An Unsupervised Feed Forward Neural Network Method for Efficient Clustering

Roya Asadi[1], Mitra Asadi[1], and Shokoofeh Asadi[2]
[1]Central Tehran Branch, Islamic Azad University, Iran
[2]Department of Agricultural Management Engineering, University of Science and Research, Iran

Abstract: *This paper presents a Real Unsupervised Feed Forward Neural Network (RUFFNN) clustering method with one epoch training and data dimensionality reduction ability to overcome some critical problems such as low training speed, low accuracy as well as high memory complexity in this area. The RUFFNN method trains a code book of real weights by utilizing input data directly without using any random values. The Best Match Weight (BMW) vector is mined from the weight codebook and consequently the Total Threshold (TT) of each input data is computed based on the BMW. Finally, the input data are clustered based on their exclusive TT. For evaluation purposes, the clustering performance of the RUFFNN was compared to several related clustering methods using various data sets. The accuracy of the RUFFNN was measured through the number of clusters and the quantity of Correctly Classified Nodes(CCN).The superior clustering accuracies of 96.63%, 96.67% and 59.36% were for the breast cancer, iris and spam datasets from the UCI repository respectively. The memory complexity of the proposed method was* $O(m.n.s_m)$ *based on the number of nodes, attributes and size of the attribute.*

Keywords: *Artificial neural network, feed forward neural network, unsupervised learning, clustering, real weight.*

Received December 2, 2014; accepted August 16, 2015

1. Introduction

A Feed Forward Neural Network (FFNN) is a popular tool for statistical decision making and inspired by the brain task. In this network, processing of data has only one forward direction from the input layer to the output layer without any cycles or backward [1, 12]. Learning is an important property of the neural network.

2. Related Works

According to the study of current Unsupervised Feed Forward Neural Network (UFFNN) clustering methods [4], Vector Quantization (VQ) [16], K-means [11] and some UFFNN clustering methods such as Kohonen's Self-Organizing Map (SOM) [16], Neural Gas (NG) [17] and Growing Neural Gas (GNG) [9] are considered as fundamental patterns in the current UFFNN clustering methods in the static and online dynamic environments. The VQ gains a suitable codebook of the weights for clustering based on probability density functions. K-means is a partitioning clustering method by using Centroid-Based technique similar to the VQ, however, it should define a number of clusters and parameters before clustering [2, 13, 18]. NG is based on the VQ and data compression. The NG dynamically partitions itself like gas and describes the number of clusters, but it cannot control the growth of the network of nodes. The GNG method, on the other hand, is able to follow dynamic distributions by adding nodes and deleting them in the network during clustering. First, two random nodes from the input data are selected and the network competition is started for the highest similarity to the input pattern. During the learning related data nodes are classified as similarities within clusters, however, the number of nodes is increased in order to get the input probability density [2, 13, 18]. SOM maps multi-dimensional data onto lower dimensional subspaces where the geometric relationships between points indicates their similarity, based on a competitive learning and adjusting the weights to be close to the "winning" nodes, by using enough and necessary data in order to develop meaningful clusters [10, 16]. Current UFFNN clustering methods have major problems in low speed and low accuracy of clustering with the high memory complexity of the network, some reasons are [2, 13, 18]: initialization the weights, thresholds and parameters for controlling clustering tasks by using random values, and the suitable values are defined through trial and error after several re-performances of the model and relearning; high dimensional data and huge datasets which cause difficulty in managing new data and noise while pruning causes data details to be lost [20].

