symmetry

Article

Intelligent Clustering and Dynamic Incremental Learning to Generate Multi-Codebook Fuzzy Neural Network for Multi-Modal Data Classification

Muhammad Anwar Ma'sum

Faculty of Computer Science, Universitas Indonesia, Kampus UI, Depok, Jawa Barat 16424, Indonesia; muhammad_anwar@cs.ui.ac.id; Tel.: +62-21-7863419 (ext. 3225)

Received: 20 January 2020; Accepted: 17 February 2020; Published: 24 April 2020

Abstract: Classification in multi-modal data is one of the challenges in the machine learning field. The multi-modal data need special treatment as its features are distributed in several areas. This study proposes multi-codebook fuzzy neural networks by using intelligent clustering and dynamic incremental learning for multi-modal data classification. In this study, we utilized intelligent K-means clustering based on anomalous patterns and intelligent K-means clustering based on histogram information. In this study, clustering is used to generate codebook candidates before the training process, while incremental learning is utilized when the condition to generate a new codebook is sufficient. The condition to generate a new codebook in incremental learning is based on the similarity of the winner class and other classes. The proposed method was evaluated in synthetic and benchmark datasets. The experiment results showed that the proposed multi-codebook fuzzy neural networks that use dynamic incremental learning have significant improvements compared to the original fuzzy neural networks. The improvements were 15.65%, 5.31% and 11.42% on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively, for incremental version 1. The incremental learning version 2 improved by 21.08% 4.63%, and 14.35% on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively. The multi-codebook fuzzy neural networks that use intelligent clustering also had significant improvements compared to the original fuzzy neural networks, achieving 23.90%, 2.10%, and 15.02% improvements on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively.

Keywords: multi-modal classification; fuzzy; neural networks; multi-codebook; intelligent clustering; dynamic incremental learning

1. Introduction

Classification is a supervised method in machine learning that is utilized to predict the output class (category) based on the input data. The input data have a set of attributes called features, and the classification method predicts its class label. The classification method has been applied in various areas, e.g., engineering, biology, human science, robotics, financial, business, social science, and in education technology. In the robotics field, machine learning is used as the core of robots' artificial intelligence systems to carry out missions, e.g., path planning mapping and object recognition, [1–3]. One machine learning challenge is that of classification in multi-modal data [4]. The multi-modal condition is a condition where a feature of data is distributed in multiple areas. Multi-modal data come from heterogeneity, both in the natural and human social worlds. The fusion of the data that comes from a heterogeneous source produces multi-modal data. For example, several users in an online shopping website have different preferences for the same choice (item). Another example is the prediction of an election result. Say there are candidates in an election. Candidate A is voted for by residents that are aged below 40 ($0 - 40$), while candidate B is voted for by residents that are aged in the range of 40–60, and candidate C is voted for by residents that are aged below 30 ($0 - 30$) and between 45 and 60 (45–60). We can say that candidate C (class C) has multi-modal data because its feature spreads in two areas, while candidates A and B (class A and class B) have unimodal distribution. Research of multi-modal data analysis has been conducted in various areas, e.g., medical image analysis, video analysis, facial expression, surveillance, and sentiment analysis [5–9]. Other studies have been conducted in the special cases of multi-modal classification, such as real-time and large-scale multi-modal classification [10,11]. Because of their high variability characteristics, multi-modal data need special treatments to learn [12,13]. In some cases, regular classifiers cannot properly fit the data because the data have multi-modality. This causes the performance of classifiers to drop significantly. This study was conducted to solve the challenge. This study was aimed at developing a classification model to handle multi-modal data classification.

In order to classify multi-modal data, a classifier should be improved from its basic structure. The improvement of classifiers can be conducted by using several approaches, e.g., clustering, incremental learning, and ensemble learning [14–17]. In an ensemble approach, the method generates and trains multiples prototypes and then combines them by using some defined rules. The samples of ensemble approaches are voting, weighted voting, and bagging. The other approach to improve classifier structure is that of incremental learning. By using an incremental learning approach, the classifier gradually builds its structure by adding new references (codebooks) during the training process. In the incremental learning method, the classifier generates a new reference if the condition to generate a new codebook is sufficient. Clustering is a method for grouping data. The data are grouped (clustered) into several groups (clusters). After being clustered, the data in each group are used to form classifier references (codebooks) before the training process.

This study is a continuation of previous studies. In a previous study, we proposed a multi-codebook Learning Vector Quantization (LVQ)-based neural networks by using clustering methods [14]. In that study, we used K-means [18], a Gaussian mixture model (GMM) [19], and intelligent K-means based on anomalous pattern clustering [20]. In another study, we tried to search for alternative intelligent clustering for generating multi-codebook neural networks [21]. In the previous study, we proposed a multi-codebook based on an intelligent K-means method based on histogram information. This study is also a continuation of our preliminary study about multi-codebook fuzzy neural networks that use incremental learning [15]. Our preliminary research showed that static incremental learning improves the performance of the neural networks, but, in some cases, its performance is lower than the original version. However, the dynamic thresholds performs better than the static incremental learning and original neural networks. In this study, we propose two types of multi-codebook fuzzy neural networks for multi-modal data classification, one using intelligent clustering and the other using dynamic incremental learning. The first contribution of this study is the further investigation into the multi-codebook neural networks based on intelligent K-means clustering that have been previously proposed [14,21]. The second contribution is the proposal of new versions of multi-codebook neural networks that use dynamic incremental learning, which is hereafter referred to as the multi-codebook fuzzy neural networks that use version 2 of incremental learning. Additionally, we continue the investigation of multi-codebook neural networks by using static incremental learning. Third, we compare multi-codebook neural networks by using intelligent clustering and incremental learning in the synthetic and benchmark datasets. Additionally, we compare the proposed method with existing popular classifiers in the benchmark datasets such as Naïve Bayes, support vector machine (SVM), multi-layer perceptron (MLP), tree bagging, and random forest [22–27]. We additionally compare the proposed method to convolutional neural networks (CNNs) in the benchmark dataset [28]. The last contribution of this paper is the proofing of a significant improvement of the proposed methods from the original neural networks by using ANOVA statistical testing.

Clustering is used to separate the data per class and build codebooks for a class. This approach is simple but can effectively increase the performance of a classifier. An intelligent clustering or non-parametric clustering is a variant of clustering where the algorithm can measure the number of the clusters on its own, while in parametric clustering, the algorithm needs a cluster number as input. In parametric clustering, we need to try several values of cluster numbers to achieve the best result, while in intelligent clustering, we do not need to observe the variation of cluster numbers. Therefore, in this study, we propose intelligent clustering to generate multi-codebook neural networks. An incremental learning method is utilized to enhance neural networks because of the method's adaptive characteristics to the dataset during the training process. The method generates new references during the training process only when necessary. In another hand, the ensemble learning and clustering approaches generate several references before the training process. The incremental learning can be conducted by using static and dynamic thresholds. A threshold is a value that is used in the condition for generating a new codebook. In this study, we propose dynamic thresholds for incremental learning to reduce experimental costs for finding the best thresholds parameter.

In this study, we utilized a single hidden layer fuzzy neural network called fuzzy neuro generalized vector quantization (FNGLVQ) as the base classifier. FNGLVQ was proposed by Setiawan et al. for arrhythmia classification [29]. The original FNGLVQ uses one codebook per class. The codebook of each class is a vector with m elements, where m is the number of features. FNGLVQ has a good performance in a unimodal dataset, although the dataset has high overlapping features. In the other side, the performance of FNGLVQ in a high overlapping multi-modal dataset is low. The other reason for choosing FNGLVQ is its simple structure compared to other popular neural networks such as deep neural networks and MLP. Because of these reasons, this study proposes the development of a multi-codebook FNGLVQ by using intelligent clustering and incremental learning. The multi-codebook FNGLVQ that uses intelligent clustering builds several prototypes (codebooks) before the training process to cover the multi-modal distribution of the features, while the multi-codebook FNGLVQ that uses incremental learning generates a new codebook during the training process when the existing codebook cannot cover the multi-modal distribution of the input data. FNGLVQ is a single hidden layered fuzzy neural network that was developed from generalized learning vector quantization (GLVQ), and GLVQ itself is an enhancement of learning vector quantization (LVQ) version 2.1 that is usually called LVQ2.1. Learning vector quantization (LVQ) is a simple artificial neural networks (ANN) that utilizes the winner take all approach for its update rule [30]. LVQ2.1 is a modification of LVQ that uses a limitation called window for the update process, while GLVQ minimizes errors during the training process [31]. Adding fuzzy a membership function to GLVQ results in FNGLVQ [29].

This paper is organized in several sections. The first section is the introduction of the study. The second section discusses the related works. The third section discusses FNGLVQ, which is used as the base classifier in the proposed method. Section four explains the intelligent clustering dynamic that uses incremental learning. The fifth section discusses the proposed method in detail. Then, the fifth section discusses the experiment results along with analysis. The seventh section is the conclusion of the study and is followed by the future works section.

2. Related Works

Multi-modal data classification has become a challenge in the field of machine learning. Several studies have been conducted to deal with multi-modal data. Corneanu et al. analyzed multi-modal data for facial expression recognition [5]. Their study used red-green-blue (RGB), 3D, and thermal data. Soleymani et al. conducted a survey regarding multi-modality in sentiment analysis problems [6]. Kumar et al. conducted a survey regarding multi-modality and multi-dimensionality in medical imaging analysis [7]. Oskouie studied multimodal feature extraction and fusion for the semantic mining of soccer videos [8]. Zhang et al. study multi-modality in Alzheimer's disease classification [32]. Several studies have been conducted to improve classifiers through the use of incremental learning. Cauwenberghs et al. utilized incremental learning to enhance SVM classifiers [33]. Molina et al. also developed a similar approach for prostatic adenocarcinoma classification [34]. Huang et al. improved extreme learning (ELM) by using incremental learning [35]. On the other hand, several studies have been conducted to improve neural networks by using clustering approaches. In the previous study, Ma'sum et al. utilized K-means, the GMM, and IK-means to enhance LVQ-based neural networks [14]. In another study, Ma'sum et al. combined clustering with unsupervised extreme learning machines to enhance neural network algorithms [36]. The other approach for enhancing classifier performance is by using ensemble learning, as was done by Krawczyk et al. and Ortiz et al. [17,37]. In this study, we only discuss intelligent clustering and incremental learning.

As mentioned before, this study used FNGLVQ as the base classifier. FNGLVQ was developed from GLVQ, and GLVQ was developed from LVQ. LVQ is a simple neural network classifier that uses the winner take all principle [29]. The winner is the class whose reference vector is closest to the input vector. LVQ uses the winner vector as the center of the training/learning and testing processes. LVQ was proposed by Kohonen et al. as the supervised version of the self-organizing map (SOM). LVQ has several variants, e.g., LVQ1, LVQ2, LVQ2.1, and LVQ3. The variants of LVQ were developed by using a limitation called the window for updating conditions in the learning process. LVQ2.1 was then enhanced by adding optimization procedures, resulting in GLVQ [30]. The optimization is applied during the training process. Therefore, the error rate is minimized from iteration to iteration. In many cases, the performance of GLVQ has been shown to be superior to all variants of LVQ. GLVQ was modified by Setiawan et al. by using a fuzzy membership function [28]. FNGLVQ was originally used for arrhythmia classification based on electrocardiogram (ECG) signals. The other development of the LVQ classifier was conducted by Kusumoputro et al., who used the fuzzy principle in LVQ, resulting in fuzzy neuro learning vector quantization (FNLVQ) [38]. FNLVQ was optimized by using particle swarm optimization (PSO), resulting in FNLVQ-PSO [39]. Another study enhanced GLVQ by engineering its adaptation property as the integration of feature extraction and classification algorithm [40].

A popular development of neural networks classifiers has been in the deep learning area. Many deep learning methods have developed from convolutional neural networks. One popular algorithm is that of the deep convolutional neural network (deep CNN) [41]. The other popular model of CNNs is that of the very deep convolutional neural networks that are as popular as the Visual Geometry Group (VGG) model [28]. The other notable development of CNNs is that of densely connected convolutional networks (DenseNet) [42], residual neural networks (ResNet) [43], neural architecture search network (NASNet) [44], and MobileNet [45]. All the above-mentioned methods have been developed for image classification. Many of the methods mentioned earlier require a 32×32 image as input, and the others require a bigger sized image as input. As for the comparison in this study, we used the modified version of the VGG model that only uses the first block of its convolution blocks. The above-mentioned methods did not handle the data used in this study, as many of the data have a far smaller number of features.

3. Fuzzy-Neuro Generalized Learning Vector Quantization (FNGLVQ)

Fuzzy-neuro generalized learning vector quantization (FNGLVQ) is an improvement of generalized learning vector quantization (GLVQ) by adding a fuzzy membership function. The fuzzy membership function can be used to handle overlapping data. FNGLVQ was proposed by Setiawan et al. for heart disease classification based on ECG signals [3]. GLVQ improves upon LVQ2.1 (a variant of LVQ) by adding error minimization during the training process. LVQ is a simple neural network that uses a winner take all approach for the learning process. The algorithm defines the winner class (w_1) as the class in which its weight (codebook/reference vector) is the closest to the input vector. During the training process, the weight of the winner vector is updated. If the winner class is the same as the input vector, then its weight is adjusted closer to the input vector. Otherwise, its weight is adjusted further from the input vector. LVQ2.1 modifies LVQ by adding an update rule for the runner up class. In the LVQ2.1 algorithm, the runner up class (w_2) is a class whose weight is closest to input vector but comes from different class labels. GLVQ and FNGLVQ are enhancements of LVQ2.1 that use the same approach. These algorithms use the winner and runner up classes in their training processes. During the training process, the methods minimize misclassification errors.

