An Extension of a Novel Classifier Ensemble Method Based on Class Weightening in Huge Dataset

Hamid Parvin, Zahra Rezaei, Alizadeh Hosein, Sajad Parvin
Nourabad Mamasani Branch
Islamic Azad University
Nourabad Mamasani, Iran.
hamidparvin@mamasaniiau.ac.ir, {rezaei, alizadeh, parvin}@iust.ac.ir

ABSTRACT: *Many methods have been proposed for combining multiple classifiers in pattern recognition such as Random Forest which uses decision trees for problem solving. In this paper, we propose a weighted vote-based classifier ensemble method. The proposed method is similar to Random Forest method in employing many decision trees and neural networks as classifiers. For evaluating the proposed weighting method, both cases of decision tree and neural network classifiers are applied in experimental results. Main presumption of this method is that the reliability of the prediction of each classifier differs among classes. The proposed ensemble method is tested on a huge Persian data set of handwritten digits and shows improvement in comparison with competitors.*

Keywords: Classifier Ensembles, Random Forest, Bagging, Optical Character Recognition, Multiclass Classification

Received: 13 May 2011, Revised 4 July 2011, Accepted 9 July 2011

1. Introduction

Hybridization of intelligent techniques, coming from different computational intelligence areas, has become popular because of the growing awareness that such combinations frequently perform better than the individual techniques coming from computational intelligence.

Practical experience has indicated that hybrid intelligence techniques might be helpful to solve some of the challenging real world problems. In a hybrid intelligence system, a synergistic combination of multiple techniques is used to build an efficient solution to deal with a particular problem. One field of the hybrid intelligent techniques that has recently been hot for researchers is ensemble algorithms.

Ensemble algorithms train multiple base classifiers and then combine their predictions. Generalization ability of an ensemble could be significantly better than a single classifier for difficult problems [8].

In [18] and [15], the relationship between the ensemble and its components, artificial neural networks (ANN), has been analyzed from the context of both regression and classification, which has revealed that it may be better to ensemble many instead of all of the ANNs at hand. They trained a number of ANNs at first. Then random weights were assigned to those networks and a genetic algorithm (GA) was employed to evolve the weights so that they can characterize to some extent the fitness of the ANNs in constituting an ensemble. Finally some ANNs were selected based on the evolved weights to make up the ensemble.

In contrary, assuming that the reliability of the classifiers differs among classes, an approach based on dynamic selection of the classifiers by taking into account their individual votes, has been proposed in [13]. In particular, a subset of the predictions of each classifier is taken into account during weighted majority voting. Others are considered as unreliable and are not used during combination.

In general, an ensemble is built in two steps: (a) generating multiple base classifiers and then (b) combining their predictions. AdaBoost [9] and Bagging [3] are considered as two famous methods in this field.

AdaBoost sequentially generates a series of base classifiers where the training instances are wrongly predicted by so far trained base classifiers will play more important role in the training of its subsequent classifier. Bagging generates many samples (or bags) from the original training set via bootstrap sampling [5] and then trains a base classifier from each of these samples whose predictions are combined via majority voting. A kind of bagging method is Random Forest, where many decision trees (DT) are trained over distinguished perspectives of training dataset [9].

Evolutionary computations are considered universal optimizers or problem solvers. It is common to take it as an optimizer in large fields of science. The most well-known of them is considered to be Genetic Algorithm (GA). John Holland first introduced GA [6]. GA which is considered as one of the optimization paradigms is based on natural processes [3]. A GA can be considered as a composition of three essential elements: first, a set of potential solutions called individuals or chromosomes that will evolve during a number of iterations (generations). This set of solutions is also called population. Second, an evaluation mechanism (fitness function) that allows assessing the quality or fitness of each individual of the population. And third, an evolution procedure that is based on some "genetic" operators such as selection, crossover and mutation. The crossover takes two individuals to produce two new individuals. GA like other machine learning algorithms is based loosely on mechanism of biological evolution. It is applied in the wide problem spaces [11] in two ways: their direct usage as classifiers [8], and their usage as optimizing tools for determining parameters of classifiers. In [2], the GA is used to find decision boundaries in feature space. Another application of the GA is optimization of parameters in classification process. Many researchers also use GA in feature subset selection [1], [4], [10], [14] and [16]. Combination of classifiers is another field that GA has a hand as an optimization tool. Indeed, GA has also been used for feature selection in classifier ensemble [9] and [12]. The quality of the individuals is assessed with a fitness function. The result is a real value for each individual. The best individuals will survive and are allowed to produce new individuals.

