Creation of an Ensemble: Diversity Production Based Approach

Hamid Parvin, Zahra Rezaei, Sajad Parvin
Nourabad Mamasani Branch
Islamic Azad University
Nourabad Mamasani
Iran

ABSTRACT: *Generally in design of combinational classifier systems, the more diverse the results of the classifiers, the more appropriate final result. In this paper, a new method for combining classifiers is proposed which its main idea is heuristic retraining of classifiers. Specifically, in the new method named Combinational Classifiers using Heuristic Retraining (CCHR) which proposes a new way for generating diversity in ensemble pool, a classifier is first run, then, focusing on the drawbacks of this base classifier, other classifiers are retrained heuristically. Each of these classifiers looks at the data with its own attitude. The main concentration in the retrained classifiers is to leverage the error-prone data. So, retrained classifiers usually have different votes about the sample points which are close to boundaries and may be likely erroneous. Experiments show significant improvements in terms of accuracies of consensus classification. This study also investigates that focusing on which crucial data points can lead to more performance in base classifiers. Also, this study shows that adding the number of all "difficult" data points like boosting method, does not always cause a better performance. The experimental results show that the performance of the proposed algorithm outperforms some of the best methods in the literature. So empirically, the authors claim that forcing crucial data points to the training set as well as eliminating them from the training set can yield to the more accurate results, conditionally.*

Keywords: Classifier Fusion, Heuristic Retraining, Neural Network Ensemble

Received: 3 October 2011, Revised 23 November 2011, Accepted 28 November 2011

1. Introduction

Nowadays, usage of recognition systems has found many applications in almost all fields [23]. Many researches are done to improve their performance [23]. Most of these algorithms have provided good performance for specific problem, but they have not enough robustness for other problems. Because of the difficulty that these algorithms are faced to, recent researches are directed to the combinational methods that have more power, robustness, resistance, accuracy and generality [23]. Although the accuracy of the classifier ensemble is not always better than the most accurate classifier in ensemble pool, its accuracy is never less than average accuracy of them [2]. Combination of multiple classifiers, CMC, can be considered as a general solution method for pattern recognition problems [21]. Inputs of CMC are result of separate classifiers and output of CMC is their final combined decisions. [4] articulates that the rationale behind the growing interest in multiple classifier systems (MCSs) is that the classical approach to design a pattern recognition system, which focuses on the search for the best individual classifier, has some serious drawbacks. The main drawback is that the best individual classifier for the classification task at hand is very difficult to identify, unless deep prior knowledge is available for such a task (Duda 2000). In addition, [5] [17] express that it is not possible to exploit the complementary discriminatory information that other classifiers may encapsulate

with only a single classifier. It is worth noting that the motivations in favor of MCS strongly resemble those of a "*hybrid*" intelligent system (Kandel and Langholz 1992). The obvious reason for this is that MCS can be regarded as a special-purpose hybrid intelligent system.

In General, it is an ever-true sentence that "combining the diverse classifiers any of which performs better than a random results in a better classification performance". Diversity is always considered as a very important concept in classifier ensemble methodology. It is considered as the most effective factor in succeeding an ensemble. The diversity in an ensemble refers to the amount of differences in the outputs of its components (classifiers) in deciding for a given sample. Assume an example dataset with two classes. Indeed the diversity concept for an ensemble of two classifiers refers to the probability that they may produce two dissimilar results for an arbitrary input sample. The diversity concept for an ensemble of three classifiers refers to the probability that one of them produces dissimilar result from the two others for an arbitrary input sample. It is worthy to mention that the diversity can converge to 0.5 and 0.66 in the ensembles of two and three classifiers respectively. Although reaching the more diverse ensemble of classifiers is generally handful, it is harmful in boundary limit. It is very important dilemma in classifier ensemble field: the ensemble of accurate/diverse classifiers can be the best. It means that although the more diverse classifiers, the better ensemble, it is provided that the classifiers are better than random.

An Artificial Neural Network (ANN) is a model which is to be configured to be able to produce the desired set of outputs, given an arbitrary set of inputs. An ANN generally composed of two basic elements: (a) neurons and (b) connections. Indeed each ANN is a set of neurons with some connections between them. From another perspective an ANN contains two distinct views: (a) topology and (b) learning. The topology of an ANN is about the existence or nonexistence of a connection. The learning in an ANN is to determine the strengths of the topology connections. One of the most representatives of ANNs is MultiLayer Perceptron. Various methods of setting the strength of connections in an MLP exist. One way is to set the weights explicitly, using a prior knowledge. Another way is to 'train' the MLP, feeding it by teaching patterns and then letting it change its weights according to some learning rule. In this paper the MLP is used as one of the base classifiers.