The FNGLVQ architecture is illustrated in Figure 1. The figure shows that the vector x is an input for the algorithm. In FNGLVQ, each class has a weight, as illustrated by triangle shape in the figure. Each class has n weights (w_1, w_2, w_3, . . . , w_n) where n is the number of features of the dataset. Please note that w_1 and w_2 in this context are the weights of features 1 and 2 of their class, not the symbols of the winner and runner up vector, as discussed before. Each weight has three values ($w_{i\ min}$, $w_{i\ mean}$, $w_{i\ max}$) because the method uses fuzzy membership. In the learning process, the method computes the distance between input vectors and the weights of each class. As the data have n features, the measurement of the distances is conducted for all features, and then the mean of the distances is calculated. Then, the method finds the winner and runner up classes based on the mean distance from the classes to the input vector. Last, the method updates the weight of the winner and runner up classes. The GLVQ classifier defines misclassification error as written in the equation below:

$$\varphi(x) = \frac{d_1 - d_2}{d_1 + d_2} \tag{1}$$

where d_1 is the distance of the input class and winner class and d_2 is the distance of the input and the runner up classes. FNGLVQ adapts the approach from GLVQ. In the training process, the FNGLVQ method uses a similarity value instead of distance to determine the winner and runner-up vectors. The distance is defined as $d = 1 - \mu$. Substituting the distance equation into the misclassification error modeled in GLVQ results in the equation below:

$$\varphi(x) = \frac{\mu_2 - \mu_1}{2 - \mu_2 - \mu_1} \tag{2}$$

where μ_1 is the similarity value between the input sample (input vector) and the winner vector, whereas μ_2 is the similarity between the input sample (input vector) and the runner up vector. The winner vector is the closest existing reference vector (codebook) from the same class $C_x = C_{w1}$, whereas the runner up vector is the closest existing reference vector from a different class $C_x = C_{w2}$. The similarity value is computed by the equation below:

$$\mu = h(x, w_{min}, w_{mean}, w_{max}) = \begin{cases} 0, & if\ x \le w_{min} \\ \frac{x - w_{min}}{w_{mean} - w_{min}}, & if\ w_{min} \le x \le w_{mean} \\ \frac{w_{max} - x}{w_{max} - w_{mean}}, & if\ w_{mean} \le x \le w_{max} \\ 0, & if\ x \ge w_{max} \end{cases} \tag{3}$$

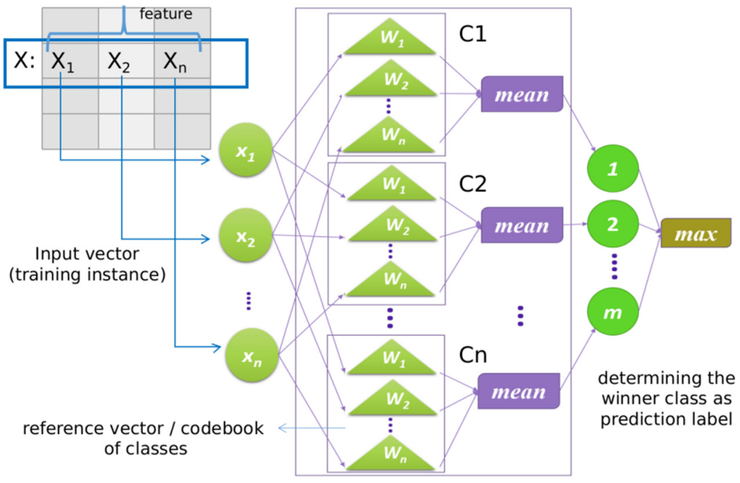


Figure 1. Fuzzy-neuro generalized learning vector quantization (FNGLVQ) architecture.

FNGLVQ uses a cost function, as defined in GLVQ, to minimize misclassification errors during the training process. The cost function (S) is defined in the equation below:

$$S = \sum_{n=1}^{N} f(\varphi(x)) \tag{4}$$

where f is monotonically increasing function and N is the number of iterations (epoch) in the training process. GLVQ uses the sigmoid function for f. In the training process, FNGLVQ uses the steepest descent method used in GLVQ, as defined in the equation below:

$$w_i(t+1) \leftarrow w_i(t) - \alpha \frac{\delta s}{\delta w_i(t)}, \text{ where } i = 1,2 \tag{5}$$

The derivation of the cost function (S) to the weight (w) is defined by the equation below:

$$\frac{\delta s}{\delta w_i} = \frac{\delta s}{\delta \varphi} . \frac{\delta \varphi}{\delta \mu} . \frac{\delta \mu}{\delta w_i} \tag{6}$$

The FNGLVQ method uses a triangular function for its reference vectors (codebooks). Therefore, each of the reference vectors (weights) has three values (w_{min}, w_{mean}, w_{max}). The derivation of the update rule to (w_{mean}) is divided into three conditions and leads to the FNGLVQ learning (training) formula, as follows:

- If $w_{min} < x \le w_{mean}$

$$w_1(t+1) \leftarrow w_1(t) - \alpha . \frac{\delta f}{\delta \varphi} . \frac{2.(1-\mu_2)}{(2-\mu_1-\mu_2)^2} . \left(\frac{x - w_{min}}{(w_{mean} - w_{min})^2} \right) \tag{7}$$

$$w_2(t+1) \leftarrow w_2(t) + \alpha . \frac{\delta f}{\delta \varphi} . \frac{2.(1-\mu_1)}{(2-\mu_1-\mu_2)^2} . \left(\frac{x - w_{min}}{(w_{mean} - w_{min})^2} \right) \tag{8}$$

- If $w_{mean} < x < w_{max}$

$$w_1(t+1) \leftarrow w_1(t) + \alpha . \frac{\delta f}{\delta \varphi} . \frac{2.(1-\mu_2)}{(2-\mu_1-\mu_2)^2} . \left(\frac{w_{max} - x}{(w_{max} - w_{mean})^2} \right) \tag{9}$$

$$w_2(t+1) \leftarrow w_2(t) - \alpha . \frac{\delta f}{\delta \varphi} \text{x} \frac{2.(1-\mu_1)}{(2-\mu_1-\mu_2)^2} . \left(\frac{w_{max} - x}{(w_{max} - w_{mean})^2} \right) \tag{10}$$

- If $x \le w_{min}$ and $x \ge w_{max}$

$$w_i(t+1) \leftarrow w_i(t), i = 1,2 \tag{11}$$

where w_1 (winner class) is the closest reference vector (codebook) from the same class as the input vector $C_x = C_{w1}$, and w_2 (runner up class) is the closest reference vector (codebook) from a different class. The update rules for the other two values in fuzzy membership (w_{min} and w_{max}) are conducted by using the equation below:

$$w_{min} \leftarrow w_{mean}(t+1) - (w_{mean}(t) - w_{min}(t)) \tag{12}$$

$$w_{max} \leftarrow w_{mean}(t+1) - (w_{mean}(t) - w_{min}(t)) \tag{13}$$

The symbol α represents the learning rate that has a value between 0 and 1. In FNGLVQ, the learning rate value decreases along with the iteration number (t). The update of the learning rate value follows in the equation below:

$$\alpha(t+1) = \alpha_0.\left(1 - \frac{t}{t_{max}}\right) \tag{14}$$

FNGLVQ defines additional rules to adjust w_{min} and w_{max} to gain better performance as follows. If (μ1 > 0 or μ2 > 0) and $\varphi < 0$, then the method increases fuzzy triangular width by using the equation below:

$$w_{min} \leftarrow w_{mean} - (w_{mean} - w_{min}).(1 + (\beta.\alpha)) \tag{15}$$

$$w_{max} \leftarrow w_{mean} + (w_{max} - w_{mean}).(1 + (\beta.\alpha)) \tag{16}$$

If input data are predicted into the wrong class ($\varphi \geq 0$), then the method decreases the triangular width by using the equation below:

$$w_{min} \leftarrow w_{mean} - (w_{mean} - w_{min}).(1 - (\beta.\alpha)) \tag{17}$$

$$w_{max} \leftarrow w_{mean} + (w_{max} - w_{mean}).(1 - (\beta.\alpha)) \tag{18}$$

If μ1 = 0 and μ2 = 0, then the width of fuzzy triangular vectors must be increased by using equation below (the ɣ. is a constant):

$$w_{min} \leftarrow w_{mean} - (w_{mean} - w_{min}).(1 - (\text{ɣ}.\alpha)) \tag{19}$$

$$w_{max} \leftarrow w_{mean} + (w_{max} - w_{mean}).(1 + (\text{ɣ}.\alpha)) \tag{20}$$

The training process of FNGLVQ is conducted with the following steps:

1. Given that the training set consist of m record instances (X1, X2, . . . , Xm), each instance is a vector of n elements because the data have n features.
2. Initiate the codebook (reference vector/weight) of each class (w) by a random selection of training set for the respective class.
3. For each instance of the training data, train the weights of the classes by using Equation (3) and Equations (7)–(20)
4. Repeat step 2 until N number of iterations (epoch)

The testing process of the FNGLVQ algorithm is simply conducted by measuring the similarities between the input vector and the codebooks of the classes. Then, the method takes the most similar class as the prediction.

4. Intelligent Clustering and Dynamic Incremental Learning

4.1. Intelligent Clustering

Intelligent clustering is a non-parametric clustering method for which the user does not need to input the number of clusters. One of the benefits of intelligent clustering is reducing the experiment cost to find the most suitable cluster number. A popular intelligent clustering method is intelligent K-means clustering based on anomalous patterns. Another version is intelligent clustering based on histogram information, which utilized in a previous study [21].

4.1.1. Intelligent K-means Based on Anomalous Pattern

Intelligent K-means (IK-means) based on anomalous patterns is a fully non-parametric-clustering method that uses the farthest point, called the anomalous pattern, to generate a new cluster. The method was proposed by Mirkin et al. as the smart choice for choosing the K in K-means clustering [20]. This algorithm is the development of a standard K-means standard with no parameter for the number of clusters. The idea in this algorithm is to cluster data in a feature space by using anomalous points to build new clusters. An anomalous point is the farthest point from the initial reference on the dataset. The procedure of intelligent K-means clustering based on anomalous patterns has many steps. The first step of the method is calculating the reference point (RP). The RP is the center of the feature space of the whole data. The second step is calculating the furthest point of the data from the RP, which is called the anomalous point (AP). The third procedure is a grouping step that is conducted by using the RP and the AP as the group's centers (centroids). The grouping process is conducted by using standard K-means clustering until convergence. Therefore, the data are separated into two groups—the group with the RP as the centroid and a new group formed from the anomalous pattern. Please note that the new group formed from anomalous patterns may have a different centroid from its original centroid that was taken from the anomalous point.

The separation of an anomalous pattern from its original dataset is shown in Figure 2. This process is iterated until the stopping condition is achieved. The condition is when all data in feature space are clustered, as notated by St = Et. This means the last group of the data is clustered, and there is no group of data that needs to be clustered. The procedure of intelligent K-means based on anomalous patterns can be conducted by using Algorithm 1.

Algorithm 1: Intelligent K-Means Clustering based on Anomalous Pattern

```
01:  procedure IK-MEANS-AnomalousPattern
02:      setting:
03:      t = 1, $E_t$ = original data set on feature space
04:      denote R as the thresholds for small cluster removal
05:      find Anomalous pattern:
06:      apply procedure Find Anomalous Pattern for each t or at t = 1 (recommended)
07:      Control statement:
08:      If $S_t$ != $E_t$ then//There is a possibility to build new cluster
09:          $E_t = E_t - S_t$
10:          t += 1
11:          go to step setting (line 02):
12:      small cluster removal:
13:      for i = 1:T do//T is the number of clusters
14:          If $|C_i| < R$ then
15:              remove $C_i$
16:      end for
17:      Denote the remaining cluster $C_1, C_2, C_3, \ldots .C_n$, and their centroids $c_1, c_2, c_3, \ldots .c_n$
18:      for i = 1:T' do //T' is the number of the remaining cluster
19:          do K-Means for $K_i$ with $c_i$ as initial seed
20:      end for
21:
22:  procedure Find Anomalous Pattern
23:      Preprocessing step:
24:      Denote the reference point $x = x_1, x_2, x_3, \ldots .x_n$
25:      Normalize the original data following the defined standard//This step is not mandatory
26:      initial setup:
27:      find most distant point k as the tentative centroid
28:      cluster update:
29:      determine cluster list L around k against the only other centroid,
  $y_t$ is assigned to $S$ if $d(y_t, c_t) < d(y_t, c_0)$
30:      centroid update:
31:      Calculate (k'), the mean value within L
32:      If k' ! = k then
33:          k = k'
34:          go to step cluster update
35:      else
36:          go to step output
37:      output:
38:      return list L and its centroid k as the anomalous pattern
```

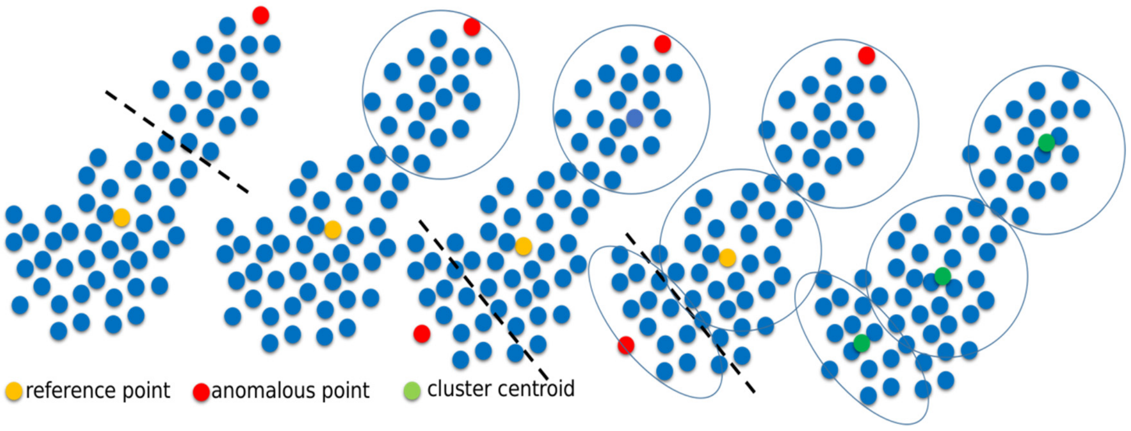


Figure 2. Intelligent K-means based on anomalous pattern.

4.1.2. Intelligent K-Means Based on Histogram Information

One of the variants of intelligent K-means is intelligent K-means based on histogram information (IK-means). This is an intelligent clustering that uses the histogram curve of the data to define the number of clusters. The algorithm is illustrated in Figure 3. As shown in Figure 3, the method has several steps. The first step is the analysis of the feature space. Then, the method draws a histogram for each feature of the data and approximates the number of peaks of the histogram. Then, the method computes the maximum number of peaks of the feature histograms and uses it as the cluster number. In the last step, the method conducts K-means clustering by using a given number of clusters. The procedure of intelligent K-means clustering based on histogram information is written in Algorithm 2.

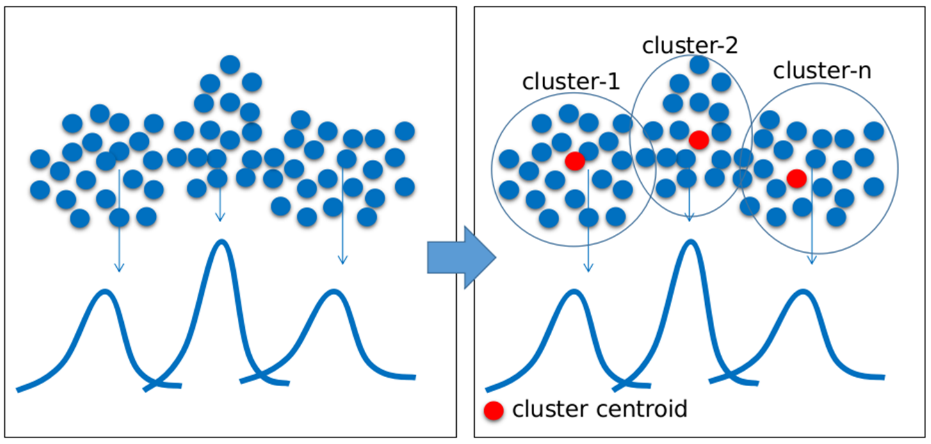


Figure 3. Intelligent K-means based on histogram information.

