A Method to Construct Diverse Classifiers

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

ABSTRACT: *Usage of recognition systems has found many applications in almost all fields. Generally in design of combinational classifier systems, the more diverse the results of the classifiers, the more appropriate final result. However, Most of classification algorithms have obtained good performance for specific problems; they have not enough robustness for other problems. Combination of multiple classifiers can be considered as a general solution method for pattern recognition problems. It has been shown that combination of classifiers can usually operate better than single classifier provided that its components are independent or they have diverse outputs. It was shown that the necessary diversity of an ensemble can be achieved manipulation of data set features. We also propose a new method of creating this diversity. The ensemble created by proposed method may not always outperforms all classifiers existing in it, it is always possesses the diversity needed for creation of ensemble, and consequently it always outperforms the simple classifier.*

Keywords: Classifier Fusion, Heuristic Retraining, Neural Network Ensemble

Received: 3 September 2011, Revised 19 November 2011, Accepted 1 December 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. [6] 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 [26]. In addition, [5] 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 [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, [13] 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.

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.

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

Figure 1. An exemplary raw data

Data comes in records of the form: (x,Y) = (x1, x2, x3,…, 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.

Refund = Yes $\Rightarrow$ cheat = No

TaxInc < 80, MarSt = (Single or Divorce), Refund = No $\Rightarrow$ cheat = No

TaxInc > 80, MarSt = (Single or Divorce), Refund = No $\Rightarrow$ cheat = Yes

Refund = No, MarSt = Married $\Rightarrow$ cheat = No

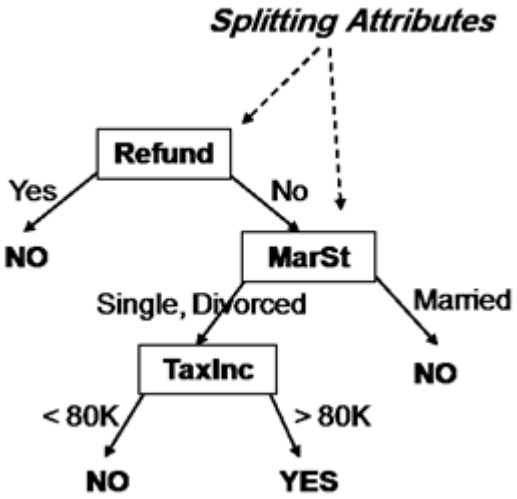


Figure 2. The process tendency for Figure 1

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.

2.4 Support Vector Machine

A support vector machine (SVM) is a concept in statistics and computer science for a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis. The standard SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the input, making the SVM a non-probabilistic binary linear classifier. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the

examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. The original SVM algorithm was invented by Vladimir Vapnik and the current standard incarnation (soft margin) was proposed by [27].

```
Proposed Algorthim(original data set);
validation data, training data, test data = extract (original data set);
for i=1 to number_of_classes
        data_of_class_validation(i)=extract_data_of_each_class(validation data);
end for
for c=1 to max_iteration
        train(classifier, training data, validation set);
        error=computer_error_on_each_class(classifier, validation set);
        for i=1 to number_of_classes
                if error(i)>error_threshold
                        data_ erroneous_nonerroneous {i} = …
                                        divide_data_in_erroneous_nonerroneous…
                                        (data_of_class_validation(i));
                end if
        end for
        train(classifier_ erroneous_nonerroneous{c}, data_ erroneous_nonerroneous);
        label training(1..c) = test(classifier_erroneous_nonerroneous{1..c}, training data);
        new training data = add(label train, training data);
        label validation (1..c) = test(classifier_erroneous_nonerroneous{1..c}, validation data);
        new validation data = add(label validation, validation data);
        label test (1..c) = test(classifier_erroneous_nonerroneous{1..c}, test data);
        new test data = add(label test, test data);
        new training data, mapping  = LDA(new training data);(optional)
        new validation data  = mapLDA(new validation data, mapping); (optional)
        new test data  = mapLDA (new test data, mapping); (optional)
        train(classifier, new training data, new validation data);
        save_classifiers(c)=classifier;
        out(i)=test(save_classifiers(i), new test data);
end for
ensemble=majority_vote(out(1.. max_iteration));
accuracy=compute_accuracy(ensemble);
return accuracy,save_classifiers, classifier_erroneous_nonerroneous{1..c};
```

Figure 3. The pseudo code of the proposed combinational algorithm

3. Proposed Method

Due to the robustness of the ensemble methods, it has found many usages in different applications. Here we first obtain an ensemble of non-persistent classifiers on training set. Then we combine the outputs those classifiers generate over validation set using simple average method.