Decision Tree (DT) is considered as one of the most versatile classifiers in the machine learning field. DT is considered as one of unstable classifiers. It means that it can converge to different solutions in successive trainings on same dataset with same initializations. It uses a tree-like graph or model of decisions. The kind of its knowledge representation is appropriate for experts to understand what it does [24].

The authors believe that Combinational methods usually result in the improvement of classification, because classifiers with different features and methodologies can cover drawbacks of each other [23]. Kuncheva using Condorcet theorem has shown that combination of classifiers can usually operate better than single classifier. It means if more diverse classifiers are used in the ensemble, then error of them can considerably be reduced. Different categorizations of combinational classifier systems are represented in [4] [5] [6] (Puuronen et al. 2001). Valentini and Masouli divide methods of combining classifiers into two categories: generative methods, non-generative methods. In generative methods, a set of base classifiers are created by a set of base algorithms or by manipulating dataset. This is done in order to reinforce diversity of base classifiers. Generally, all methods which aggregate the primary results of the fixed independent classifiers are non-generative. They are also named fusion methods (Skalak 1994; Kohonen 1990).

Neural network ensembles as an example of combinational methods in classifiers are also becoming a hot spot in machine learning and data mining recently [7]. Many researchers have shown that simply combining the output of many neural networks can generate more accurate predictions than that of any of the individual networks. Theoretical and empirical works show that a good ensemble is one where the individual networks have both accuracy and diversity, namely the individual networks make their errors on difference parts of the input space [8] [9].

2. Background

In generative methods, diversity is usually made using two groups of methods. One group of these methods obtains diverse individuals by training classifiers on different training set, such as bagging [10], boosting (Freund and [11], cross validation [9] and using artificial training examples [12]. More details about these methods will be appeared in section 2.

Another group of methods for creating diversity employs different structures, different initial weighing, different parameters and different base classifiers to obtain ensemble individuals. For example, (Rosen 1996) adapted the training algorithm of the

network by introducing a penalty term to encourage individual networks to be decorrelated. [14] used negative correlation learning to generate negatively correlated individual neural network.

The third group is named selective approach group where the diverse components are selected from a number of trained accurate networks. For example, [15] proposed a generic algorithm to search for a highly diverse set of accurate networks. [16] proposed a pruning algorithm to eliminate redundant classifiers. [17] proposed another selective algorithm based on bias/variance decomposition. GASEN proposed by [18] and PSO based approach proposed by [19] also were introduced to select the ensemble components. In the rest of this paper, a new method to obtain diverse classifiers is proposed which uses manipulation of dataset structures.

Inspired from boosting method, in this paper a new sort of generative approaches is presented which creates new training sets from the original one. The base classifiers are trained focusing on the crucial and error prone data of the training set. This new approach which is called "Combination of Classifiers using Heuristic Retraining, CCHR" is described in section 2 in detail. In fact, the question of "how to create a number of diverse classifiers?" is answered in that section. Section 3 addresses the empirical studies in which we show the great accuracy and robustness of CCHR method for different datasets. Finally, section 4 discusses the concluding remarks.

2.1 Artificial Neural Network

A first wave of interest in ANN (also known as 'connectionist models' or 'parallel distributed processing') emerged after the introduction of simplified neurons by McCulloch and Pitts in 1943. These neurons were presented as models of biological neurons and as conceptual components for circuits that could perform computational tasks. Each unit of an ANN performs a relatively simple job: receive input from neighbors or external sources and use this to compute an output signal which is propagated to other units. Apart from this processing, a second task is the adjustment of the weights. The system is inherently parallel in the sense that many units can carry out their computations at the same time. Within neural systems it is useful to distinguish three types of units: input units (indicated by an index i) which receive data from outside the ANN, output units (indicated by an index o) which send data out of the ANN, and hidden units (indicated by an index h) whose input and output signals remain within the ANN. During operation, units can be updated either synchronously or asynchronously. With synchronous updating, all units update their activation simultaneously; with asynchronous updating, each unit has a (usually fixed) probability of updating its activation at a time t, and usually only one unit will be able to do this at a time. In some cases the latter model has some advantages.

