Jan. 2010, doi: 10.1016/j.neunet.2009.08.007.

[5] 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, doi: 10.1145/331499.331504.

[6] Y.-X. Wang and H. Xu, "Noisy sparse subspace clustering," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 320–360, 2016.

[7] D. Forsyth, "Clustering," in *Applied Machine Learning*. Cham, Switzerland: Springer, 2019, pp. 155–182.

[8] M. Garza-Fabre, J. Handl, and J. Knowles, "An improved and more scalable evolutionary approach to multiobjective clustering," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 515–535, Aug. 2018, doi: 10.1109/tevc.2017.2726341.

[9] Z. Ge, Z. Song, S. X. Ding, and B. Huang, "Data mining and analytics in the process industry: The role of machine learning," *IEEE Access*, vol. 5, pp. 20590–20616, 2017.

[10] A. Dutt, M. A. Ismail, and T. Herawan, "A systematic review on educational data mining," *IEEE Access*, vol. 5, pp. 15991–16005, 2017, doi: 10.1109/access.2017.2654247.

[11] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 4–37, Jan. 2000.

[12] H. Strasburger, I. Rentschler, and M. Jüttner, "Peripheral vision and pattern recognition: A review," *J. Vis.*, vol. 11, no. 5, p. 13, 2011, doi: 10.1167/11.5.13.

[13] T. Wiatowski and H. Bolcskei, "A mathematical theory of deep convolutional neural networks for feature extraction," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1845–1866, Mar. 2018, doi: 10.1109/tit.2017.2776228.

[14] B. Liu, X. Yu, P. Zhang, A. Yu, Q. Fu, and X. Wei, "Supervised deep feature extraction for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 4, pp. 1909–1921, Apr. 2018, doi: 10.1109/tgrs.2017.2769673.

[15] L. Ai, J. Yu, Z. Wu, Y. He, and T. Guan, "Optimized residual vector quantization for efficient approximate nearest neighbor search," *Multimedia Syst.*, vol. 23, no. 2, pp. 169–181, Mar. 2017, doi: 10.1007/s00530-015-0470-9.

[16] C. Karri and U. Jena, "Fast vector quantization using a bat algorithm for image compression," *Eng. Sci. Technol., Int. J.*, vol. 19, no. 2, pp. 769–781, Jun. 2016.

[17] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018, doi: 10.1109/tpami.2017.2699184.

[18] D. Marmanis, K. Schindler, J. D. Wegner, S. Galliani, M. Datcu, and U. Stilla, "Classification with an edge: Improving semantic image segmentation with boundary detection," *ISPRS J. Photogramm. Remote Sens.*, vol. 135, pp. 158–172, Jan. 2018, doi: 10.1016/j.isprsjprs.2017.11.009.

[19] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Netw.*, vol. 107, pp. 3–11, Nov. 2018, doi: 10.1016/j.neunet.2017.12.012.

[20] G. Pang, L. Yang, and G. E. Karniadakis, "Neural-net-induced Gaussian process regression for function approximation and PDE solution," *J. Comput. Phys.*, vol. 384, pp. 270–288, May 2019, doi: 10.1016/j.jcp.2019.01.045.