Algorithm 2 shows that the clustering algorithm analyzes each feature of the dataset. Then, for each feature, the algorithm draws the histogram curve of the feature. The algorithm then counts the number of peaks of the curve. To count the peaks, the method uses a gradient from an edge to the next edge. If the gradient change is from positive to negative, then a peak is detected. The method analyzes all features of the data and finds the maximum peaks. Then, the method conducts the K-means clustering by using maximum peaks as the number of clusters.

Algorithm 2: Intelligent K-Means Clustering based on Histogram Information

```
01: procedure IK-MEANS-HistogramInformation
02:     denote $f_1, f_2, f_3, \ldots .f_m$ as data features
03:     maxPeak = 0 as the current maximum peak
04:     for c = 1:m do
05:         $peak_i$ = Approximate Histogram Peak ($f_i$)
06:         If $peak_i$ > maxPeak then
07:             maxPeak = $peak_i$
08:     end for
09:     do K-Means clustering with K = maxPeak
10:
11: procedure Approximate Histogram Peak (feature)
12:     sign1 = 0, sign2 = 0, numPeak = 0
13:     histVal = histogram(feature)//array of histogram values
14:     for i = 2: lengthof(histVal)-1 do
15:         sign2 = sign1
16:         If histVal(i) > histVal(i − 1) then
17:             sign1 = 1
18:         else If histVal(i) < histVal(i − 1) then
19:             sign1 = −1
20:         else
21:             sign1 = 0
22:         if sign2 == 1 and sign1 == 1 then
23:             numPeak = numPeak + 1
24:     end for
25:     return numPeak
```

4.2. Dynamic Incremental Learning

Incremental learning is a learning methods that gradually builds or improves a classifier's structure during the training process. Different from the clustering approach, incremental learning is conducted by various approaches. For example, in GLVQ, incremental learning is used to optimize neural networks [31]. Another example of incremental learning is the generation of a new reference vector when necessary, as was done by Losing et al. [46]. The other example of incremental learning is the development of a sequential structure for neural networks and training the model by using chunks of data, as done by Liang et al. [47]. Another example of incremental learning is the expansions of the classifier when there are sufficient samples in the training state [48]. In summary, the process of incremental learning is the adjustment of the structure of neural networks. The adjustment process can be a modification, the addition of a reference vector (codebook), or the deletion of the existing codebook. In neural networks, incremental learning can be applied to add/remove layers, generate/delete a new reference vector (codebook) within a layer, add/delete neurons of a codebook within a layer, and any other operations necessary. The action takes place when the current states of codebooks and the input sample meet the defined condition.

This study used the generating/inserting a new codebook approach, as used by Losing et al. During the training process, the FNGLVQ classifier checks whether it is necessary to generate a new codebook. If yes, then the classifier generates a new codebook. Some of the conditions to generate a new codebook are:

1. Too low similarity value between the input sample and closest codebook from the same class.
2. The similarity between the input sample and the most distant codebook from the different classes is higher than the similarity between the input sample and the closest codebook from the same class.

During the training process, the classifiers are also able to delete the existing codebook. Some of the conditions for deleting an existing codebook are:

1. Too high similarity value between the input sample and the closest codebook from different classes.
2. The similarity between the input sample and the most distant reference vector from the same class is less than the similarity between input sample and closest codebook from different classes.

5. Proposed Method: Multi-Codebook Fuzzy Neuro Generalized Learning Vector Quantization (Multi-codebook FNGLVQ)

5.1. The Problem, Motivation, and Idea

As mentioned in the introduction section, multi-modal data are data who features are spread in several areas. An illustration of multimodal-data is shown in Figure 4. Look at the *X*-axis in Figure 4. The figure shows the scatter plot and distribution of two classes—A (orange) and B (green). The figure shows that the features of class A are distributed in two areas, while the features of class B are distributed only in one area. We can say that class A has a multi-modal distribution, while class B has a unimodal distribution. The figure also shows that the features of class B are distributed in the area between the class A areas. To fit the data distribution, the fuzzy neural networks generate a fuzzy reference vector (codebook), as illustrated by the triangle. Figure 4 demonstrates the two approaches of fitting: by using a a single codebook (single reference vector), which is shown in the upper image, and a multi-codebook (multiple reference vectors), which is shown in the bottom image. The figure shows that the fitting done by using a single codebook produces a high overlap between class A and class B, while fitting by using a multi-codebook produces less overlapping between class A and class B. Therefore, the multi-codebook approach fits the multi-modal data better than the single-codebook approach. The following paragraph continues the discussion between a single-codebook and a multi-codebook fitting by using mathematical and numerical analyses.

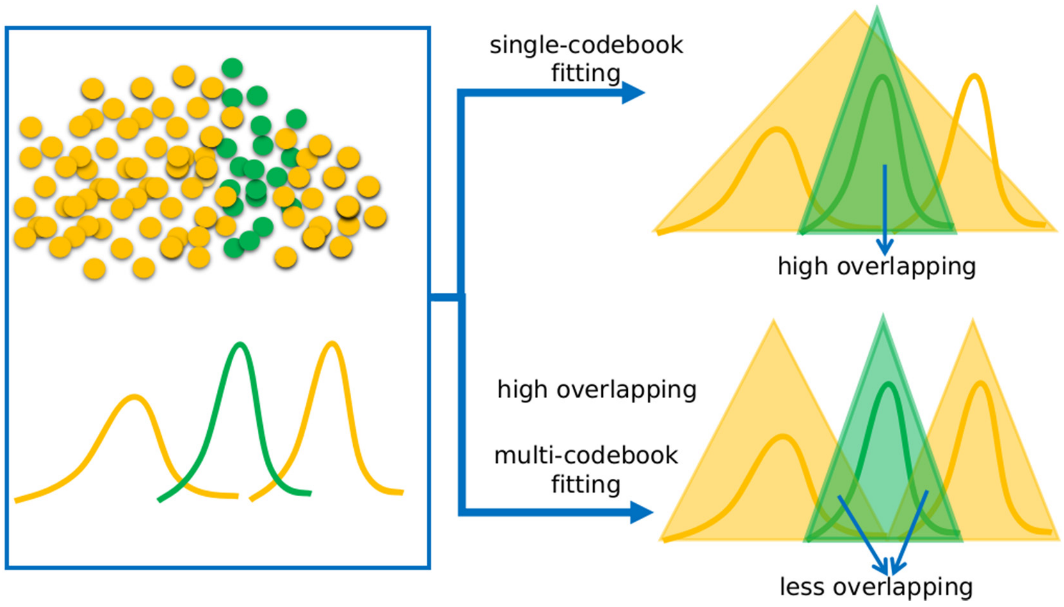


Figure 4. Single-codebook vs. multi-codebook for multi-modal data fitting.

Figures 4 and 5 show that the overlapping of class A and class B when using a multi-codebook is less than the overlapping of A and class B when using a single-codebook. In a single-codebook, the codebook of class A is illustrated by triangle 1, whereas the codebook of class B is illustrated by triangle 2, and the similarity (overlapping) between the classes is noted as h_{12}. Each fuzzy triangle has three values (min, mean, and max), e.g., triangle 1 has a_1, b_1, and c_1 representing its min, mean, and max. FNGLVQ uses a balance triangle so that c1–b1 is the same as b1–a1. In multi-codebook approach, triangle 1 is substituted by triangles 3 and 4, and the similarity between classes is noted as h_{23} and h_{24}. Since there are two codebooks, the worst case overlapping value is max (h_{23}, h_{24}) The figure shows that h_{23} and h_{24} are less than h_{12}. By using the fuzzy membership function, as used in FNLVQ [37], we could measure h_{12}, h_{23}, and h_{24}. In the triangle 1 perspective, we could compute h_{12} as follows:

$$\frac{c_1 - x}{c_1 - b_1} = \frac{h_{12}}{1} \tag{21}$$

$$c_1 - x = h_{12}(c_1 - b_1) \tag{22}$$

$$x = c_1 - h_{12}(c_1 - b_1) = c_1 - h_{12}l_1 \tag{23}$$

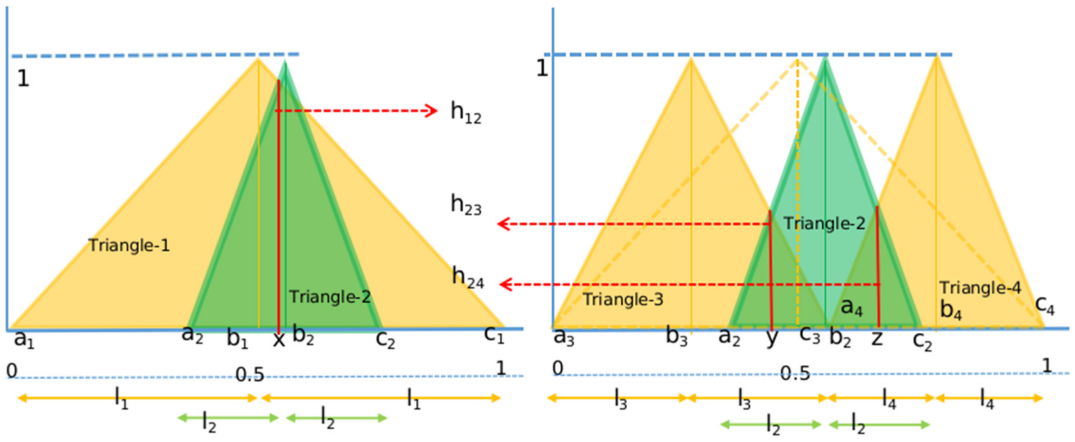


Figure 5. The similarity between classes in single-codebook vs. multi-codebook for multi-modal data fitting.

In the triangle 2 perspective, we could compute h_{12} as follows:

$$\frac{x - a_2}{b_2 - a_2} = \frac{h_{12}}{1} \tag{24}$$

$$x - a_2 = h_{12}(b_2 - a_2) \tag{25}$$

$$x = a_2 + h_{12}(b_2 - a_2) = a_2 + h_{12}l_2 \tag{26}$$

By combining Equations (23) and (25) for same x, we could compute h_{12} as follows:

$$x - a_2 = h_{12}(b_2 - a_2) \tag{27}$$

$$c_1 - h_{12}l_1 = a_2 + h_{12}l_2 \tag{28}$$

$$h_{12}(l_1 + l_2) = c_1 - a_2 \tag{29}$$

$$h_{12} = \frac{c_1 - a_2}{l_1 + l_2} \tag{30}$$

By using the same method, we could obtain the values of h_{23} and h_{24} as follows:

$$h_{23} = \frac{c_3 - a_2}{l_3 + l_2} \tag{31}$$

$$h_{24} = \frac{c_2 - a_4}{l_2 + l_4} \tag{32}$$

Assuming that we used a normalized feature space, for the case above we used the following conditions: $a_1 = 0$, $c_1 = 1$, $l_1 = 0.5$, $0 < a_2 < 0.5$, $0.5 < c_2 < 1$, $0.5 < c_3 < 1$, $a_4 = c_3$, $a_3 = a_1$, $c_4 = c_1$, $l_4+l_3 = l_1 = 0.5$, $c_3 < c_2$, $0 < l_2 < 0.5$, and $0 < l_3 < 0.5$. Table 1 shows several probable values of the triangles that result in values of h_{12}, h_{23}, and h_{24} that satisfy the conditions above. Table 1 shows that the value of h_{23} and h_{24} were less than h_{12}. The table supports visual description in Figure 5 that by using the multi-codebook approach, the similarity (overlapping) between class A and class B in FNGLVQ was less than the similarity (overlapping) between class A and class B when using single-codebook approach.

Table 1. Comparison of Overlapping Values between the Single-Codebook and Multi-codebook Approaches Given the Triangles Parameters.

a_2	c_2	l_2	$c_3 = a_4$	l_3	$l_4 = 0.5 - l_3$	h_{12}	h_{23}	h_{24}
0.4	0.9	0.25	0.8	0.4	0.1	0.80	0.62	0.29
0.3	0.9	0.3	0.7	0.3	0.2	0.88	0.67	0.40
0.25	0.9	0.325	0.6	0.2	0.3	0.91	0.67	0.48
0.4	0.8	0.2	0.7	0.4	0.1	0.86	0.50	0.33
0.3	0.8	0.25	0.6	0.3	0.2	0.93	0.55	0.44
0.25	0.8	0.275	0.5	0.2	0.3	0.97	0.53	0.52
0.4	0.75	0.175	0.65	0.4	0.1	0.89	0.43	0.36
0.3	0.75	0.225	0.6	0.3	0.2	0.97	0.57	0.35
0.25	0.75	0.25	0.55	0.2	0.3	1.00	0.67	0.36

Because of the motivation and analysis above, in this study, we propose multi-codebook FNGLVQ. In this paper, we propose two types of a multi-codebook FNGLVQ—one that uses intelligent clustering and another type that uses dynamic incremental learning. The use of the multi-codebook in fuzzy neural networks is aimed at properly fitting the multi-modal distribution of the dataset. Intelligent clustering is utilized to produce codebook instances before the training process, while dynamic incremental learning is used to build codebooks during the training process. In a multi-codebook method that uses intelligent clustering, codebooks are generated by using intelligent clustering before the training process, and then the codebooks are updated during the training process. The motivation of using intelligent clustering is to minimize the parameter tuning of the classifier. By using intelligent clustering, the method does not need the input cluster number. Therefore, in the experiment, there is no need to find the best cluster number. The intelligent clustering methods in this study are the intelligent K-means clustering based on anomalous patterns and intelligent K-means based on histogram information.

The second type of the proposed method is multi-codebook FNGLVQ that uses dynamic incremental learning. This is the type of the proposed multi-codebook fuzzy neural networks that gradually builds its structure during the training (learning) process. When the condition to generate a new codebook is satisfied, the method generates a new codebook. Similar to the intelligent clustering case, the motivation of using dynamic incremental learning is to minimize parameter tuning in the experiment. By using dynamic incremental learning, there is no need to find the best thresholds for the condition to generate a new codebook. Additionally, our preliminary research showed that incremental learning with dynamic thresholds shows better performance compared to incremental learning with a static thresholds [15].

5.2. Architecture

The architecture of the multi-codebook FNGLVQ is similar to the architecture of the original FNGLVQ. As shown in Figure 6, the multi-codebook FNGLVQ has three layers: the input layer, the middle (hidden) layer, and the output layer. The input layer acquires the input sample for the classifier, the middle layer contains the reference vectors (codebooks) for each class, and the output layer determines the winner vector. The main difference between the multi-codebook FNGLVQ and the original FNGLVQ is in its middle layer. The original FNGLVQ has one codebook for each class, while the multi-codebook FNGLVQ has several codebooks for each class. The number of codebooks between a class and another class may differ. For example, class A and class B have three codebooks, but class C has only two codebooks.