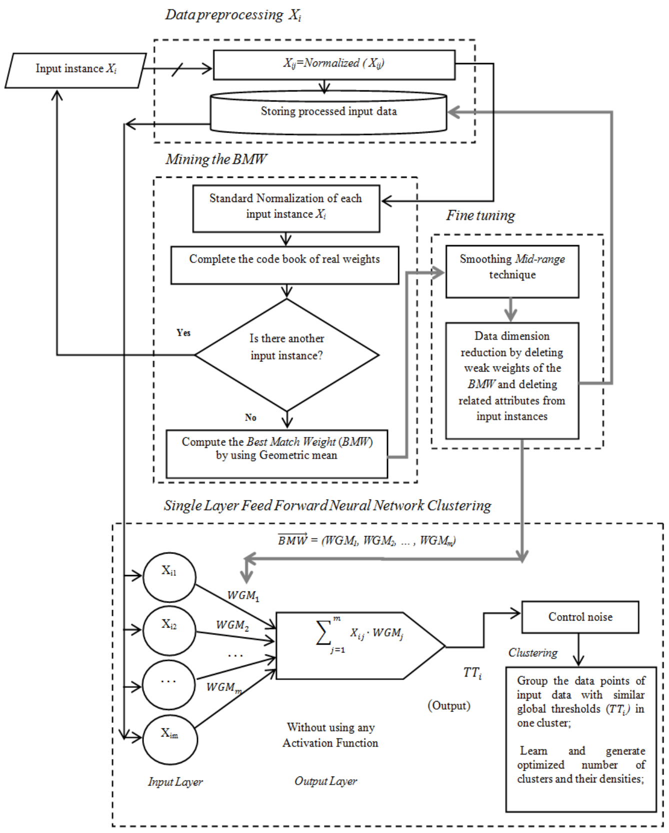
3. Methodology

In order to solve the aforementioned problems, we proposed a Real Unsupervised Feed Forward Neural Network (RUFFNN) clustering. Figure 1 shows its design.

1. *Data Pre-Processing*: mostly pre-processing is the contributing factor in developing efficient techniques for low training time and high accuracy of feed forward neural network clustering [4].

In this study, the MinMax normalization as a data pre-processing technique [4, 13] is used in order to consider the input value in the range (0, 1). Therefore, each attribute X_{ij} is normalized according to Equation 1:

$$X_{ij} = [(X_{ij} - Min(X_{ij}))/ (Max(X_{ij}) - Min(X_{ij}))]\times(1\text{-}0)+0 \tag{1}$$



Data preprocessing X_i
Input instance X_i
X_{ij}=Normalized (X_{ij})
Storing processed input data
Mining the BMW
Standard Normalization of each input instance X_i
Complete the code book of real weights
Yes
Is there another input instance?
No
Compute the *Best Match Weight* (*BMW*) by using Geometric mean
Fine tuning
Smoothing *Mid-range* technique
Data dimension reduction by deleting weak weights of the *BMW* and deleting related attributes from input instances
Single Layer Feed Forward Neural Network Clustering
$\overrightarrow{BMW}$ = (WGM_1, WGM_2, … , WGM_m)
X_{i1}
X_{i2}
…
X_{im}
WGM_1
WGM_2
WGM_m
$\sum_{j=1}^{m} X_{ij} \cdot WGM_j$
TT_i
(Output)
Control noise
Clustering
Group the data points of input data with similar global thresholds (TT_i) in one cluster;
Learn and generate optimized number of clusters and their densities;
Without using any Activation Function
Input Layer
Output Layer

Figure 1. The design of the RUFFNN clustering method.

After this phase, the proposed method considers X_{ij} as the normalized value for the next computations.

2. *Creating a Real Weights Code Book*: the proposed method is based on the SOM and torque vector [3, 21], however, it generates a code book of non-random weights. Each attribute of the input data X_{ij} is normalized based on the Standard Gaussian Distribution (SGD), as shown in Equation 2:

$$SGD(X_{ij}) = (X_{ij} - \mu_i)/\sigma_i \tag{2}$$

μ_i and σ_i are mean and standard deviation of the input data record. Each $SGD(X_{ij})$ shows the distance of each input value of each instance (record) from the mean of the input data X_i. The RUFFNN method considers each $SGD(X_{ij})$ as the weight W_{ij} for that value as shown in Equation 3, therefore, each weight vector of the codebook is computed based on processing on each input data instance and is not at random. This phase can be processed in parallel.

$$W_{ij} = SGD(X_{ij});\ i=1, 2, ..., n;\ j=1, 2, ..., m \tag{3}$$

3. *Mining the Best Match Weight* (*BMW*) *Vector from the Codebook*: the BMW vector is the global geometric mean [14] vector of the weights code book. The BMW consists of the components WGM_j for attributes as geometric mean weight. The WGM_j is computed by taking the n_{th} root of the product of the real weights of each attribute of the input data. The parameter n is the number of input data, i is current number of node of input data; m is the number of attributes and j is the current number of the attribute of input data, as shown in Equations 4 and 5.

$$WGM_j = (W_{1j} . W_{2j} . ... .W_{nj})^{1/n} \tag{4}$$

$$BMW = (WGM_1, WGM_2, ..., WGM_m) \tag{5}$$

The RUFFNN method tries to learn the *BMW* vector as the criterion weight vector. For example, the *BMW* vectors of the Breast Cancer Wisconsin (BCW) and Iris datasets from UCI [5] are as follows:

BMW of BCW={0.132488, 0.095497, 0.096868, 0.10093, 0.097176, 0.141244, 0.105377, 0.107894, 0.122526};

BMW of Iris={0.159178919, 0.3448055, 0.205496179, 0.290519403}

4. *Fine Tuning*: this process refers to modifying the weights accurately in order to succeed better results of clustering the input data, as follows:

   - *Smoothing the Weights*: one of the smoothing technique includes flexible and robust parameters of the FFNN clustering tasks is the weights interconnection to improve speed, accuracy and capability of the training and optimization [11, 22]. *Mid-range* is a popular smoothing technique [11, 22]. Some attributes of the input data have too high weight amounts which may cause them to overlook the high thresholds and high effect the results of clustering. Therefore, when some components of the *BMW* vector are extremely higher than the other components, the *Mid-range* technique can be used. In the *Mid-range* technique, the average of the high weight components of the *BMW* vector is calculated and considered as the Middle range (*Mid-range*). If some components of the *BMW* vector are upper than the *Mid-range*, the method fixes their weights to the *Mid-range* value.

- *Data Dimension Reduction*: high dimensional data and huge data set cause difficulty in handling new data and noise while pruning causes data details to be lost [2, 15]. The RUFFNN method reduces the dimension of data by recognizing the weak and ineffective WGM_j and removing the related attributes, that it affects high speed and low network memory usage complexity [2, 7]. Hence, the weights can be controlled and pruned in advance.

5. *Single Layer FFNN Clustering:* The topology is very simple as Figure 1 shows, it contains of just an input layer with *n* nodes equal the number of attributes and an output layer with just one node. The units of the input layer are fed by the normalized data values from

The data pre-processing phase. Each unit applies a related weight component WGM_j of the *BMW* vector. The output layer consists of one unit with a weighted sum function for computing a threshold as the actual desired output. The RUFFNN training is carried out just in one epoch and is based on real weights, without any class label, parameter for controlling tasks, weight updating, activation function and error function such as mean square error. Due to apply the geometric mean of the weights for computing the *BMW*, the range and properties of the input data values cannot dominate the values of the thresholds. Figure 2 shows an example about each X_{ij} which creates its own torque vector [3, 21] ratio to the global mean or the gravity center of the dataset.

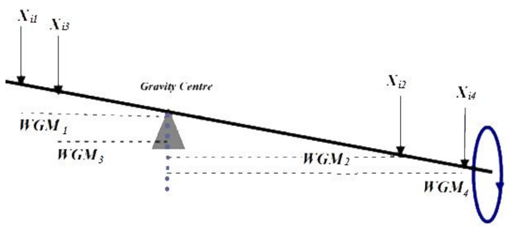


Figure 2. Distribution of normalized input data attributes and their distances from the gravity center of the training data set.

Each X_{ij} by using WGM_j as its arm, shows the distance of X_{ij} from the gravity center of the matrix, and creates a torque vector, which is equal the threshold T_{ij} as shown in Equation 6.

$$T_{ij} = X_{ij} \,.\, WGM_j \tag{6}$$

The vectors are evaluated together and eventually will reach the equilibrium. After equivalence, the Total Threshold (*TT*) of each X_i is computed, as shown in Equations 7 and 8.

$$TT_i = \sum_j X_{ij} \,.\, WGM_j \,; \quad j=1, 2, ..., m \tag{7}$$

$$\text{OR} \quad TT_i = \sum_j T_{ij} \,; \quad j=1, 2, ..., m \tag{8}$$

- *Clustering of the Input Data*: each vector value of the X_i takes place its own position on the torque axis. Therefore, the input data based on their exclusive *TT* lay on the torque axis respectively, and has an exclusive and individual threshold. If there are two input data with the equally TT_i , but different clusters or classes, as error of the clustering method, decrease the clustering accuracy. The RUFFNN considers the input data with near *TT* into one cluster. After clustering, each cluster will be delegated to the special class which is most frequent in the cluster. Figure 3 shows the BCW dataset from UCI repository [5] which is clustered to two clusters by the RUFFNN. The input data point *A* has TT_A=0.004299222 and lies inside of the cluster1 or the cluster of the "Malignant".