An ANN has to be configured such that the application of a set of inputs produces the desired set of outputs. Various methods to set the strengths of the connections exist. One way is to set the weights explicitly, using a priori knowledge. Another way is to 'train' the ANN by feeding it teaching patterns and letting it change its weights according to some learning rule. For example, the weights are updated according to the gradient of the error function. For further study the reader must refer to an ANN book such as Haykin's book on theory of ANN [25].

2.2 Decision Tree Learning

DT as a machine learning tool uses a tree-like graph or model to operate deciding on a specific goal. DT learning is a data mining technique which creates a model to predict the value of the goal or class based on input variables. Interior nodes are the representative of the input variables and the leaves are the representative of the target value. By splitting the source set into subsets based on their values, DT can be learned. Learning process is done for each subset by recursive partitioning. This process continues until all remain features in subset has the same value for our goal or until there is no improvement in Entropy. Entropy is a measure of the uncertainty associated with a random variable.

Data comes in records of the form: $(x,Y) = (x1, x2, x3,\ldots, xn,Y)$. The dependent variable, Y, is the target variable that we are trying to understand, classify or generalize. The vector x is composed of the input variables, x1, x2, x3 etc., that are used for that task. To clarify that what the DT learning is, consider Figure 1. Figure 1 has 3 attributes Refund, Marital Status and Taxable Income and our goal is cheat status. We should recognize if someone cheats by the help of our 3 attributes. To do learn process, attributes split into subsets. Figure 2 shows the process tendency. First, we split our source by the Refund and then MarSt and TaxInc.

For making rules from a decision tree, we must go upward from leaves as our antecedent to root as our consequent. For example consider Figure 2. Rules such as following are apprehensible. We can use these rules such as what we have in Association Rule Mining.

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Refund=YesÞcheat=No

TaxInc<80, MarSt= (Single or Divorce), Refund=NoÞcheat=No

TaxInc>80, MarSt= (Single or Divorce), Refund=NoÞcheat=Yes

Refund=No, MarSt=MarriedÞcheat=No

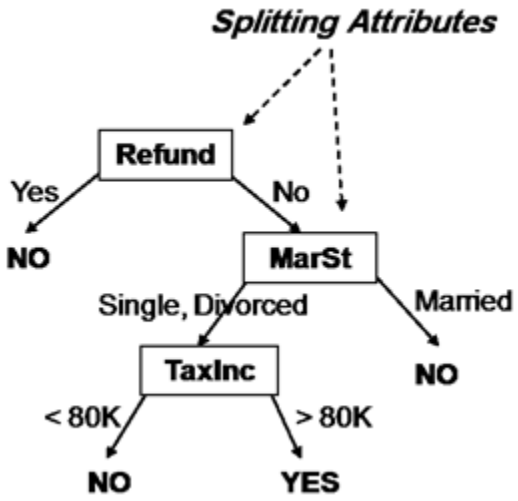

Figure 2. The Process Tendency for Figure 1

2.3 K-Nearest Neighbor Algorithm

k-nearest neighbor algorithm (k-NN) is a method for classifying objects based on closest training examples in the feature space. k-NN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification. The k-nearest neighbor algorithm is amongst the simplest of all machine learning algorithms: an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of its nearest neighbor.

As it is obvious, the k-NN classifier is a stable classifier. A stable classifier is the one converge to an identical classifier apart from its training initialization. It means the 2 consecutive trainings of the k-NN algorithm with identical k value, results in two

classifiers with the same performance. This is not valid for the MLP and DT classifiers. We use 1-NN as a base classifier in the paper.

3. Proposed Method

The main idea of the proposed method is heuristically retraining of MLPs on different sets of data. In this method, base MLPs are trained on some possible permutations of 3 datasets named: TS, NS, and EPS. They are abbreviation for Train Set, Neighbor Set and Error-Prone Set, respectively. In the next step, the results of all these base classifiers are combined using simple average method.

3.1 Preparing Different Sets from the Main Dataset

Firstly, a simple Multi Layer Perceptron is trained on TS. Then, using this neural net (MLP), the data which may be misclassified are recognized. This work is done for different perspectives of training-test datasets. It means that it is tried to detect all error-prone data on TS. It can be implemented using leave-one-out technique and Cross-Validation.
In cross-validation which is also called the rotation method, an integer K (preferably a factor of N) is chosen and the dataset is randomly divided into K subsets of size N/K. Then, a classifier is trained on dataset-{i-th subset of the dataset} and evaluated using i-th subset. This procedure is repeated K times, choosing a different part for testing each time. When N=K, the method is called the leave-one-out or U-method.