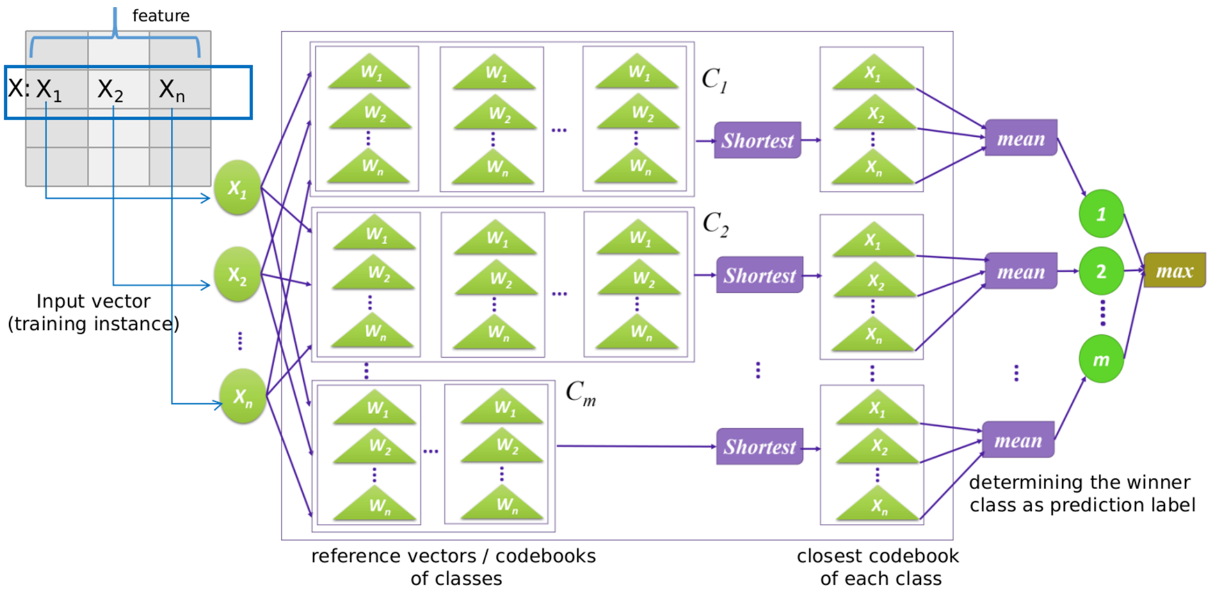

feature
X: X1 X2 Xn
Input vector
(training instance)
C1
C2
Cm
Shortest
mean
max
determining the winner class as prediction label
reference vectors / codebooks of classes
closest codebook of each class

Figure 6. Architecture of FNLGVQ.

In this study, we propose two types of multi-codebook FNGLVQ methods. The first type is multi-codebook FNGLVQ that uses intelligent clustering. This type has two versions; the first version is a multi-codebook FNGLVQ that uses intelligent K-means clustering based on anomalous patterns, and the second version is multi-codebook FNGLVQ that uses intelligent K-means clustering based on histogram information. The multi-codebook FNGLVQ that uses dynamic incremental learning also has two versions, and we call them the multi-codebook FNGLVQ that uses dynamic incremental learning 1 and the multi-codebook FNGLVQ that uses dynamic incremental learning 2. Those two versions have different conditions for generating a new codebook. Detailed explanations for proposed methods are discussed in the following sub-section.

5.3. Proposed Method: Multi-codebook Fuzzy Neuro Generalized Learning Vector Quantization by Using Intelligent Clustering

As mentioned before, the first type of the proposed method is the multi-codebook FNGLVQ that uses intelligent clustering. In this type, codebooks are generated by using intelligent clustering. The idea of this method is to divide the data into several groups by using the clustering method. Then, each group is approximated by a reference vector (codebook) of the FNGLVQ classifier. Therefore, multiple codebooks are expected to fit the multiple groups of the data. In other words, the multiple codebooks are expected to cover the multi-modality of the data. The clustering methods utilized in this study are intelligent K-means based on anomalous patterns and K-means based on histogram information. As mentioned before, the motivation of using intelligent clustering is to minimize the parameter tuning during the experiment, as the intelligent clustering does not need the cluster number as the input. The clustering method automatically determines the best cluster number.

There multi-codebook FNGLVQ that uses intelligent clustering has several steps. The first step is dividing the dataset based on class label. The second step is the clustering of the data of each class into a number cluster. Say that the number of clusters is C: In intelligent clustering, the value of C is automatically calculated by the method, not given manually. The next step is the pruning of the small cluster. The clusters with small members are removed from groups. However, the pruning process is optional due to the preference. The next step is the codebook initiation. The method generates a codebook from each cluster. The method then calculates the min, mean, and max values from the instances in each cluster. The process is called fuzzification. The tuple (min, mean, and max) then becomes a codebook for its class in FNGLVQ. If a class has C clusters, then it has C codebooks that are generated before training. Then, the method performs the training process by using the FNGLVQ training procedure. The training process of the multi-codebook FNGLVQ that uses intelligent clustering is conducted with the procedure shown in Algorithm 3.

Algorithm 3: Multi-codebook Fuzzy Neuro Generalized Vector Quantization By Using Intelligent Clustering

```
01: procedure MC-FNGLVQ-CLUSTERING
02:     denote $x_1, x_2, x_3, \ldots .x_n$ as instances
03:     denote $f_1, f_2, f_3, \ldots .f_m$ as data features
04:     denote $c_1, c_2, c_3, \ldots .c_y$ as class labels
05:     for $c_i = c_1{:}c_y$ do
06:         denote C as number of clusters
07:         denote $f_c = f_1{:}f_m$ where $c_c = c_i$
08:         do clustering to $f_c$, where C is a number of clusters as the result of intelligent clustering
09:         for j = 1:C do//C = number of cluster
10:             denote $C_j$ as member of cluster-j
11:             find min, mean, meax of $C_j$
12:             Use the min, mean, max to generate FNGLVQ codebook
13:         end for
14:     end for
15:     for $x_i = x_1{:}x_n$ do
16:         train x by using FNGLVQ method
17:     end for
18      Repeat step 15–17 until the number of iteration (epoch) is satisfied
```

5.4. Proposed Method: Multi-Codebook FNGLVQ by Using Dynamic Incremental Learning

As stated in the introduction section, the second type of the proposed method is the multi-codebook FNGLVQ that uses dynamic incremental learning. Different from the previous type of multi-codebook methods, in the multi-codebook FNGLVQ that uses incremental learning, the reference vector (codebook) of each class is gradually adjusted in the training process. Before training, the method generates one reference vector per class. The generation of the initial codebook is the same as in the original FNGLVQ. The initial codebook can be generated by using all instances or samplings of the data. Then, in the training process, its codebook is gradually built (adjusted). The adjustment is conducted if the condition to generate a new codebook in respect to the current input data is sufficient.

In this paper, we developed two variants of multi-codebook FNGLVQ methods that use static incremental learning and dynamic incremental learning. In the static version, the condition to generate a new codebook is set by using an input thresholds, while in the dynamic version, the condition to generate a new codebook is set by using variables whose values change during the training process. In this study, we propose two versions of multi-codebook methods that use dynamic incremental learning, and these are named multi-codebook that uses dynamic incremental learning 1 and multi-codebook that uses dynamic incremental learning 2. Presented here are the conditions for the static and dynamic versions of the proposed methods.

1. Multi-codebook FNGLVQ that uses static incremental learning: In this version, a new codebook is generated if the similarity of the winner vector is less than the given thresholds. The winner vector is the nearest codebook/reference vector to the class that is the same as the input vector class. The condition is written in the equation below, and the threshold is a constant that is defined before training:

$$\mu_1 < \mathit{threshold} \tag{33}$$

2. Multi-codebook FNGLVQ that uses dynamic incremental learning 1: In this version, a new codebook is generated if the similarity of the input vector to its codebooks is less than the minimum similarity of the input vector to the codebooks of other classes. The condition is written in the equation below:

$$\mu_1 < \mu_{i\ min},\ \mu_{i\ min} = \min(\mu_i),\ \textit{i is the class label},\ i \neq \textit{input class label} \tag{34}$$

3. Multi-codebook FNGLVQ that uses dynamic incremental learning 2: In this version, a new codebook is generated if the similarity of the input vector to its codebooks is less than the mean similarity of the input vector to the codebooks of other classes. The condition is written in the equation below:

$$\mu_1 < \mu_{i\ mean},\ \mu_{i\ mean} = \text{mean}(\mu_i),\ \textit{i is class label},\ i \neq \textit{input class label} \tag{35}$$

There is no codebook removal in the proposed multi-codebook FNGLVQ. It is applied to avoid a create-and-delete loop. The flow multi-codebook FLGLVQ that uses dynamic incremental learning is shown in Algorithm 4:

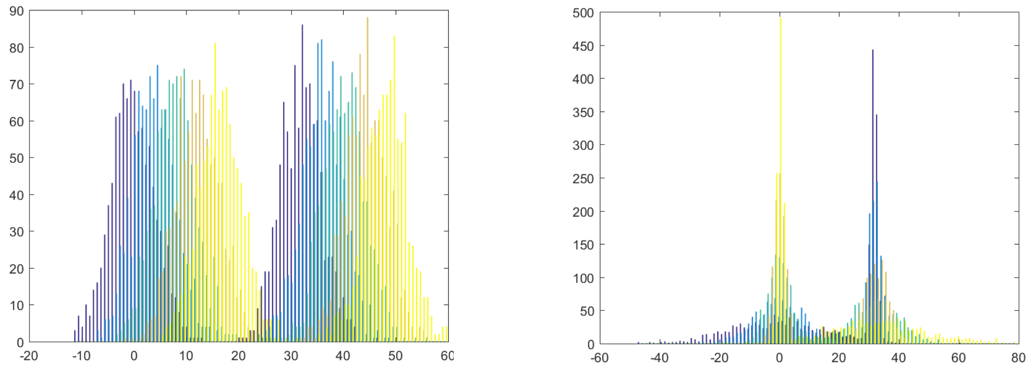
Algorithm 4: Multi Codebook Fuzzy Neuro Generalized Vector Quantization Using Incremental Learning

```
01: procedure MC-FNGLVQ-CLUSTERING
02:     denote $x_1, x_2, x_3, \ldots .x_n$ as instances
03:     denote $w_1, w_2, w_3, \ldots .w_p$ as reference vector of class 1,2, … p
04:     initiate $w_1, w_2, w_3, \ldots .w_p$
05:     denote T as the condition of generating a new codebook as defined in eq 34 or 35//eq 33 for the static version
06:     for $x_i = x_1{:}x_n$ do
07:         denote y as the class label of x
08:         compute $\mu$ of x using equation 3, compute the similarity of x to codebooks of all classes
09:         if $\mu$ does not satisfy T then//T can be equation 34 or 35
10:             train x using FNGLVQ method
11:         else
12:             state $w_y$new as new codebook for class y
13:             $w_y new_{mean} = x$
14:             $w_y new_{min} = x - \text{average}(w_{ymean} - w_{ymin})$
15:             $w_y new_{max} = x - \text{average}(w_{ymax} - w_{ymean})$
16:     end for
```

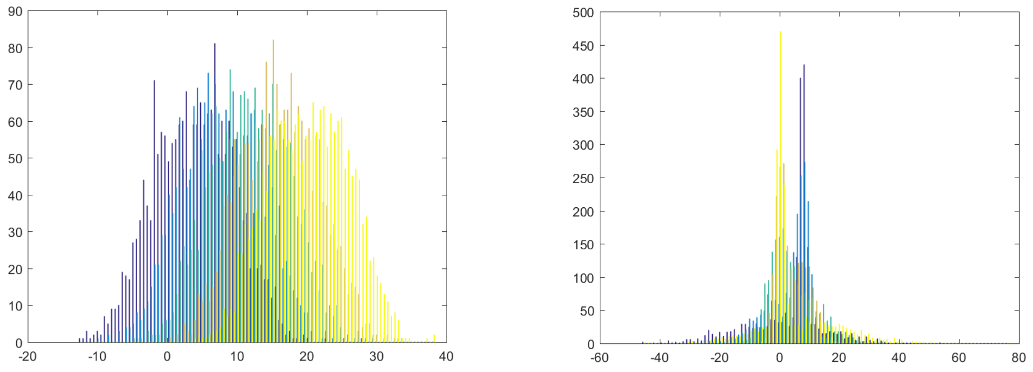
6. Experiment Result and Analysis

6.1. Dataset

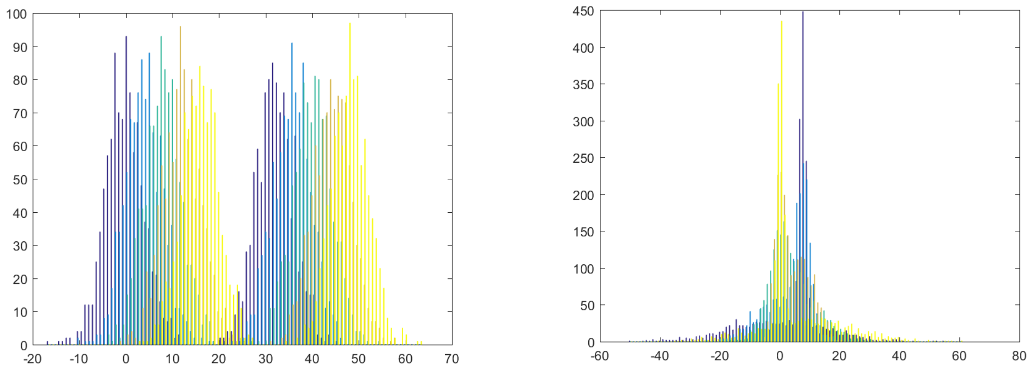
In this paper, we used a three package dataset that consisted of synthetic and benchmark datasets. The SyntheticA package is a simple synthetic dataset that has two peaks, five classes, and two features. This package has four instances, and the distribution of the data is shown in Figure 7. The second package is SyntheticB, which is a synthetic dataset that contains eight instances of data. The datasets have different numbers of classes and peaks, but all of them have five features.



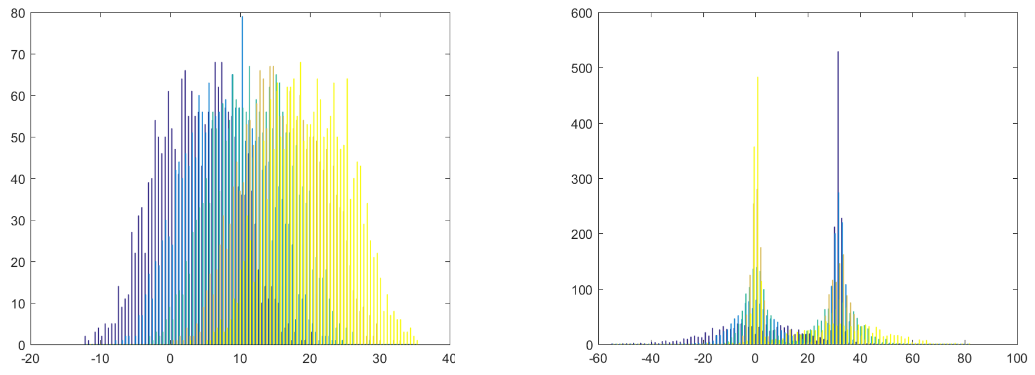
(**a**)



(**b**)



(**c**)



(**d**)

Figure 7. Distribution of the SyntheticA dataset: (**a**) instance 1, (**b**) instance 2, (**c**) instance 3, and (**d**) instance 4.