Definition: A data point will be defined as an erroneous data point if support difference between the support of its correct class and the one from other possible classes after the correct class is more than a threshold; here we consider this threshold equal to 2%.

This method gets data set as input, and puts it into three partitions: training set, testing set and validation set. Then the data of each class is extracted from the original validation data set. The proposed algorithm assumes that a classifier is first trained on training set, and then this classifier is added to our ensemble. Now using this classifier, we can obtain erroneous data points on validation data set. Using this work we partition validation data points into two classes: erroneous and non-erroneous. At this step, we label validation data points according the two above classes and then using a pairwise classifier we approximate probability of the error occurrence. This pairwise classifier indeed works as an error detector. Next all data, including training, testing and validation are served as input for that classifier, and then their outputs are considered as new features of those data points. At the next step, using linear discriminant analysis (LDA) we reduce the dimensionality of the above new space to that of previous space [26]. We repeat this process in predefined number of iterations. Repeating the above process as many as the predefined number causes to creation of that predefined number of data sets and consequently also that number of classifiers.

Pseudo code of the proposed algorithm is shown in Figure 3. It can be said about time order of this algorithm that the method just multiplies a constant multiplicand in the time order of simple algorithm (training a simple classifier). Suppose that the time order of training a simple classifier on a data set with n data points and c classes to be O(f (n, c)), also assume that in the worst case the time order of training pairwise classifier on that data set to be O(g (n, c)) and also m to be the number of max_iteration (or that predefined number). Then the time order of this method is $\Omega(3 * m * f (n, c))$. Consequently the time order of the method will be $\Omega(m * f (n, c))$. This shows time order of the algorithm relevant to just a constant factor is reduced, that this waste of time is completely tolerable against important achieved accuracy.

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.

4. Experimental Results

The *"Iris"* data set contains 150 samples in 3 classes. Each of classes contains 50 samples. Each class of this data set refers to a type of iris plant. One class is linearly separable from the other two. Each sample has four continuous-valued features. The *"Wine"* data set contains 178 samples in 3 classes. Classes contain 59, 71 and 48 respectively where each class refers to a type of wine. These data are the results of a chemical analysis of wines grown in the same region but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. And finally the *"Bupa"* data set contains 345 samples in 2 classes. Classes contain 145 and 200 respectively. Each data point has six features. In this data set, the first 5 features are all blood tests which are thought to be sensitive to liver disorders that might arise from excessive alcohol consumption.

The predefined number of max_iteration in the algorithm is experimentally considered 3 here. Here, training set, test set and validation set are considered to contain 60%, 15% and 25% of entire dataset respectively. The summery of the results are reported in Table 1. All classifiers used in the ensemble are support vector machines (SVM).

As it is inferred from tables 1, different iterations have resulted in diverse and usually better accuracies than initial classifier. Of course the ensemble of classifiers is not always better than the best classifier over different iterations, but always it is above the average accuracies and more important is the fact that it almost outperforms initial classifier and anytime it is not worse than the first. Indeed the first classifier (classifier in the iteration 1) is simple classifier that we must compare its results to ensemble results. In these tables each row is one independent run of algorithm, and each column of it is the accuracy obtained using that classifier generated in iteration number corresponds to column number. The ensemble column is the ensemble accuracy of those classifiers generated in iteration number 1-3.