In this paper, the dataset is divided into three partitions: training, evaluation and test sets. The leave-one-out technique is applied to train set for obtaining the Error-Prone Set, EPS. As it is mentioned, using leave-one-out technique an MLP on TS-{one of its data} is trained and evaluate whether that MLP misclassifies that out data or not. If it is misclassified we add it to EPS. As it is obvious, we run this work in number of items in training set. If training dataset is very large, the cross-validation technique can be used instead of leave-one-out technique, too.

In this study, the cross-validation technique is applied to {train set + validation set} for deriving the neighbor set, NS. Whereas the cross-validation is an iterative technique, the K-1 subset is considered to be as train set and the one subset as validation set, for each iteration. The errors in validation set are added to error set. In the next step, for each instance of error set, the nearest neighbor instance which belongs to the same label of that instance is found. This neighbor set is named NS.

Num.	TS	Resultant Classifier
1	TS	Creation of base classifiers
2	TS+NS	Classification by complex boundaries with more concentration on crucial points and neighbor of errors (NS)
3	TS+EPS+NS	Classification by complex boundaries with more concentration on error prone(EPS) and crucial points(NS)
4	TS-EPS+NS	Classification by simple boundaries with more concentration on crucial points
5	TS+EPS	Classification by complex boundaries with more concentration on error prone points (EPS)
6	TS-EPS	Classification by very simple boundaries

Table 1. Different Data Combinations and Reasons of their Usages

3.2 Creating an Ensemble of Diverse Classifiers

The EPS and NS are obtained from previous section. In this section some MLPs are trained based them. The more diverse and accurate base classifiers, the better results in final. So, some combinations as shown in Table 1 are used to create diversity in our ensemble. The used permutations and the reasons of their usage are shown in Table 1. Training of MLPs, using the combinations in Table 1, results in the classifiers that each of them concentrates on a special aspect of data. This can result in very good diversity in the ensemble.

In this paper, 6 MLPs are trained using different data according to Table 1. Their results are used in our classifier ensemble. Our proposed algorithm is shown in Figure 3.

```
NS: Neighbor Set, NS={};
EPS: Error Prone Set, EPS={};
Program CCHR
        1. NS=FindNS();          //calculating NS
        2. EPS=FindEPS();        // calculating EPS
        3. Train 6 MLPs according to Table 1.
        4. Combine the results using  simple average.
```

Figure 3. The Proposed CCHR Algorithm

3.3 Combining Classifiers

After creating diverse classifiers for our classifier ensemble, the next step is finding a method to fuse their results and make final decision. The part of making final decision is named combiner part. There are many different combiners. Combination method of base classifier decisions depend on their output type. Some traditional methods of classifier fusion which are based on soft/ fuzzy outputs are as below:

Majority vote: assume that we have k classifiers. Classifier ensemble vote to class j if a little more than half of base classifiers vote to class j.

Simple average: the average of results of separate classifiers is calculated and then the class that has the most average value is selected as final decision.

Weighted average: it is like simple average except that a weight for each classifier is used for calculating that average.

Train set	Classifier number as Table 1						CCHR
	1	2	3	4	5	6	
70%	95.01	95.20	95.20	94.97	**95.37**	95.07	**95.97**
50%	95.95	95.75	95.87	95.89	**96.24**	95.78	**96.60**
30%	93.57	93.26	93.17	93.64	**93.99**	93.48	**95.22**

Table 2. Average results on Iris dataset

Train set	Classifier number as Table 1						CCHR
	1	2	3	4	5	6	
50%	91.58	91.64	92.66	91.98	**93.77**	91.29	**96.74**
30%	88.72	**88.91**	89.31	88.23	88.83	88.60	**93.76**

Table 3. Average results on Wine dataset

Table 4 shows the result of performance of classification using our method and traditional methods comparatively.

As it is obvious from Table 4, recognition ratio is improved considerably. Because of low number of features and records in Iris, the improvement is more significant on Wine dataset.