[21] L. A. Tawalbeh, R. Mehmood, E. Benkhlifa, and H. Song, "Mobile cloud computing model and big data analysis for healthcare applications," *IEEE Access*, vol. 4, pp. 6171–6180, 2016, doi: 10.1109/access.2016.2613278.
[22] B. Suvarnamukhi and M. Seshashayee, "Big data concepts and techniques in data processing," *Int. J. Comput. Sci. Eng.*, vol. 6, no. 10, pp. 712–714, Oct. 2018, doi: 10.26438/ijcse/v6i10.712714.
[23] S. Grossberg, "Adaptive resonance theory," *Scholarpedia*, vol. 8, no. 5, p. 1569, 2013, doi: 10.4249/scholarpedia.1569.
[24] *Encyclopedia of Machine Learning Data Mining—ART*. Accessed: Nov. 17, 2019. [Online]. Available: http://sites.bu.edu/steveg/files/2016/09/Encyclopedia-of-Machine-Learning-Data-Mining-ART-Carpenter-Grossberg-2016.pdf
[25] G. A. Carpenter and S. Grossberg, "ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures," *Neural Netw.*, vol. 3, no. 2, pp. 129–152, 1990, doi: 10.1016/0893-6080(90)90085-y.
[26] G. P. Amis and G. A. Carpenter, "Self-supervised ARTMAP," *Neural Netw.*, vol. 23, no. 2, pp. 265–282, Mar. 2010, doi: 10.1016/j.neunet.2009.07.026.
[27] C. Banbury, R. Mason, I. Styles, N. Eisenstein, M. Clancy, A. Belli, A. Logan, and P. G. Oppenheimer, "Development of the self optimising kohonen index network (SKiNET) for Raman spectroscopy based detection of anatomical eye tissue," *Sci. Rep.*, vol. 9, no. 1, Dec. 2019, Art. no. 10812, doi: 10.1038/s41598-019-47205-5.
[28] E. F. Galutira, A. C. Fajardo, and R. P. Medina, "A novel Kohonen self-organizing maps using exponential decay average rate of change for color clustering," in *Intelligent and Interactive Computing* (Lecture Notes in Networks and Systems). Singapore: Springer, 2019, pp. 23–33, doi: 10.1007/978-981-13-6031-2_28.
[29] S. Dehuri and S.-B. Cho, "A comprehensive survey on functional link neural networks and an adaptive PSO–BP learning for CFLNN," *Neural Comput. Appl.*, vol. 19, no. 2, pp. 187–205, Mar. 2010, doi: 10.1007/s00521-009-0288-5.
[30] F. Herrera, F. Charte, A. J. Rivera, and M. J. Jesus, *Multilabel Classification*. Cham, Switzerland: Springer, 2016.
[31] A. Tharwat, "Classification assessment methods," *Appl. Comput. Informat.*, Aug. 2020, doi: 10.1016/j.aci.2018.08.003.
[32] W. Zhang, J. Wang, D. Jin, L. Oreopoulos, and Z. Zhang, "A deterministic self-organizing map approach and its application on satellite data based cloud type classification," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Seattle, WA, USA, Dec. 2018, pp. 2027–2034.
[33] B. B. Misra and S. Dehuri, "Functional link artificial neural network for classification task in data mining," *J. Comput. Sci.*, vol. 3, no. 12, pp. 948–955, Dec. 2007.
[34] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990, doi: 10.1109/5.58325.
[35] A. Ultsch, "Clustering with SOM: U*C," in *Proc. Workshop Self-Organizing Maps*, Paris, France, 2005, pp. 75–82.
[36] E. Diday, G. Govaert, Y. Lechevallier, and J. Sidi, "Clustering in pattern recognition," in *Digital Image Processing* (NATO Advanced Study Institutes Series: Mathematical and Physical Sciences), vol. 77, J. C. Simon and R. M. Haralick, Eds. Dordrecht, The Netherlands: Springer, 1981.
[37] J. Qian, N. P. Nguyen, Y. Oya, G. Kikugawa, T. Okabe, Y. Huang, and F. S. Ohuchi, "Introducing self-organized maps (SOM) as a visualization tool for materials research and education," *Results Mater.*, vol. 4, pp. 1–14, Dec. 2019.
[38] S. Dehuri, "A novel learning scheme for Chebyshev functional link neural networks," *Adv. Artif. Neural Syst.*, vol. 2011, pp. 1–10, Jan. 2011, doi: 10.1155/2011/107498.
[39] D. Dua and C. Graff. (2020). *UCI Machine Learning Repository: Citation Policy*. [Online]. Available: https://archive.ics.uci.edu/ml/citation_policy.html
[40] W. Sheng, P. Shan, J. Mao, Y. Zheng, S. Chen, and Z. Wang, "An adaptive memetic algorithm with rank-based mutation for artificial neural network architecture optimization," *IEEE Access*, vol. 5, pp. 18895–18908, 2017.
[41] G. P. Zhang, "Neural networks for classification: A survey," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 30, no. 4, pp. 451–462, Nov. 2000, doi: 10.1109/5326.897072.
[42] L. Wang, "On competitive learning," *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 1214–1217, Sep. 1997, doi: 10.1109/72.623224.
[43] X. Yang, J. Cao, Y. Long, and W. Rui, "Adaptive lag synchronization for competitive neural networks with mixed delays and uncertain hybrid perturbations," *IEEE Trans. Neural Netw.*, vol. 21, no. 10, pp. 1656–1667, Oct. 2010.
[44] S. Xavier-de-Souza, J. A. K. Suykens, J. Vandewalle, and D. Bolle, "Coupled simulated annealing," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 40, no. 2, pp. 320–335, Apr. 2010, doi: 10.1109/TSMCB.2009.2020435.
[45] Y. LeCun and C. Cortes. (2010). *MNIST Handwritten Digit Database*. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[46] A. Krizhevsky, N. Vinod, and H. Geoffrey, "Learning multiple layers of features from tiny images," in *Proc. Krizhevsky LearningML*, 2009, pp. 1–60.
[47] B. Rasti, D. Hong, R. Hang, P. Ghamisi, X. Kang, J. Chanussot, and J. A. Benediktsson, "Feature extraction for hyperspectral imagery: The evolution from shallow to deep (overview and toolbox)," *IEEE Geosci. Remote Sens. Mag.*, early access, Apr. 29, 2020, doi: 10.1109/MGRS.2020.2979764.
[48] R. Hang, Q. Liu, D. Hong, and P. Ghamisi, "Cascaded recurrent neural networks for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 8, pp. 5384–5394, Aug. 2019.
[49] D. Hong, N. Yokoya, and X. X. Zhu, "Learning a robust local manifold representation for hyperspectral dimensionality reduction," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 6, pp. 2960–2975, Jun. 2017, doi: 10.1109/JSTARS.2017.2682189.
[50] D. Hong, N. Yokoya, J. Chanussot, J. Xu, and X. X. Zhu, "Learning to propagate labels on graphs: An iterative multitask regression framework for semi-supervised hyperspectral dimensionality reduction," *ISPRS J. Photogramm. Remote Sens.*, vol. 158, pp. 35–49, Dec. 2019.
[51] X. Junyuan, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. ICML*, 2016, pp. 478–487.
[52] K. G. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5736–5745.
[53] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: An unsupervised and generative approach to clustering," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 1965–1972, doi: 10.24963/ijcai.2017/273.
[54] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5147–5156.
[55] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, vol. 6, pp. 39501–39514, 2018, doi: 10.1109/access.2018.2855437.
[56] X. Peng, J. Feng, J. T. Zhou, Y. Lei, and S. Yan, "Deep subspace clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Feb. 17, 2020, doi: 10.1109/tnnls.2020.2968848.
[57] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[58] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," Cornell Univ., Ithaca, NY, USA, Tech. Rep. abs/1312.6114. [Online]. Available: https://arxiv.org/pdf/1312.6114.pdf