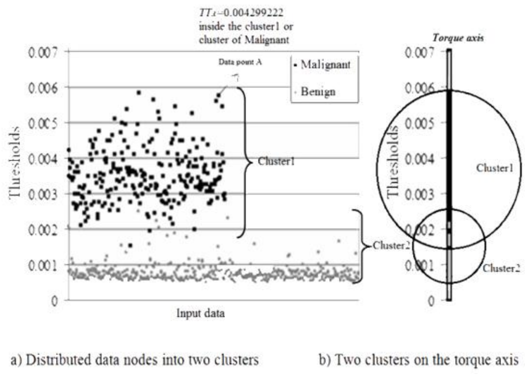


Figure 3. The outlook of clustering the BCW dataset by RUFFNN.

- *Pruning the Noise*: The RUFFNN distinguishes isolated input data point through solitary *TT*, which is not near the *TTs* of other clustered data point. Therefore, the isolated data lies out of the locations of other clusters. The proposed method sets apart these data points as noise and removes them. The action of removing the noise causes high speed and clustering accuracy with low memory usage of the network.

During training, all phases of the proposed method must be performed completely, however, during testing just by having normalized data values of the test set and the *BMW* components from training phase, the single layer FFNN is able to cluster the of the test data, immediately. The algorithm of the RUFFNN clustering method is, as follows:

```
Input :  Data set N;  Output: Clusters of dataset;
Initialize the parameters:
 {
1- { Input a new vector $X_i$;
    //Data preprocessing ;  For all $X_{ij}$ = Normalized($X_{ij}$);
    //Compute the weight  vector for all $X_i$ ;
      $SGD(X_{ij}) = (X_{ij} - \mu_i)/\sigma_i$ ;  $W_{ij} = SGD(X_{ij})$;
    }
2- {// Mining the  Best Match Weight vector (BMW);
      Each $WGM_j=(W_{1j} . W_{2j} . ... . W_{nj})^{1/n}$ ;
      $BMW= (WGM_1,WGM_2,...,WGM_m)$;
3- // Fine tuning through two techniques: Smoothing the
components of the BMW, and Data dimension reduction;
       For all componnts of the BMW: Mid-range($WGM_j$) ;
       Delete attributes with weak $WGM_j$;
4- // Process   of  the single  layer  UFFNN clustering
   // Compute  the Total Threshold of each input data  $X_i$ ;
        $TT_i = \sum X_{ij} . WGM_j$;
     Delete isolated input data as noise with solitary TT ;
     Clustering; Group the $X_i$ with similar TT in one cluster;
     }
   }
```

4. Experimental Results and Comparison

The methods were implemented in Visual C#.Net under Microsoft Windows 7 Professional operating system with 4 GHz Pentium processor. The proposed method was tested for the BCW, Iris and Spambase datasets from the UCI repository [5], as shown in Table 1. The accuracies of the methods were measured through the number of clusters and the quantity of the Correctly Classified Nodes (CCN), which was equal true positive and true negative nodes [6, 8], and showed the total of nodes and their densities, with the correct class in the correct related cluster, in all created clusters by the method. Furthermore, the accuracy was also measured by using the *F-measure* function with 10 folds of the test set. The results were the averages of three time performances.

Table 1. The information of selected datasets in this study from the UCI Repository.

Data Set	Data Set Characteristics	Attribute Characteristics	Number of Instances	Number of Attributes	Classes
Breast Cancer Wisconsin (Original)	Multi variable	Integer	699	10	Two classes: benign and malignant
Iris	Multi variable	Real	150	4	Three classes: Iris Setosa, Iris Versicolour and Iris Virginica
Spambase	Multi variable	Integer-Real	4601	57	Two classes: Spam and Non-Spam

4.1. Breast Cancer Wisconsin Data Set

The BCW dataset is clustered by the RUFFNN, as shown in Figure 3, and the result is compared with the related methods [6, 8], as shown in Table 2. That, the SOM produced 660 *CCN* after 20 epochs with 96.63% density of *CCN*, as the best UFFNN result. The RUFFNN after one epoch in 8.7262 milliseconds, had 660 *CCN*, 96.63% density of *CCN* and 98.06% of the *F-measure*. All clustering methods show two clusters for this dataset. The Back Propagation Network (BPN) [23], as a popular supervised FFNN, classified this dataset after 1000 epochs with 99.28% by F-measure accuracy.

Table 2. Comparison of clustering the BCW data set by different UFFNN methods.