Table 5 shows the results of performance of classification accuracy of CCHR method and other traditional methods comparatively. These results are average of the ten independent runs of the algorithm. In this comparison, the parameter K in K-Nearest

Classifier Type	Wine		Iris		
	50%	30%	70%	50%	30%
MLP	91.58	88.72	95.01	95.95	93.37
KNN	71.36	68.73	95.05	94.73	95.11
CCHR	**96.74**	**93.76**	**95.97**	**96.60**	**95.22**

Table 4. CCHR vs. other methods

	Wine			Iris		
	Train 30%	Train 50%	Train 70%	Train 30%	Train 50%	Train 70%
KNN	69.31	69.26	69.22	94.86	95.20	95.32
MLP	88.72	91.58	93.09	93.37	95.95	95.01
Simple Ensemble	92.70	94.05	95.41	94.77	96.00	95.03
Random Forest	88.32	93.37	95.56	91.52	94.67	**96.22**
Arc-X41	**96.4**	96.13	96.42	94.86	94.07	95.33
Arc-X42	95.52	95.73	96.22	**95.33**	96.20	96.07
CCHR	93.76	**96.74**	**96.56**	95.22	**96.60**	95.97

Table 5. CCHR vs. other ensemble methods

Neighbor algorithm, KNN, is set to one. Also, the average accuracy of KNN method is reported over the 100 independent runs by randomly selecting a part of data as the training set, each time. To validate the CCHR method with harder benchmarks, an ensemble of simple MLPs is also implemented. These MLPs have the same structural parameters of the base MLPs of CCHR, i.e. two hidden layer with 10 and 5 neurons respectively in each of them. Like what is in the CCHR method, the voting method is chosen for combining their results.

The CCHR algorithm is compared with the two state of the art combination methods: random forest and boosting. Here, the ensemble size of the random forest is 21. The ensemble size for Arc-$X4_1$ is 5 classifiers. While the ensemble size for Arc-$X4_2$ is 11 classifiers.

5. Conclusion

In this paper, a new method for improving performance of combinational classifier systems, CCHR, is proposed. CCHR is based on heuristic retraining of base classifiers on different datasets as training data. Also, it is observed that different datasets result in different classifiers. It is shown that the classifiers with complex boundaries and also concentrate on error-prone data act better than others. It shows that emphasizing on crucial data causes improvement in results. With regard to the obtained results, we can conclude that the method 5 that is trained on {TS+EPS} is relatively more robust than other methods. We can consider this method as the best way for heuristically retraining of MLP. Also we showed that usage of different datasets causes to quite diverse classifiers. The higher accuracy of their ensemble validates this fact.

Another interesting conclusion of this paper is that emphasizing on the boundary data points, as boosting algorithm is not always very good. Although, boosting of the boundary data points in many cases is good, there are some cases of datasets where elimination of such points is better. The Monk's problem is one of such cases which deleting error-prone data leads to

better results. Also, in data mining tasks which deal with huge data, the small size of ensemble is very interesting which is satisfied in the CCHR method as well.

References

List and number all bibliographical references in 10-point Times New Roman, single-spaced, at the end of your paper. For example, [1] is for a journal paper, [2] is for a book and [3] is for a conference(symposium) paper.

[1] Gunter, S., Bunke, H. (2002). Creation of classifier ensembles for handwritten word recognition using feature selection algorithms, IWFHR 2002 on January 15.

[2] Kuncheva, L. I. (2005). Combining Pattern Classifiers, Methods and Algorithms, New York: Wiley.

[3] Shapley, L., Grofman, B. (1984). Optimizing group judgemental accuracy in the presence of interdependencies, *Public Choice*, 43, 329–343.

[4] Roli, F., Kittler, J. (2001). Editors. Proc. 2nd International Workshop on Multiple Classifier Systems (MCS 2001), V. 2096 of Lecture Notes in Computer Science LNCS Springer- Verlag, Cambridge, UK.

[5] Roli, F., Kittler, J. (2002). Editors. Proc. 3rd Int. Workshop on Multiple Classifier Systems(MCS 2002), V. 2364 of Lecture Notes in Computer Science LNCS Springer Verlag, Cagliari, Italy.

[6] Lam, L. (2000). Classifier combinations: implementations and theoretical issues. In J. Kittler and F. Roli, editors, Multiple Classifier Systems, V. 1857 of Lecture Notes in Computer Science, Cagliari, Italy, Springer, p. 78–86.