In addition to synthetic datasets, in this paper, we used a benchmark dataset. In this study, we utilized an open dataset that was published in the UCI online database. The benchmark dataset also consisted of the dataset from previous research. The benchmark dataset that was used in this study has multi-modality in its features. The detail of the dataset used in this paper is described in Table 2. The SyntheticA dataset is a small feature (two features) dataset, where all classes have multi-modal distribution. The SyntheticB dataset is similar to the previous SyntheticA dataset, but there are variations of numbers of classes and peaks, and its feature size is larger. The benchmark dataset was taken from real datasets collected in previous studies where some features may have multi-modal distribution and some features may not have had such. The distribution of all synthetic and benchmark datasets is shown in Appendix A section.

Table 2. Detailed Information of the Datasets.

No	Dataset	Information	#Classes	#Features	#Instances
1	SyntheticA-i1	2peak;5class;2feature	5	2	10,000
2	SyntheticA-i2	2peak;5class;2feature	5	2	10,000
3	SyntheticA-i3	2peak;5class;2feature	5	2	10,000
4	SyntheticA-i4	2peak-5class;2feature	5	2	10,000
5	SyntheticB-i1	2peak;2class;5feature	2	5	4000
6	SyntheticB-i2	2peak;3class;5feature	3	5	6000
7	SyntheticB-i3	2peak;4class;5feature	4	5	8000
8	SyntheticB-i4	2peak;5class;5feature	5	5	10,000
9	SyntheticB-i5	3peak;2class;5feature	2	5	6000
10	SyntheticB-i6	3peak;3class;5feature	3	5	9000
11	SyntheticB-i7	3peak;4class;5feature	4	5	12,000
12	SyntheticB-i8	3peak;5class;5feature	5	5	15,000
13	Pinwheel	[49]	5	2	5000
14	Glass	UCI	6	9	214
15	Ionosphere	UCI	2	33	351
16	Breast Cancer Coimbra	UCI	2	9	116
17	Wall Following	UCI	3	2	5456
18	Segment	UCI	7	19	2310
19	Ecoli	UCI	4	7	336
20	Odor	[37]	12	8	2400

6.2. Experiment Setup

In this paper, we developed two versions of multi-codebook FNGLVQ that use intelligent clustering, three versions of multi-codebook FNGLVQ that use static incremental learning, and two versions of multi-codebook FNGLVQ that use dynamic incremental learning.

Three packages of datasets were used in the experiment, and these consisted of synthetic and benchmark datasets. The SyntheticA dataset was used in the first experiment. The aim of the first experiment was to evaluate the performance of the proposed methods in the multi-modal dataset because all features in the SyntheticA dataset had multi-modal distribution. In the experiment, we analyzed the performance of the proposed method compared to the original FNGLVQ. The SyntheticB dataset was used in the second experiment. The second experiment aimed to observe the performance of the proposed methods in the variation of peaks and classes of the multi-modal dataset. Overall, the synthetic dataset was used to evaluate the proposed methods in multi-modal data because all features in the synthetic dataset had multi-modality. The benchmark dataset was used in the third experiment. The benchmark dataset used in this study had a multi-modal distribution of its features. The aim of this experiment was to observe the performance of the proposed method in a real dataset. The benchmark dataset had a sign of multi-modality in its features. However, not all features in the data had multi-modal distribution. This meant that some features had multi-modal distribution but some features did not. Therefore, the benchmark dataset was used to measure the performance of the proposed methods in the partly multi-modal data.

In the experiment, we evaluated both multi-codebook FNGLVQ methods that use intelligent clustering and incremental learning. As mentioned before, we tried three static threshold values, i.e., 0.1, 0.2, and 0.25, for the multi-codebook method that uses static incremental learning. We had two versions of multi-codebook FNGLVQ that use dynamic incremental learning. Version 1 used Equation (34) to generate a new codebook, while version 2 used Equation (35) to generate a new codebook. For the multi-codebook FNGLVQ that uses clustering, we used intelligent clustering based on anomalous patterns and intelligent K-means clustering that uses histogram information. In this study, the proposed method was run for 100 iterations (epoch), alpha = 0.05, gamma = 0.00005, beta = 0.00005, and delta = 0.1. The experiment was conducted by using five-fold cross-validation. The parameter was chosen based on a preliminary experiment to find the most suitable FNGLVQ parameter. In the preliminary experiment, we tried the combination of alpha = {0.1, 0.05}, beta = {0.00001, 0.00005}, and gamma = {0.00001, 0.00005}, and we found that the last parameter had the best performance.

In the benchmark dataset, we compared the proposed method to the classical machine learning methods such as Naïve Bayes, SVM, MLP, tree bagging, and random forest. Additionally, the proposed method was compared to convolutional neural network (CNN) and dense layer methods. The comparator methods were implemented in Python, while the proposed method and original FNGLVQ were implemented in Matlab. The SVM method uses the radial basis function (RBF) kernel. The MLP uses a quasi-Newton optimizer, epoch = 200, and a 3 x 10 hidden layer size. The tree bagging and random forest use 10 estimators (weak classifiers) and the Gini index as criteria. There were two versions of the CNN used in the experiment. The first CNN version was CNN2L, which was taken from the VGG16 model, but it only uses one convolution block. The second version of the CNN has 10 convolution layers, but the previous version only used 2 layers. There are three versions of dense layer classifiers: the first version has two dense layers, the second version has 10 layers, and the third version has 10 layers and a dropout value of 0.2. The previous version of the classifiers including CNN used a dropout value of 0.5. The CNNs and dense layers used in this experiment had the following structure:

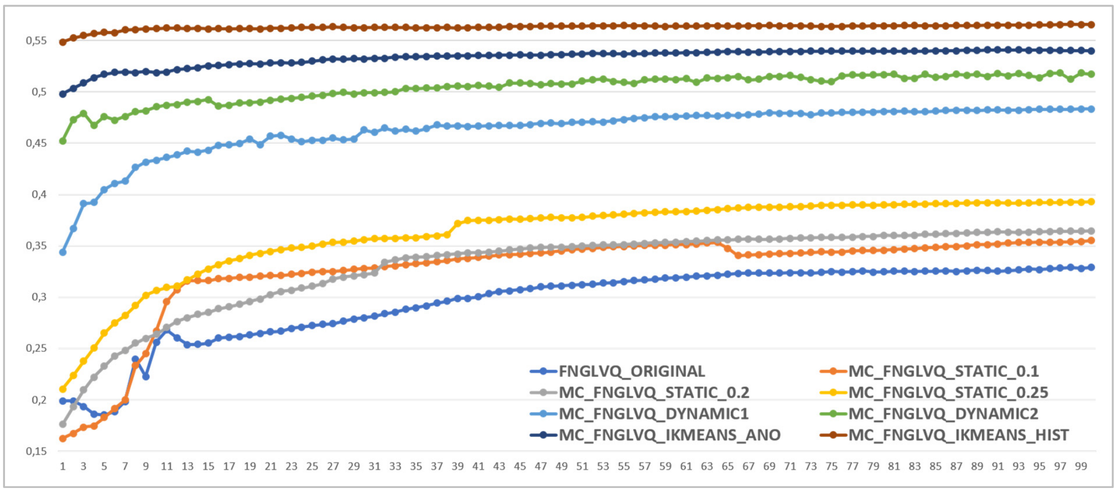
1. CNN2L: Input-Padd-Conv-Padd-Conv-Pool-Flat-Dense-Drop-Dense-Drop–Output.
2. CNN10L: Input-Padd-Conv-Padd-Conv-Padd-Conv-Padd-Conv-Padd-Conv-Padd-Conv-Padd-Conv-Padd-Conv-Padd-Conv-Padd-Conv-Pool-Flat-Dense-Drop-Dense-Drop-Output.
3. Dense2L: Dense-Drop-Dense-Drop-Output.
4. Dense10L: Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop–Output.
5. Dense10L02: Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Dense-Drop-Output, dropout value = 0.2.

6.3. Result of Scenario 1: Experiment on SyntheticA Dataset

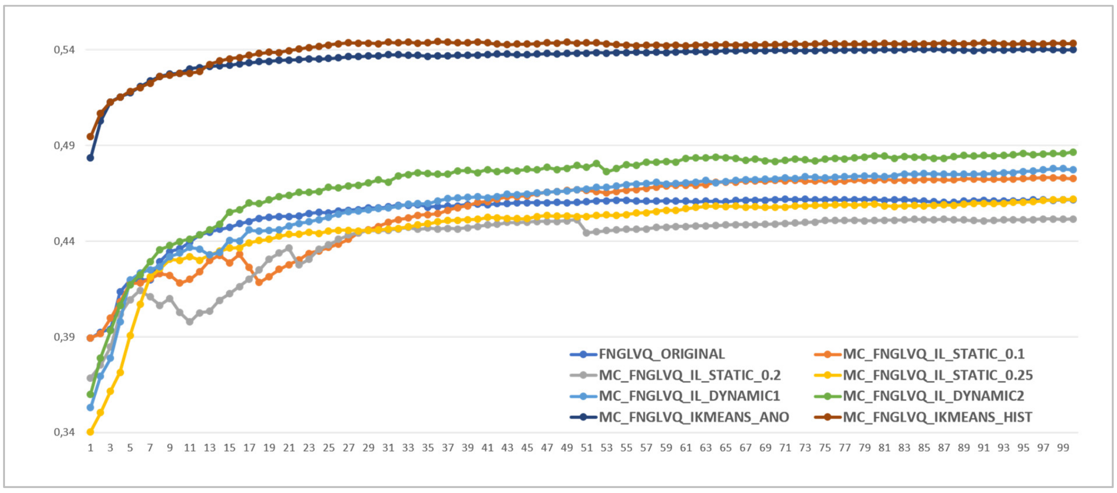
The result of the scenario 1 experiment is shown in Figure 8. The figure shows that from four instances of the SyntheticA package dataset, multi-codebook FNGLVQ that uses intelligent clustering had a higher accuracy than incremental learning by a 4% margin. The dynamic incremental learning achieved a better performance than static incremental learning by a 5% margin.

The highest accuracy was achieved by multi-codebook FNGLVQ that uses intelligent K-means clustering based on histogram information, followed by the multi-codebook FNGLVQ that uses intelligent clustering based on anomalous patterns, the multi-codebook FNGLVQ that uses dynamic incremental learning 2, the multi-codebook FNGLVQ that uses dynamic incremental learning 1, and the multi-codebook FNGLVQ that uses static incremental learning. However, the performance of the static incremental learning versions was not stable. It is shown in the figure that in SyntheticA instance 1, the static incremental learning version with 0.2 thresholds had a better accuracy than the original FNGLVQ, but in SyntheticA instances 2, 3, and 4, it had a lower accuracy than the original FNGLVQ. The static incremental learning version with 0.1 thresholds achieved a good performance in SyntheticA instances 1 and 2, but its performance in SyntheticA instances 3 and 4 dropped. Similar to the previous cases, static incremental learning with 0.25 thresholds achieved a good performance in SyntheticA instances 1, 3 and 4, but on the other hand, it had low performance in SyntheticA instance 2.

Overall, both multi-codebook methods that use intelligent clustering and dynamic incremental learning had better performances than the original FNGLVQ. However, the multi-codebook FNGLVQ that uses static incremental learning has worse performance than the original FNGLVQ in some cases, i.e., instances 2 and 4 in the SyntheticA dataset. The scenario 1 experiment also showed that in the SyntheticA dataset, the multi-codebook FNGLVQ that uses intelligent clustering had a better performance than the multi-codebook FNGLVQ that uses dynamic incremental learning.

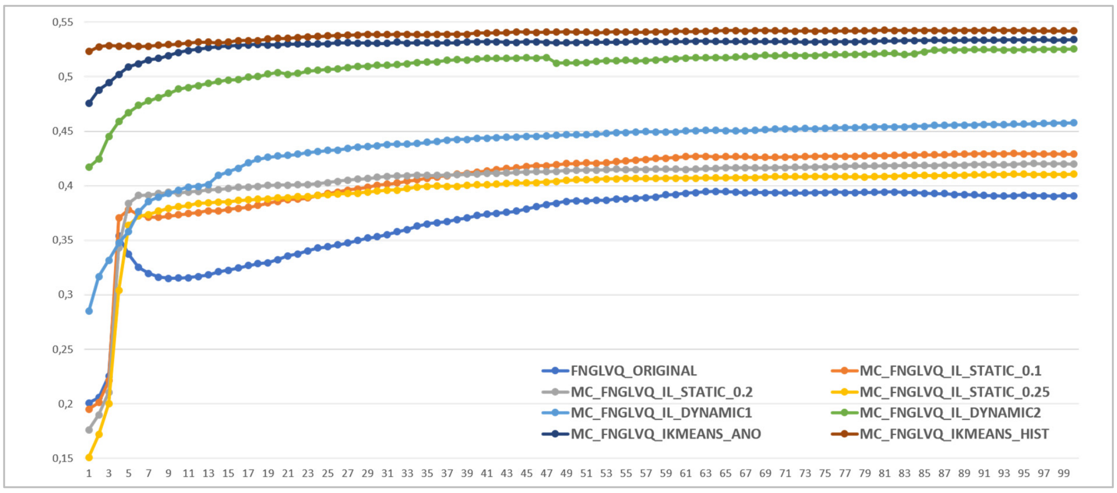


(**a**)

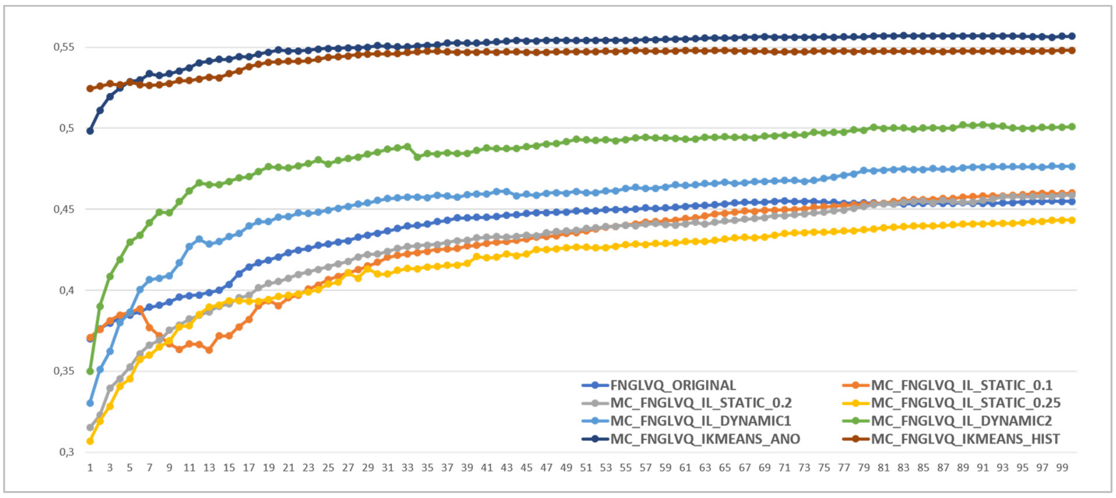


(**b**)

Figure 8. *Cont.*



(**c**)



(**d**)

Figure 8. Results of the SyntheticA dataset: (**a**) instance 1, (**b**) instance 2, (**c**) instance 3, and (**d**) instance 4. *X*-axis = epoch; *y*-axis = accuracy.