The Clustering Method	CCN	Density of CCN %	Epoch
SOM	660	96.63	20
K-Means	657	96.19	20
Neural Gas	657	96.19	20
GNG	477	69.84	5
RUFFNN	660	96.63	1

4.2. Iris Data Set

The Iris dataset is clustered by the RUFFNN, as shown in Figure 4, and the result is compared with the related methods [6, 8], as shown in Table 3. That, the *CCN* of the NG was 139 after 20 epochs with 92.67% density of *CCN*, as the best UFFNN result [7]. The RUFFNN after one epoch in 4.1744 milliseconds, had 145 *CCN*, 96.67% density of *CCN*, and 97.33% of the *F-measure.*

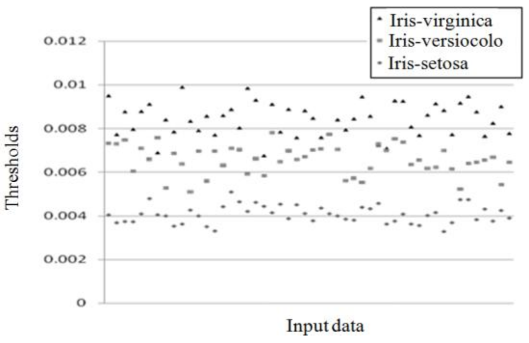


Figure 4. The clusters of Iris data set by the RUFFNN model.

Table 3. Comparison of clustering the Iris data set by different UFFNN methods.

The Clustering Method	CCN	Density of CCN %	Epoch
SOM	123	82.00	20
K-Means	134	89.33	20
Neural Gas	139	92.67	20
GNG	135	90.00	10
RUFFNN	145	96.67	1

All clustering methods show three clusters for this dataset. The BPN classified this dataset after 14 epochs with 94% by the *F-measure* accuracy.

4.3. Spambase Data Set

The Spambase dataset is clustered by the RUFFNN, as shown in Figure 5, and the result is compared with the related methods [6, 8], as shown in Table 4. That, the SOM produced 1210 CCN after 20 epochs with 26.30% density of *CCN*, as the best UFFNN results. The RUFFNN clusters this dataset, after one epoch in 337.1057 milliseconds, with 2731 CCN, 59.36% density of CCN, and 66.46% of the F-measure. All clustering methods show two clusters for this dataset. The BPN classified the Spambase after 2000 epochs with 80% accuracy by the F-measure.

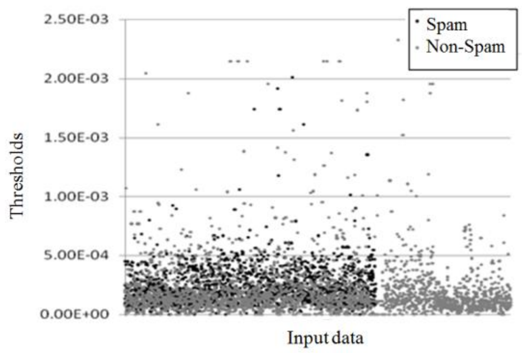


Figure 5. The clusters of the Spambase dataset by the RUFFNN method.

Table 4. Comparison of clustering the Spambase dataset by different UFFNN methods.

The Clustering Method	CCN	Density of CCN %	Epoch
SOM	1210	26.30	20
K-Means	1083	23.54	20
Neural Gas	1050	22.82	20
GNG	967	21.02	5
RUFFNN	2731	59.36	1

5. Discussion and Conclusions

To overcome the problems of the low training speed and clustering accuracy, and high memory usage, the RUFFNN clustering method with one epoch training, data dimensionality reduction and controlling noise abilities was proposed. Table 5 shows time and memory complexities of the RUFFNN and some related UFFNN methods [4].

Table 5. Comparison of time complexities and memory complexities of the RUFFNN method with some related methods.

Method	Time Complexity	Memory Complexity
K-means	$O(c.k.n.m)$	$O((n+k).m.s_m)$
NG	$O(c.n^2.m)$	$O(c.n^2.m.s_m)$
GNG	$O(c.n^2.m)$	$O(c.n^2.m.s_m)$
SOM	$O(c.n.m^2)$	$O(c.n.m^2.s_m)$
RUFFNN	$O(n.m)$	$O(n.m.s_m)$

The RUFFNN is a linear clustering method has the time and memory complexities of $O(n.m)$ and $O(n.m.s_m)$, respectively. The parameters c, k, n, m, s_m are the number of epochs, clusters, nodes, attributes and size of each attribute. The experimental results showed the superior outcomes of the RUFFNN method. For future work, an online dynamic RUFFNN is suggested by improving the RUFFNN method.

References

[1] Andonie R. and Kovalerchuk B., "Neural Networks for Data Mining: Constrains and Open Problems," *in Proceeding of Ellensburg European Symposium on Artificial Neural Networks*, Bruges, pp. 449-458, 2007.