[7] Qiang, F., Shang-xu, H., Sheng-ying, Z. (2005). Clustering-based selective neural network ensemble, *Journal of Zhejiang University SCIENCE*, Fu et al. / J Zhejiang Univ SCI 6A, (5) 387-392.

[8] Hansen, L. K., Salamon, P. (1990). Neural network ensembles, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 12 (10) 993-1001.

[9] Krogh, A., Vedelsdy, J.(1995). Neural Network Ensembles Cross Validation, and Active Learning. *In*: Tesauro, G., Touretzky, D., Leen, T.(Eds.), Advances in Neural Information Processing Systems, V. 7. MIT Press, Cambridge, MA, p.231-238.

[10] Breiman, L. (1996). Bagging predictors, *Machine Learning*, 24 (2) 123-140.

[11] Schapire, R. E. (1990). The strength of weak learn ability. Machine Learning, 5 (2) 1971-227.

[12] Melville, P., Mooney, R. (2003). Constructing Diverse Classifier Ensembles Using Artificial Training Examples, *In:* Proc. of the IJCAI-2003, Acapulco, Mexico, p.505-510.

[13] Rosen, B. E. (1996). Ensemble learning using decorrelated neural network, *Connection Science*, 8 (3-4) 373-384.

[14] Liu, Y., Yao, X. (2000). Evolutionary ensembles with negative correlation learning. *IEEE Trans. Evolutionary Computation*, 4 (4) 380-387.

[15] Opitz, D., Shavlik, J. (1996). Actively searching for an effective neural network ensemble. Connection Science, 8 (3-4) 337-353.

[16] Lazarevic, A., Obradovic, Z. (2001). Effective pruning of neural network classifier ensembles, *In*: Proc. International Joint Conference on Neural Networks, 2, 796-801.

[17] Navone, H. D., Verdes P. F., Granitto, P. M., Ceccatto, H. A. (2000). Selecting Diverse Members of Neural Network Ensembles. *In:* Proc. 16th Brazilian Symposium on Neural Networks, p.255-260.

[18] Zhou, Z. H., Wu, J. X., Jiang, Y., Chen, S.F. (2001). Genetic algorithm based selective neural network ensemble, *In*: Proc. 17th International Joint Conference on Artificial Intelligence, 2, 797-802.

[19] Fu, Q., Hu, S. X., Zhao, S. Y. (2004). A PSO-based approach for neural network ensemble. *Journal of Zhejiang University* (Engineering Science), 38 (12) 1596-1600 (in Chinese).

[20] Minaei-Bidgoli, B., Kortemeyer, G. and Punch, W. F. (2004). Optimizing Classification Ensembles via a Genetic Algorithm for a Web-based Educational System, (SSPR /SPR 2004), Lecture Notes in Computer Science (LNCS), V. 3138, Springer-Verlag, p. 397-406.

[21] Saberi, A., Vahidi, M., Minaei-Bidgoli, B. (2007). Learn to Detect Phishing Scams Using Learning and Ensemble Methods. IEEE/WIC/ACM International Conference on Intelligent Agent Technology, Workshops (IAT 07), p. 311-314, Silicon Valley, USA, November 2–5.

[22] Minaei-Bidgoli, B., Kortemeyer, G., Punch, W. F. (2004). Mining Feature Importance: Applying Evolutionary Algorithms within a Web-Based Educational System, *In*: Proc. of the Int. Conf. on Cybernetics and Information Technologies, Systems and Applications, CITSA.

[23] Parvin, H., Alizadeh, H., Fathi, M., Minaei-Bidgoli, B. (2008). Improved Face Detection Using Spatial Histogram Features, The 2008 Int. Conf. on Image Processing, Computer Vision, and Pattern Recognition (IPCV'08), Las Vegas, Nevada, USA, July 14-17.

[24] Yang, T. (2006). International Journal of Computational Cognition 4(4), 34–46.

[25] Haykin, S. (1999). Neural Networks, a comprehensive foundation. Prentice Hall International.

Authors Biography

Hamid Parvin received his B.Sc. and MSc. degrees in computer engineering from Chamran University, Ahvaz, in 2006 and Iran University of Science and Technology (IUST), Tehran, in 2008, respectively. Now, he is a PhD student in computer engineering with specialization in Artificial Intelligence in Iran University of Science and Technology, Iran. His research interests include pattern recognition, machine learning, particularly, data clustering, classification, and ensemble methods.