6.4. Result of Scenario 2: Experiment on SyntheticB Dataset

The experiment result in the SyntheticB dataset is shown in Table 3. Here, the multi-codebook FNGLVQ that uses intelligent K-means based on histogram information had the highest performance of 83.91%, on average. This was followed by the multi-codebook FNGLVQ that uses dynamic incremental learning 2, the multi-codebook FNGLVQ that uses IK-means based on anomalous patterns, and the multi-codebook method that uses incremental learning 1 with 81.79%, 81.77%, and 75.33% performances, respectively.

Both multi-codebook FNGLVQ methods that use intelligent clustering and dynamic incremental learning had better accuracy values than the original FNGLVQ in all instances of the SyntheticB dataset. On average, the dynamic incremental learning version 1 achieved a margin of more than 25% compared to the original method, while the dynamic incremental learning version 1 achieved a margin greater than 20%. The multi-codebook method that uses intelligent K-means clustering based on histogram information achieved an improvement greater than 28%, while the intelligent K-means clustering based on anomalous patterns version achieved an improvement greater than 26%, as compared to the original classifier. The static incremental learning version with 0.1 thresholds achieved the lowest accuracy on average, which was even less than the original classifier accuracy. The static incremental learning version with the 0.25 thresholds achieved a better accuracy with a 10% margin on average, though not in all instances of the SyntheticB dataset. It had a better performance than the original classifier. The static incremental learning with 0.2 thresholds also had a higher accuracy than the original method with a 7% margin on average, even though it had the same case as the static version when it had 0.25 thresholds where not in all instances of the SyntheticB dataset It had better performance than the original version. Those static versions with 0.2 and 0.25 thresholds had lower accuracy values than the original version in some instances of the SyntheticB dataset, e.g., in the fourth instance.

Table 3. Experiment Results of the SyntheticB Dataset.

Dataset	FNGLVQ ORIGINAL	MC FNGLVQ IL STATIC 0.1	MC FNGLVQ IL STATIC 0.2	MC FNGLVQ IL STATIC 0.25	MC FNGLVQ IL DYNAMIC1	MC FNGLVQ IL DYNAMIC2	MC FNGLVQ IK-MEANS ANO	MC FNGLVQ IK-MEANS HIST
SyntheticB-i1	70.1	70.02	70.6	71.15	85.97	84.8	85.88	86.8
SyntheticB-i2	52.35	52.3	74.28	75.35	82.1	85.58	87.3	87.85
SyntheticB-i3	53.46	50	59.94	59.59	78.61	83.34	86.46	85.32
SyntheticB-i4	46.17	46.67	45.91	45.59	58.63	70.6	73.02	71.63
SyntheticB-i5	62.62	63.1	74.56	86.97	89.32	89.48	87.21	91.5
SyntheticB-i6	54.3	54.23	66.3	73.14	82.8	84.51	83.59	89.29
SyntheticB-i7	53.54	51.41	56.99	61.71	64.84	82.47	78.76	84.48
SyntheticB-i8	48.17	51.6	51.53	52.31	60.37	73.5	71.93	74.43
Average	**55.09**	**54.92**	**62.51**	**65.73**	**75.33**	**81.79**	**81.77**	**83.91**

6.5. Result of Scenario 3: Experiment on Benchmark Dataset

The results of the experiment on the benchmark dataset is shown in Table 4. The table shows that on the benchmark dataset, on average, all versions of the proposed method had a better accuracy than the original FNGLVQ. In the benchmark dataset experiment, the best accuracy was achieved by the multi-codebook method that uses dynamic incremental learning version 1, followed by dynamic incremental learning version 2, static incremental learning 0.25, static incremental learning 0.2, intelligent K-means clustering based on histogram information, intelligent K-means clustering based on anomalous patterns, and static incremental learning 0.1, with 83.91%, 83.23%, 81.92%, 80.92%, 80.78%, 78.88% and 78.74% average accuracy values, respectively. Compared to the original FNGLVQ, the multi-codebook methods that use dynamic incremental learning 1 and dynamic incremental learning 2 achieved 5.31% and 4.63% margins, respectively. Surprisingly, the multi-codebook method that uses intelligent clustering had a lower performance of 3% below the dynamic incremental learning version. However, It was still higher than the original version with a 2% margin. The static incremental learning versions also had a better accuracy than the original version with 3%, 2% and 0.16% margins for static incremental learning with thresholds 0.25, 0.2 and 0.1, respectively.

Table 4. Experiment Results of the Benchmark Dataset.

Dataset	Pinwheel	Glass	Ionosphere	Breast Cancer Coimbra	Wall Following	Segment	Ecoli	Odor	Average
FNGLVQ-ORIGINAL	92.24	59.25	88.60	64.67	78.42	80.86	89.28	75.45	**78.60**
MC-FNGLVQ-IL-STATIC-0.1	92.96	57.01	90.31	64.67	83.76	79.69	86.28	75.21	**78.74**
MC-FNGLVQ-IL-STATIC-0.2	94.78	60.20	90.60	64.67	79.46	85.67	90.46	81.54	**80.92**
MC-FNGLVQ-IL-STATIC-0.25	95.24	60.74	90.31	66.30	79.03	87.01	91.63	85.08	**81.92**
MC-FNGLVQ-IL-DYNAMIC1	94.76	62.20	91.17	68.20	86.44	87.88	94.05	86.58	**83.91**
MC-FNGLVQ-IL-DYNAMIC2	99.44	71.43	90.04	62.86	86.99	93.68	75.46	85.92	**83.23**
MC-FNGLVQ-IK-MEANS-ANO	94.41	55.72	93.43	63.96	73.15	85.58	87.20	77.58	**78.88**
MC-FNGLVQ-IK-MEANS-HIST	91.08	59.85	91.15	69.18	84.04	86.70	86.00	78.25	**80.78**
Naïve Bayes	90.00	53.00	84.00	61.00	90.00	79.00	63.00	71.00	**73.88**
SVM	100.00	56.00	90.00	52.00	93.00	60.00	85.00	89.00	**78.13**
MLP	99.00	35.00	90.00	48.00	40.00	13.00	72.00	58.00	**56.88**
Bagging Tree	73.00	72.00	89.00	61.00	84.00	95.00	82.00	87.00	**80.38**
Random Forest	93.00	58.00	86.00	57.00	94.00	78.00	82.00	45.00	**74.13**
CNN2L	99.9	24.34	59.14	53.04	97.51	13.55	99.4	99.37	**68.28**
CNN10L	39.72	31.44	70.29	55.65	40.42	15.29	52.23	24.42	**41.18**
Dense2L	98.16	21.11	57.14	44.35	98.55	13.82	99.7	60.23	**56.41**
Dense10L	23.49	31.37	55.99	46.96	66.95	14.42	42.69	15.71	**37.20**
Dense10L02	62.28	23.52	61.14	46.96	40.42	14.02	34.92	88.38	**46.46**

Compared to the commonly used classifiers, the proposed method still had a better performance overall. The proposed method achieved 3%. 5%, 9%, 10%, 10%, and 27% performances compared to the tree bagging, SVM, random forest, Naïve Bayes, and MLP methods. The CNN and dense layers classifiers performed very well in some datasets, but they performed very poorly in other datasets. For example, the classifiers performed very well in the pinwheel, wall following, ecoli, and odor datasets, but they performed very poorly in the glass and segment datasets. Overall, the best achievement came from the CNN and dense layers methods at 68.28% and 56.88%, respectively. This means that, compared to the CNN and dense layers methods, the proposed had a higher accuracy with a margin of more than 15%. Table 4 also shows than adding layers to the CNN and dense layers methods did not increase their performances. It did increase the performance in a few datasets but decreased the performances in other datasets by a significant amount. Using a dropout value of 0.2 in Dense10L02 version increased the performance of the classifiers from 37.20% to 46.46%, on average. However, if we look closer, the improvement only occurred in three of the eight tested datasets.

6.6. Improvement of Proposed Method Compared to Original FNGLVQ

The improvement of the proposed methods compared to original FNGLVQ on both the synthetic and benchmark datasets is summarized in Table 5. Table 5 shows that multi-codebook FNGLVQ that uses intelligent K-means based on anomalous patterns achieved 22.24%, 0.28%, and 13.26% margins on the synthetic dataset, the benchmark dataset and the average of all datasets, respectively. The multi-codebook FNGLVQ that uses intelligent K-means based on histogram information achieved 23.90%, 2.10% and 15.02% margins on the synthetic dataset, the benchmark dataset and the average of all datasets, respectively. The multi-codebook FNGLVQ that uses dynamic incremental learning version 1 achieved 15.65%, 5.31%, and 11.42% margins on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively. The multi-codebook FNGLVQ that uses dynamic incremental learning version 2 achieved 21.08%, 4.63%, and 14.35% margins on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively. The multi-codebook FNGLVQ that uses static incremental learning with a 0.25 threshold value achieved 7.70%, 3.32%, and 5.91% margins on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively. The static incremental learning version with a 0.2 threshold value achieved 5.44%, 2.33% and 4.16% margins on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively. However, the static incremental version with a 0.1 threshold value a had a slight improvement. The improvement was less than 1% in the synthetic dataset, the benchmark dataset, and the average of all datasets.

Table 5. Improvements of the Proposed Method.

Dataset	MC FNGLVQ IL STATIC 0.1	MC FNGLVQ IL STATIC 0.2	MC FNGLVQ IL STATIC 0.25	MC FNGLVQ IL DYNAMIC1	MC FNGLVQ IL DYNAMIC2	MC FNGLVQ IK-MEANS ANO	MC FNGLVQ IK-MEANS HIST
SyntheticA-i1	2.63	3.56	6.39	15.43	18.83	21.12	23.66
SyntheticA-i2	1.1	−1.02	0.01	1.56	2.48	7.83	8.16
SyntheticA-i3	3.83	2.91	1.99	6.7	13.47	14.33	15.12
SyntheticA-i4	0.52	0.39	−1.14	2.16	4.62	10.2	9.32
SyntheticB-i1	−0.08	0.5	1.05	15.87	14.7	15.78	16.7
SyntheticB-i2	−0.05	21.93	23	29.75	33.23	34.95	35.5
SyntheticB-i3	−3.46	6.48	6.13	25.15	29.88	33	31.86
SyntheticB-i4	0.5	−0.26	−0.58	12.46	24.43	26.85	25.46
SyntheticB-i5	0.48	11.94	24.35	26.7	26.86	24.59	28.88
SyntheticB-i6	−0.07	12	18.84	28.5	30.21	29.29	34.99
SyntheticB-i7	−2.13	3.45	8.17	11.3	28.93	25.22	30.94
SyntheticB-i8	3.43	3.36	4.14	12.2	25.33	23.76	26.26
Pinwheel	0.72	2.54	3	2.52	7.2	2.17	−1.16
Glass	−2.24	0.95	1.49	2.95	12.18	−3.53	0.6
Ionosphere	1.71	2	1.71	2.57	1.44	4.83	2.55
Breast Cancer Coimbra	0	0	1.631	3.531	−1.811	−0.714	4.51
Wall Following	5.34	1.04	0.61	8.02	8.57	−5.27	5.62
Segment	−1.17	4.81	6.15	7.02	12.82	4.72	5.84
Ecoli	−3	1.18	2.35	4.77	−13.82	−2.08	−3.28
Odor	−0.24	6.09	9.63	11.133	10.47	2.13	2.8
Average Synthetic	**0.56**	**5.44**	**7.70**	**15.65**	**21.08**	**22.24**	**23.90**
Average Benchmark	**0.14**	**2.33**	**3.32**	**5.31**	**4.63**	**0.28**	**2.19**
Average All	**0.39**	**4.16**	**5.91**	**11.42**	**14.35**	**13.26**	**15.02**

6.7. Analysis of Variance (ANOVA) Test for Significance Testing

In the previous subsection, we showed that the proposed multi-codebook methods had performance improvements over the original FNGLVQ. This sub-section discusses whether the improvements are significant. We conducted significant testing by using a one-way analysis of variance (ANOVA) test. A one-way ANOVA test is a statistical technique that is used to compare means of two or more samples by using F-distribution [50]. The ANOVA tests a null hypothesis if two or more sets of observations are drawn from populations with the same mean. In this study, the ANOVA test was used to test the significance difference between two classifiers based on their performance on the tested datasets. In this analysis, if the null hypothesis was not rejected, then the two tested classifiers were not significantly different. In the other side, if the null hypothesis was rejected, then the performance of the two tested classifiers was significantly different.

Each of the proposed multi-codebook versions was tested by using an ANOVA test with the original FNGLVQ. The ANOVA test was conducted by using all 20 instances of the datasets (12 synthetic datasets and 8 benchmark datasets), as mentioned in the previous subsection. The test was conducted by using a significant level (α) 0.05. There were two conditions that could be used to reject the null hypothesis. The first condition was based on the F-test, where the null hypothesis was rejected if the F value was more than Fc. The second condition was based on the *p*-value, where the null hypothesis was rejected if the *p*-value was less than 0.05.

Table 6 shows the summary of significance testing by using NOVA for the proposed methods compared to the original method. The table shows that from 20 cases of the tested dataset, the proposed multi-codebook FNGLVQ that use intelligent clustering and dynamic incremental learning had significant improvements compared to the original classifier, whereas the multi-codebook method that uses static incremental learning version had no significant improvement. However, looking to the values of F, Fc, and *p*-value, it can be seen that the multi-codebook FNGLVQ that uses dynamic incremental learning version 1 had a slightly different value from the thresholds. The F value of the method was 4.355, which is close to the Fc value 4.098. Its *p*-value was also near the thresholds (0.05) with 0.0437.

Table 6. Summary of Significance Testing by Using One-Way ANOVA.

	F	Fc	*p*-Value	Significance
MC_FNGLVQ_STATIC_0.1	0.005	4.098	0.9439	No
MC_FNGLVQ_STATIC_0.2	0.559	4.098	0.4593	No
MC_FNGLVQ_STATIC_0.25	1.083	4.098	0.3047	No
MC_FNGLVQ_DYNAMIC1	4.355	4.098	0.0437	Yes
MC_FNGLVQ_DYNAMIC2	7.63	4.098	0.0088	Yes
MC_FNGLVQ_IKMEANS_ANO	7.066	4.098	0.0114	Yes
MC_FNGLVQ_IKMEANS_HIST	9.215	4.098	0.0043	Yes

6.8. Discussion and Recommendation

The proposed multi-codebook FNGLVQ was tested in the synthetic and benchmark datasets. The synthetic dataset was the dataset where all features of the data had multi-modal distribution. The synthetic dataset consisted of two, and tree modals (peaks), and two to five classes. The synthetic dataset also had a high overlap between classes. The high overlap between classes is shown in the Appendix A section. Both multi-codebook FNGLVQ methods based on intelligent clustering and dynamic incremental learning achieved higher performances than the original FNGLVQ on each dataset instance. On the SyntheticB dataset, the proposed method achieved 83.91% and 81.79% performances for multi-codebook methods that use intelligent clustering and incremental learning, respectively, while the original FNGLVQ achieved a performance of 55.09%. The proposed method's performance was indeed still below 90%. However, this study was intended to show the improvements of the proposed method compared to the original FNGLVQ. The proposed method achieved a 23.90% improvement on the SyntheticA and SyntheticB datasets. The good news is the proposed multi-codebook FNGLVQ still has room for improvements in future studies.