[2] Asadi R. and Abdul-Kareem S., "Review of Feed Forward Neural Network Classification Preprocessing Techniques," *in Proceeding of 3rd Internationa Conference Mathematical Sciences*, Kuala Lumpur, pp. 567-573, 2014.

[3] Asadi R., Abdul-Kareem S., Asadi M., and Asadi S., "A Single-Layer Semi-Supervised Feed Forward Neural Network Clustering Method," *Malaysian Journal of Computer Science*, vol. 28, no. 3, pp. 189-212, 2015.

[4] Asadi R., Hasan H., Sameem A., and Abdul-Kareem S., "Review of Current Online Dynamic Unsupervised Feed Forward Neural Network Classification," *in Proceeding of Computer Science and Electronics Engineering*, Kuala Lumpur, pp. 21-28, 2014.

[5] Asuncion A. and Newman D., http://www.ics.uci.edu/~mlearn/MLRepository, Last Visited 2007.

[6] Camastra F. and Verri A., "A Novel Kernel Method for Clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 801-805, 2005.

[7] Chattopadhyay M., Dan P., and Mazumdar S., "Principal Component Analysis and Self-Organizing Map for Visual Clustering of Machine-Part Cell Formation in Cellular Manufacturing System," *Systems Research Forum*, vol. 5, no. 1, pp. 1-25, 2011.

[8] Costa J. and Oliveira R., "Cluster Analysis using Growing Neural Gas and Graph Partitioning," *International Joint Conference on Neural Networks*, Orlando, pp. 3051-3056, 2007.

[9] Germano, T., http://davis.wpi.edu/~matt/courses/soms, Last Visited 1999.

[10] Goebel M. and Gruenwald L., "A Survey of Data Mining and Knowledge Discovery Software Tools," *ACM SIGKDD Explorations Newsletter*, vol. 1, no. 1, pp. 20-33, 1999.

[11] Gui V., Vasiu R., and Bojković Z., "A New Operator for Image Enhancement," *Facta Universitatis-Series: Electronics and Energetics*, vol. 14, no. 1, pp. 109-117, 2001.

[12] Han J., Pei J., and Kamber M., *Data Mining, Southeast Asia Edition: Concepts and Techniques*, Morgan Kaufmann, 2006.

[13] Han J. and Kamber M., *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2006.

[14] Jacquier E., Kane A., and Marcus A., "Geometric or Arithmetic Mean: A Reconsideration,"

Financial Analysts Journal, vol. 59, no. 6, pp. 46-53, 2003.

[15] Kohonen T., *Self-Organizing Maps*, Springer, 1997.

[16] Linde Y., Buzo A., and Gray R., "An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84-95, 1980.

[17] Martinetz T ., Berkovich S ., and Schulten K ., "Neural-Gas" Network for Vector Quantization and its Application to Time-Series Prediction," *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 558-569, 1993.

[18] Meskine F. and Bahloul S., "Privacy Preserving K-means Clustering: A Survey Research," *The International Arab Journal of Information Technology*, vol. 9, no. 2, pp. 194-200, 2012.

[19] Murugappan I. and Vasudev M., "PCFA: Mining of Projected Clusters in High Dimensional Data Using Modified FCM Algorithm," *The International Arab Journal of Information Technology*, vol. 11, no. 2, pp. 168-177, 2014.

[20] Serway R. and Jewett J., *Physics for Scientists and Engineers with Modern Physics*, Brooks Cole, 2013.

[21] Tesauro G., Touretzky D., and Leen T., *Advances in Neural Information Processing Systems 7*, MIT Press, 1995.

[22] Tong X., Qi L., Wu F., and Zhou H., "A Smoothing Method for Solving Portfolio Optimization with CVaR and Applications in Allocation of Generation Asset," *Applied Mathematics and Computation*, vol. 216, no. 6, pp. 1723-1740, 2010.

[23] Werbos P., *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, Harvard University, 1974.

Roya Asadi is a PhD of artificial intelligence (neural network) in computer science from the University of Malaya. Her interests include Artificial Neural Network modeling, Medical Informatics, Machine Learning, Data Mining, Image Processing.

Mitra Asadi is a PhD candidate of Entrepreneurship Technology in Islamic Azad University- Central Tehran Branch, Iran. Her interests include: English Language Translation, Entrepreneurship Technology, and Management.

Shokoofeh Asadi is Master of Agricultural Management Engineering from University of Science and research in Tehran- Iran. Her interests include: English Language Translation, Biological and Agricultural engineering, Management and leadership.