BHABANI SHANKAR PRASAD MISHRA was born in Talcher, Odisha, India, in 1981. He received the B.Tech. degree in computer science and engineering from Biju Pattanaik Technical University, Odisha, in 2003, the M.Tech. degree in computer science and engineering from KIIT University, Bhubaneswar, Odisha, in 2005, and the Ph.D. degree in computer science from F. M. University, Balasore, Odisha, in 2011. In 2013, he did his Postdoctoral Research at the Soft Computing Laboratory, Yonsei University, South Korea. He is currently working as an Associate Professor with the School of Computer Engineering, KIIT University. He has published more than 80 research papers in reputed journals and conferences, has edited more than five books of current importance. His research interests include pattern reorganization, data mining, soft computing, big data, and machine learning. 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 Silver Medal during his M.Tech. for the best Post Graduate in the University.

OM PANDEY is currently pursuing the bachelor's degree with the School of Computer Engineering, KIIT University, Odisha, India. He is also actively working in the field of deep learning for the last two years. He has worked as a Research Assistant under various professors and efficiently carried out the objective on-site and off-site. He is also an active open-source Contributor and have contributed to the python tutorials on the Tensorflow library on tasks, such as Machine Translation. His technical experience includes interning at BitBroker Labs which is a startup at Berkeley SkyDeck.

SATCHIDANANDA DEHURI (Member, IEEE) received the M.Tech. and Ph.D. degrees in computer science from Utkal University, Vani Vihar, Odisha, India, in 2001 and 2006, respectively. He has been working as a Professor of information and communication technology and the Chairman of Post Graduate Council, Fakir Mohan University, Balasore, Odisha, since 2013 and 2019, respectively. His research interests include multi-objective optimization, soft computing, pattern recognition, data warehousing and mining, and bioinformatics. He has already published about 250 research papers in reputed journals and refereed conferences, has published five text books for undergraduate and post graduate students and edited more than 15 books of contemporary relevance. He visited as a BOYSCAST Fellow to the Soft Computing Laboratory, Yonsei University, Seoul, South Korea, under the BOYSCAST Fellowship Program of DST, Government of India, in 2008. In 2010, he received the Young Scientist Award in Engineering and Technology for the year 2008 from Odisha Vigyan Academy, Department of Science and Technology, Government of Odisha.

SUNG-BAE CHO (Senior Member, IEEE) received the Ph.D. degree in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Taejeon, South Korea, in 1993.

He was an Invited Researcher with the Human Information Processing Research Laboratories, Advanced Telecommunications Research (ATR) Institute, Kyoto, Japan, from 1993 to 1995, and a Visiting Scholar at the University of New South Wales, Canberra, ACT, Australia, in 1998. He was also a Visiting Professor with the University of British Columbia, Vancouver, BC, Canada, from 2005 to 2006. Since 1995, he has been a Professor with the Department of Computer Science, Yonsei University. His research interests include neural networks, pattern recognition, intelligent man-machine interfaces, evolutionary computation, and artificial life. He is a member of the Korea Information Science Society, the IEEE Computer Society, the IEEE Systems, Man, and Cybernetics Society, and the Computational Intelligence Society.