Beside being tested in the synthetic dataset, the proposed multi-codebook methods were also tested in the benchmark dataset. The aim of the experiment on the benchmark dataset was to test the method with real data that were naturally taken from the real world. The datasets had signs of multi-modal on their features (but not all features). This is shown in Appendix A figures. The experiment result showed that overall the proposed methods achieved better performance than the original FNGLVQ. The highest performance was achieved by a multi-codebook method that uses dynamic incremental learning 1, with a 5.3% improvement compared to the original FNGLVQ. This was followed by the multi-codebook method that uses dynamic incremental learning 2, with a 4.63% improvement. In the benchmark dataset, the multi-codebook method that uses intelligent clustering performed below the dynamic incremental version, with only a 2.19% improvement, on average. Based on the experiment results, the author recommends using the multi-codebook FNGLVQ that uses intelligent K-means clustering based on histogram information in datasets where all features have multi-modal distribution. Meanwhile, in datasets where only several (not all) features have multi-modal distribution, the author recommends using the multi-codebook FNGLVQ that uses dynamic incremental learning.

This study also proves that the proposed multi-codebook FNLGVQ that use intelligent clustering and dynamic incremental learning have significant improvements compared to the original FNGLVQ. The significant improvement was tested by using the ANOVA test from 20 instances of the synthetic and benchmark datasets. The ANOVA test also proved that the multicode-book FNGLVQ that uses static incremental learning has a insignificant improvement over the original FNGLVQ

7. Conclusions

In this study, we proposed multi-codebook FNGLVQ that use intelligent clustering and dynamic incremental learning. There are two variations of the multi-codebook FNGLVQ that uses the intelligent clustering approach: One uses intelligent clustering based on anomalous patterns, and the other uses intelligent clustering based on histogram information. There are two variations of the multi-codebook FNGLVQ that uses the dynamic incremental learning approach: One compares the similarity of the winner class and the mean of other class similarities, and the other compares the similarity of the winner class and the min of other classes similarities to generate a new codebook. Overall, the multi-codebook FNGLVQ that uses incremental learning 1 was found to achieve improvements of 15.65%, 5.31% and 11.42% on the synthetic dataset, the benchmark dataset and the average of all datasets, respectively. The multi-codebook FNGLVQ that uses incremental learning 2 was found to achieve improvements 21.08%, 4.63%, and 14.35% on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively. The multi-codebook FNGLVQ that uses intelligent clustering based on anomalous patterns was found to achieve improvements of 22.24%, 0.28%, and 13.26% on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively. The multi-codebook FNGLVQ that uses intelligent clustering based on histogram information was found to achieve improvements of 23.90%, 2.10%, and 15.02% on the synthetic dataset, the benchmark dataset, and the average of all datasets, respectively. An ANOVA test concluded that the multi-codebook FNGLVQ that use intelligent clustering and dynamic incremental learning have significant improvements compared to the original FNGLVQ. The author recommends utilizing multi-codebook FNGLVQ that uses intelligent K-means clustering based on histogram information if all the features have multi-modal distribution and utilizing the multi-codebook FNGLVQ that uses dynamic incremental learning 1 if only several features have multi-modal distribution.

8. Future Works

As stated in the previous sub-section, the proposed methods do have room for improvements to achieve better performance in multi-modal data classification. There are several insights that are worth trying in future studies. The first insight is combining intelligent clustering and dynamic incremental learning into one integrated method. The second insight is modifying incremental learning with an add-and-delete codebook mechanism. The codebook addition is used when the existing codebooks cannot properly fit the input vector, while the codebook deletion is used when the existing codebooks overfit the input vector. However, the add-and-delete mechanism should be supported by additional control to void add-and-delete loop during training. The third insight is using a distribution approximation method such as that of the Markov Chain Monte Carlo (MCMC) to approximate the data distribution and resample the data. The resampling will be beneficial for data with a small number of instances and in a cluster with few members. Using more samples will produce a better codebook, such as a codebook that has better generalization capability.

Author Contributions: All the authors contributed equally to the conception of the idea, implementing and analyzing the experimental results, and writing the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This paper is founded by Universitas Indonesia Q1Q2 Research Grant 2019 Grant Number NKB-0210/UN2.R3.1/HKP.05.00/2019.

Acknowledgments: The authors sincerely thanks to Directorate of Research and Community Engagements Universitas Indonesia and Research Division of Faculty of Computer Science Universitas Indonesia. This paper is supported by Universitas Indonesia Q1Q2 Research Grant 2019 Grant Number NKB-0210/UN2.R3.1/HKP.05.00/2019.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A The Distribution of SyntheticA Dataset

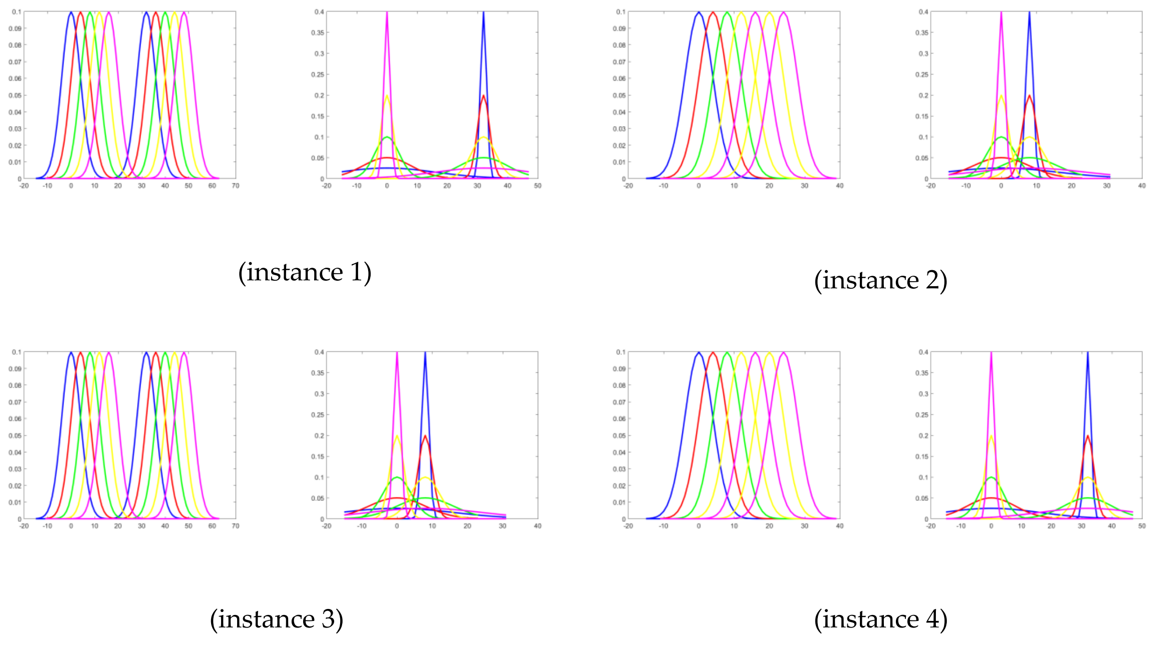
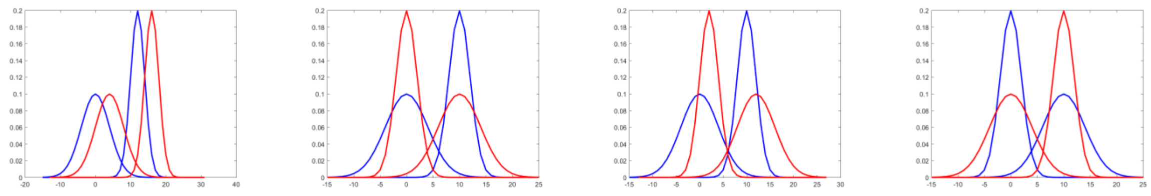
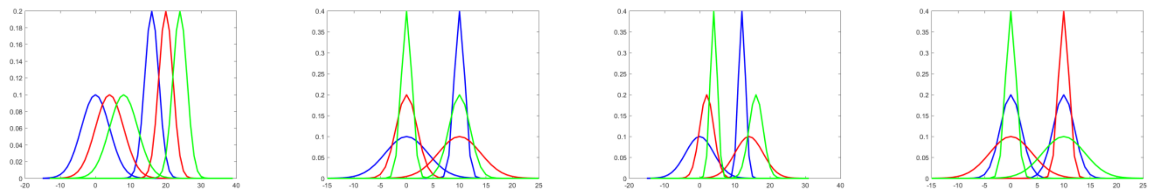


Figure A1. The Distribution of SyntheticA Dataset.

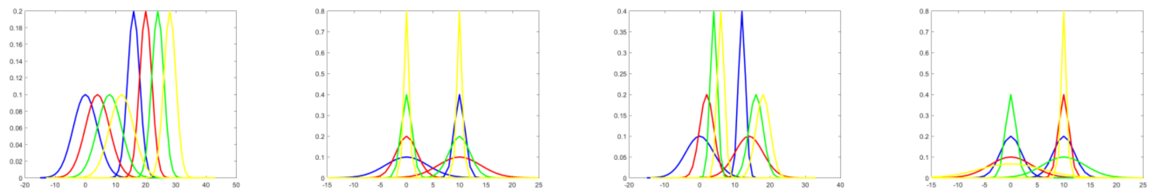
Appendix B The Distribution of SyntheticB Dataset



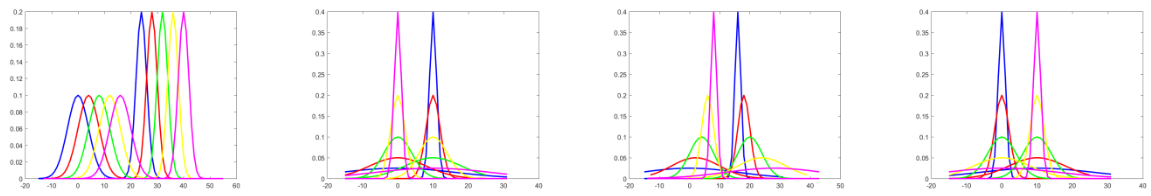
(2 class 2 peak)



(3 class 2 peak)

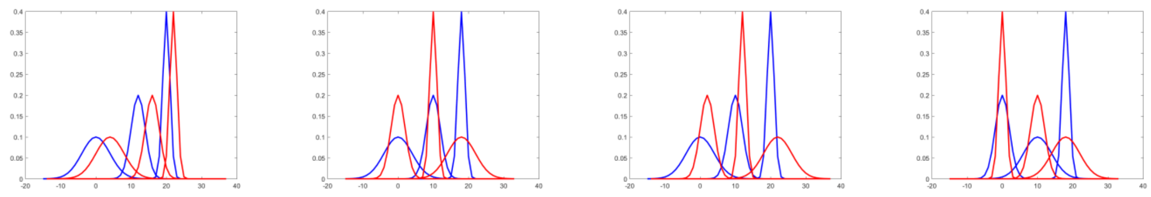


(4 class 2 peak)

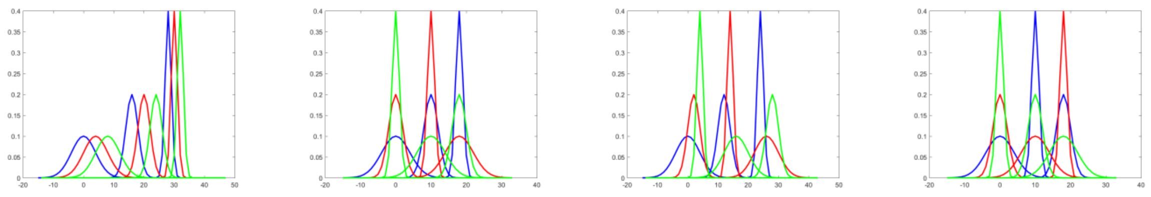


(5 class 2 peak)

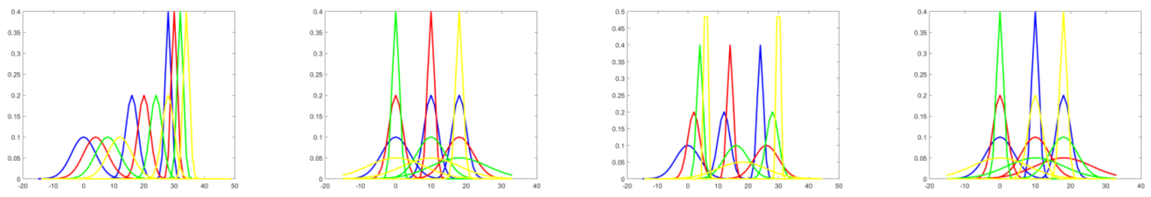
Figure A2. *Cont.*



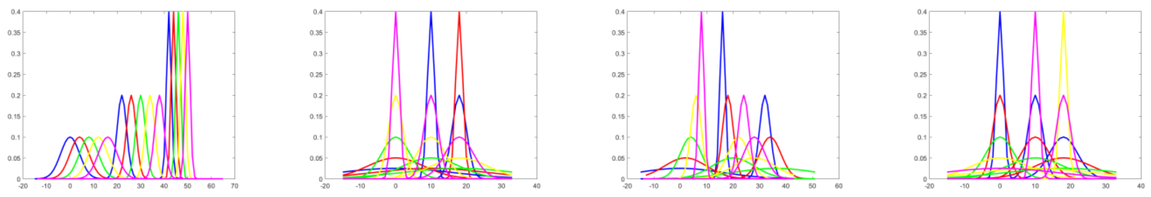
(2 class 3 peak)



(3 class 3 peak)



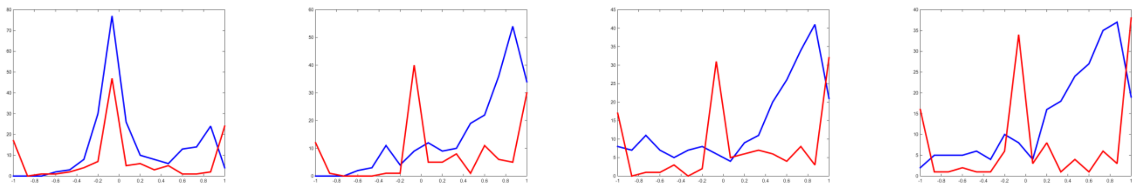
(4 class 3 peak)



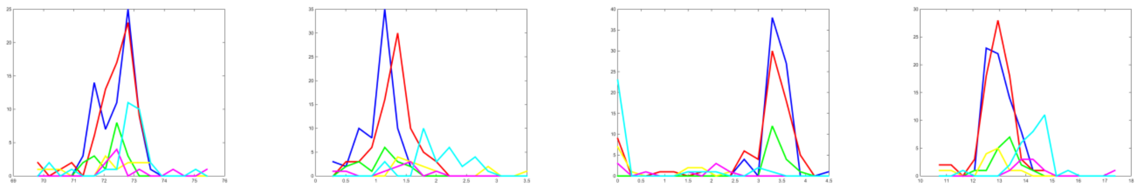
(5 class 3 peak)

Figure A2. The Distribution of SyntheticB Dataset.