"Bupa"	Iteration 1	Iteration 2	Iteration 3	Ensemble
Run 1	0.61765	0.69118	0.48529	0.67647
Run 2	0.67647	0.66176	0.73529	0.67647
Run 3	0.72059	0.75	0.70588	0.75
Run 4	0.66176	0.57353	0.64706	0.66176
Run 5	0.66176	0.66176	0.67647	0.69118
Run 6	0.63235	0.60294	0.66176	0.64706
Run 7	0.66176	0.65686	0.65196	0.68137

Table 1. A summary of seven independent runs of algorithm over "*Bupa*" data sets

As it is inferred from tables 1, different iterations have resulted in diverse and usually better accuracies than initial classifier. Of course the ensemble of classifiers is not always better than the best classifier over different iterations, but always it is above the average accuracies and more important is the fact that it almost outperforms initial classifier and anytime it is not worse than the first. Indeed the first classifier (classifier in the iteration 1) is simple classifier that we must compare its results to ensemble results. In these tables each row is one independent run of algorithm, and each column of it is the accuracy obtained using that classifier generated in iteration number corresponds to column number. The ensemble column is the ensemble accuracy of those classifiers generated in iteration number 1-3.

In the second experimentation, the predefined number of max_iteration in the algorithm is experimentally considered 7 here. Here, training set, test set and validation set are considered to contain 60%, 15% and 25% of entire dataset respectively. The summery of the results are reported in Table 2. All reported results are averaged over 10 distinct runs.

		Dataset Name		
	Base Classifier Type	Iris	Wine	Bupa
Proposed Method	MLP	95.32	98.79	68.48
	KNN	93.25	79.82	63.91
	SVM	95.04	99.01	68.35
	DT	96.12	99.58	70.21
Simple Ensemble	MLP	94.36	98.16	67.68
	KNN	93.28	79.82	63.86
	SVM	94.88	98.73	68.22
	DT	95.14	99.033	69.20

Table 2. Proposed method vs. simple ensemble

As it is obvious from Table 2, recognition ratio is improved considerably when DT is the base classifier rather other base classifiers. Because of low number of features and records in Iris, the improvement is more significant on Wine dataset.

Table 2 shows the results of performance of classification accuracy of the proposed method. These results are average of the ten independent runs of the algorithm. In these results, the parameter K in K-Nearest Neighbor algorithm, KNN, is set to one. The MLPs have two hidden layer with 10 and 5 neurons respectively in each of them.

5. Conclusion

It was shown that the necessary diversity of an ensemble can be achieved by this algorithm. The method was explained in detail above and the result over some real data set proves the correctness of our claim. Although the ensemble created by proposed method may not always outperforms all classifiers existing in all iterations, it is always possesses the diversity needed for creation of ensemble, and consequently it always outperforms the first or the simple classifier. We also showed that time order of this mechanism is not much more than simple methods. Indeed using manipulation of data set features we inject that diversity in the classifiers, it means this method is a type of generative methods that manipulates data set in another way different with previous methods such as bagging and boosting.

References

[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., editors. ((2001). *In*: Proc. 2nd International Workshop on Multiple Classifier Systems (MCS 2001), 2096 of Lecture Notes in Computer Science LNCS Springer- Verlag, Cambridge, UK.

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

[6] Lam, L. (2000). Classifier combinations: implementations and theoretical issues. *In*: Kittler, J., Roli, F. 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. Clustering-based selective neural network ensemble, *Journal of Zhejiang University* Science, Fu et al. / J Zhejiang Univ SCI 2005 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. 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., 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), 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(in press).

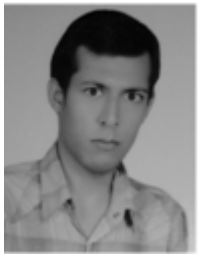
[24] Yang, T. (2006). *International Journal of Computational Cognition* 4 (4) 34–46.

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

[26] Duda, R. O., Hart, P. E., Stork, D. G. (2001). Pattern Classification, 2nd ed. John Wiley & Sons, NY.

[27] Corinna Cortes., Vapnik, V. (1995). Support-Vector Networks, Machine Learning, 20.

Authors Profile



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, artificial intelligence in Iran University of Science and Technology, Iran. His research interests are in pattern recognition, machine learning, particularly, data clustering, classification, and ensemble methods.