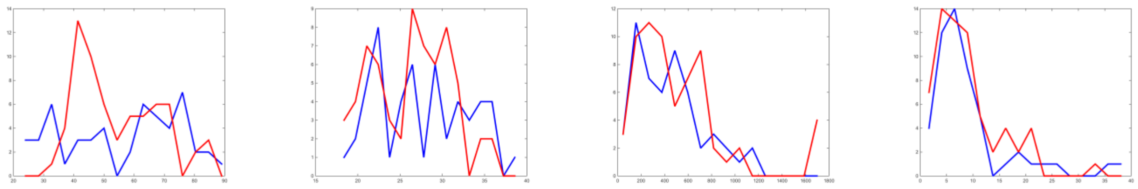
Appendix C The Distribution of Benchmark Dataset



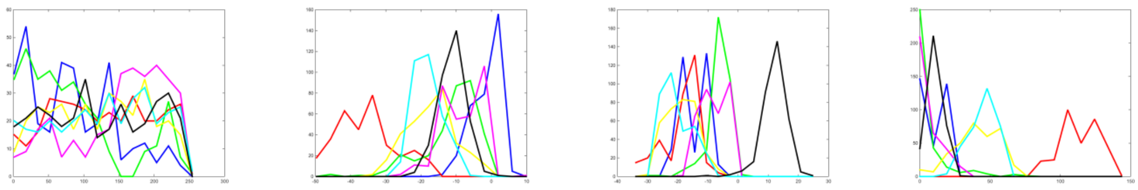
(Ionosphere)



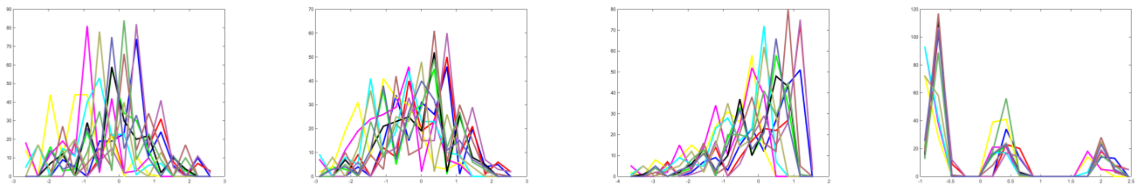
(Glass)



(Breast Cancer)

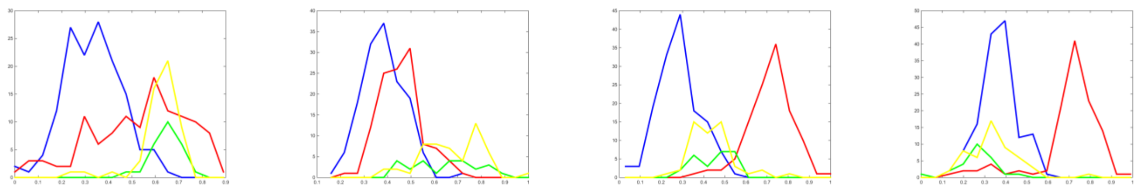


(Segment)

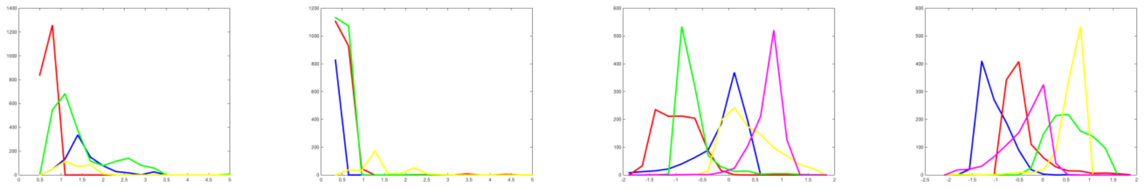


(Odor)

Figure A3. *Cont.*



(Ecoli)



(Wall following) (Pinwheel)

Figure A3. The Distribution of Benchmark Dataset.

References

1. Ma'Sum, M.A.; Arrofi, M.K.; Jati, G.; Arifin, F.; Kurniawan, M.N.; Mursanto, P.; Jatmiko, W. Simulation of intelligent unmanned aerial vehicle (uav) for military surveillance. In Proceedings of the IEEE 2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS), Kuta, Bali, 28–29 September 2013; pp. 161–166.
2. Nüchter, A.; Joachim, H. Towards semantic maps for mobile robots. *Robot. Auton. Syst.* **2008**, *56*, 915–926. [CrossRef]
3. Martinez-Cantin, R.; Nando de, F.; Arnaud, D.; José, A.C. Active policy learning for robot planning and exploration under uncertainty. In *Robotics: Science and Systems*; MIT Press: Cambridge, MA, USA, 2007; Volume 3, pp. 334–341.
4. Baltrušaitis, T.; Chaitanya, A.; Louis-Philippe, M. multi-modal machine learning: A survey and taxonomy. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**. [CrossRef]
5. Corneanu, C.A.; Simón, M.O.; Cohn, J.F.; Guerrero, S.E. Survey on rgb, 3d, thermal, and multimodal approaches for facial expression recognition: History, trends, and affect-related applications. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 1548–1568. [CrossRef] [PubMed]
6. Soleymani, M.; Garcia, D.; Jou, B.; Schuller, B.; Chang, S.F.; Pantic, M. A survey of multimodal sentiment analysis. *Image Vis. Comput.* **2017**, *65*, 3–14. [CrossRef]
7. Kumar, A.; Kim, J.; Cai, W.; Fulham, M.; Feng, D. Content-based medical image retrieval: A survey of applications to multidimensional and multimodality data. *J. Digit. Imaging* **2013**, *26*, 1025–1039. [CrossRef] [PubMed]
8. Oskouie, P.; Alipour, S.; Eftekhari-Moghadam, A.M. Multimodal feature extraction and fusion for semantic mining of soccer video: A survey. *Artif. Intell. Rev.* **2014**, *42*, 173–210. [CrossRef]
9. Abidi, B.R.; Aragam, N.R.; Yao, Y.; Abidi, M.A. Survey and analysis of multimodal sensor planning and integration for wide area surveillance. *ACM Comput. Surv. (CSUR)* **2009**, *41*, 7. [CrossRef]
10. Kiela, D.; Grave, E.; Joulin, A.; Mikolov, T. Efficient large-scale multi-modal classification. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
11. Vortmann, L.M.; Schult, M.; Benedek, M.; Walcher, S.; Putze, F. Real-Time Multimodal Classification of Internal and External Attention. In Proceedings of the Adjunct of the 2019 International Conference on Multimodal Interaction, ACM, Suzhou, China, 14–18 October 2019; p. 14.
12. Poria, S.; Cambria, E.; Hussain, A.; Huang, G.B. Towards an intelligent framework for multimodal affective data analysis. *Neural Netw.* **2015**, *63*, 104–116. [CrossRef]

13. Atrey, P.K.; Hossain, M.A.; El Saddik, A.; Kankanhalli, M.S. Multimodal fusion for multimedia analysis: A survey. *Multimed. Syst.* **2010**, *16*, 345–379. [CrossRef]
14. Ma'sum, M.; Sanabila Anwar, H.R.; Jatmiko, W. Multi codebook lvq-based artificial neural networks using clustering approach. In Proceedings of the IEEE 2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS), Depok, Indonesia, 10–11 October 2015; pp. 263–268.
15. Anwar Ma'sum, M.; Wisnu, J. Multi-codebook Fuzzy Neural Network Using Incremental Learning for Multimodal Data Classification. In Proceedings of the IEEE 2019 4th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS), Nagoya, Japan, 13–15 July 2019; pp. 205–210.
16. Losing, V.; Barbara, H.; Heiko, W. Incremental on-line learning: A review and comparison of state of the art algorithms. *Neurocomputing* **2018**, *275*, 1261–1274. [CrossRef]
17. Krawczyk, B.; Minku, L.L.; Gama, J.; Stefanowski, J.; Woźniak, M. Ensemble learning for data stream analysis: A survey. *Inf. Fusion* **2017**, *37*, 132–156. [CrossRef]
18. Hartigan, J.A.; Manchek, A.W. Algorithm AS 136: A k-means clustering algorithm. *J. R. Stat. Soc. Ser. C (Appl. Stat.)* **1979**, *28*, 100–108. [CrossRef]
19. Banfield, J.D.; Adrian, E. Model-based Gaussian and non-Gaussian clustering. *Biometrics* **1993**, *49*, 803–821. [CrossRef]
20. Chiang, M.M.-T.; Boris, M. Intelligent choice of the number of clusters in k-means clustering: An experimental study with different cluster spreads. *J. Classif.* **2010**, *27*, 3–40. [CrossRef]
21. Anwar Ma'sum, M.; Dewa, M.S.A.; Indra, H.; Wisnu, J.; Adi, N. Multicodebook Neural Network Using Intelligent K-Means Clustering Based on Histogram Information for Multimodal Data Classification. In Proceedings of the IEEE 2018 International Workshop on Big Data and Information Security (IWBIS), Jakarta, Indonesia, 12–13 May 2018; pp. 129–135.
22. Safavian, S.R.; David, L. A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man Cybern.* **1991**, *21*, 660–674. [CrossRef]
23. Rish, I. An empirical study of the naive Bayes classifier. In Proceedings of the IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, Seattle, DC, USA, 4 August 2001; Volume 3, pp. 41–46.
24. Chung, K.-M.; Kao, W.-C.; Sun, C.-L.; Wang, L.-L.; Lin, C.-J. Radius margin bounds for support vector machines with the RBF kernel. *Neural Comput.* **2003**, *15*, 2643–2681. [CrossRef]
25. Pal, S.K.; Sushmita, M. Multilayer perceptron, fuzzy sets, and classification. *IEEE Trans. Neural Netw.* **1992**, *3*, 683–697. [CrossRef]
26. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
27. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
28. Simonyan, K.; Andrew, Z. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
29. Setiawan, I.M.A.; Imah, E.M.; Jatmiko, W. Arrhytmia classification using Fuzzy-Neuro Generalized Learning Vector Quantization. In Proceedings of the 2011 International Conference on Advanced Computer Science and Information System (ICACSIS), Jakarta, Indonesia, 17–18 December 2011; pp. 385–390.
30. Kohonen, G.T. Learning Vector Quantization for Pattern Recognition. In *Report TKK-F-A601*; Helsinki University of Technology: Espoo, Finland, 1986.
31. Sato, A.; Yamada, K. A formulation of learning vector quantization using a new misclassification measure. In Proceedings of the IEEE Computer Society 14th International Conference on Pattern Recognition-, ICPR '98, Washington, DC, USA, 16–20 August 1998; Volume 1, p. 322.
32. Zhang, D.; Wang, Y.; Zhou, L.; Yuan, H.; Shen, D.; Initiative, A.D.N. Multimodal classification of Alzheimer's disease and mild cognitive impairment. *Neuroimage* **2011**, *55*, 856–867. [CrossRef] [PubMed]
33. Cauwenberghs, G.; Tomaso, P. Incremental and decremental support vector machine learning. *Adv. Neural Inf. Process. Syst.* **2001**, 409–415. Available online: http://papers.nips.cc/paper/1814-incremental-and-decremental-support-vector-machine-learning.pdf (accessed on 1 January 2020).
34. Molina, J.F.G.; Zheng, L.; Sertdemir, M.; Dinter, D.J.; Schönberg, S.; Rädle, M. Incremental learning with SVM for multimodal classification of prostatic adenocarcinoma. *PLoS ONE* **2014**, *9*, e93600.
35. Huang, G.-B.; Chen, L. Convex incremental extreme learning machine. *Neurocomputing* **2007**, *70*, 3056–3062. [CrossRef]

36. Anwar Ma'sum, M.; Dewa, M.S.A.; Novian, H.; Wisnu, J. Enhance generalized learning vector quantization using unsupervised extreme learning machine and intelligent k-means clustering. In Proceedings of the IEEE 2017 International Workshop on Big Data and Information Security (IWBIS), Jakarta, Indonesia, 23–24 September 2017; pp. 77–83.
37. Roszkowska, E.; Tomasz, W. Application of fuzzy TOPSIS to scoring the negotiation offers in ill-structured negotiation problems. *Eur. J. Oper. Res.* **2015**, *242*, 920–932. [CrossRef]
38. Kusumoputro, B.; Hary, B.; Wisnu, J. Fuzzy-neuro LVQ and its comparison with fuzzy algorithm LVQ in artificial odor discrimination system. *ISA Trans.* **2002**, *41*, 395–407. [CrossRef]
39. Jatmiko, W.; Rochmatullah, R.; Kusumoputro, B.; Sekiyama, K.; Fukuda, T. Fuzzy learning vector quantization based on particle swarm optimization for artificial odor discrimination system. *WSEAS Trans. Syst.* **2009**, *8*, 1239–1252.
40. Imah, E.M.; Wisnu, J.; Basaruddin, T. Adaptive Multilayer Generalized Learning Vector Quantization (AMGLVQ) as new algorithm with integrating feature extraction and classification for Arrhythmia heartbeats classification. In Proceedings of the 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Seoul, Korea, 14–17 October 2012; pp. 150–155.
41. Krizhevsky, A.; Ilya, S.; Geoffrey, E.H. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2017**, 1097–1105. [CrossRef]
42. Huang, G.; Zhuang, L.; Laurens, V.D.M.; Kilian, Q. Weinberger. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
43. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
44. Zoph, B.; Vijay, V.; Jonathon, S.; Quoc, V.L. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8697–8710.
45. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
46. Losing, V.; Barbara, H.; Heiko, W. Interactive online learning for obstacle classification on a mobile robot. In Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–16 July 2015; pp. 1–8.
47. Liang, N.-Y.; Huang, G.-B.; Saratchandran, P.; Sundararajan, N. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Trans. Neural Netw.* **2006**, *17*, 1411–1423. [CrossRef]
48. Saffari, A.; Christian, L.; Jakob, S.; Martin, G.; Horst, B. On-line random forests. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops, Kyoto, Japan, 27 September–4 October 2009; pp. 1393–1400.
49. Glarner, T.; Patrick, H.; Janek, E.; Reinhold, H.-U. Full Bayesian Hidden Markov Model Variational Autoencoder for Acoustic Unit Discovery. In Proceedings of the Interspeech, Hyderabad, India, 2–6 September 2018; pp. 2688–2692.
50. Howell, D.C. Statistical methods for psychology. In *Wadsworth Cengage Learning*; Cengage Wadsworth: Belmont, CA, USA, 2009.