A common and obvious way for classifying an instance is from a sequence of questions, so that next question is asked with regard to this current question. Using trees are the most common representation way for these question-answers. DT is used to create a classifier ensemble, expansively. Also, they are used for the application of data mining and clustering. Their functionality is understandable for human. Besides, unlike other methods such as ANN, they are very quick. It means their learning phase is quicker than other methods [6].

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 [17].

Its intrinsic instability can be employed as a source of the diversity which is needed in classifier ensemble. The ensemble of a number of DTs is a well-known algorithm called Random Forest (RF) which is considered as one of the most powerful ensemble algorithms. The algorithm of RF was first developed by Breiman [3].

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.

An ANN has to be configured to be able to produce the desired set of outputs, given an arbitrary set of inputs. Various methods of setting the strength of connections which are considered as ANN learning exist. One way is to set the weights explicitly, using a prior knowledge. Another way which is employed in multi-layer perceptron (MLP) neural network is to 'train' the ANN, feeding it by teaching patterns and then letting it change its weights according to some learning rule [11]. In this paper an MLP neural network is used as classifier.

This paper focuses on Persian handwritten digit recognition (PHDR), especially Hoda dataset [7]. Although there are well works on PHDR, it is not rational to compare them with each other, because there was no standard dataset in the PHDR field until 2006 [7]. The contribution is only compared with those used the same dataset used in this paper, i.e. Hoda dataset.

1.1 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 [3].

1.2 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: $(x,Y) = (x_1, x_2, x_3,\ldots, x_n, Y)$. 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, x_1, x_2, x_3 etc., that are used for that task. To clarify that what the DT learning is, consider table that 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. First, we split our source by the *Refund* and then *MarSt* and finally *TaxInc*. For making rules from a decision tree, we must go upward from leaves as our antecedent to root as our consequent.

2. Proposed Algorithm

Let us to assume that total number of obtained classifiers is denoted by *M*. Let the total number of labels (classes) be denoted by *N*. The solution of the selection problem encoded in the form of a chromosome which has $N \times M$ genes. First *N* genes belong to the first classifier. Second subsequent *N* genes belong to the second classifier. *i*-th *N* genes belong to the *i*-th classifier. The encoding of a chromosome is illustrated in Figure 1, for $N=10$. The genes of each chromosome have real data type. In the exemplary chromosome depicted in Figure 1, the first classifier is not allowed to vote for only fourth, fifth, eighth and tenth classes, and it is allowed to vote for first, second, third, sixth, seventh and ninth classes with coefficients 0.5, 0.6, 0.6, 1, 1 and 1 respectively.

In the proposed algorithm, a multiclass classifier is first trained. Its duty is to obtain confusion matrix over validation set. Note that this classifier is trained over the total train set. At next step, the pair-classes which are mostly confused with each other and are also mostly error-prone are detected. After that, a number of pairwise classifiers are employed to reinforce the drawbacks of the main classifier in those error-prone regions. A set of distinct classifiers is used for each class as an ensemble which is to learn that class. Considering the outputs of the main multiclass classifier and ones of the pairwise classifiers totally as a new space, GA is finally used to determine the weight of each classifier to vote in the ensemble. So, GA is run as many as the number of classes. It means GA is utilized as an aggregator in an ensemble detecting a class. Assume that the number of classes is denoted by *c*. So GA is run *c* times, each of them is denoted by GA_1, GA_2, ..., GA_c. GA_i means the running of GA to detect *i*-th class or equivalently (*i*-1)-th digit.

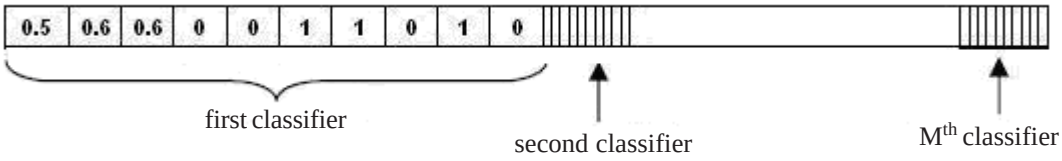

Figure 1. Encoding of a chromosome of used GA, provided the number of classes is N=10 in the problem

Let us denote a chromosome by *b*, an array of $N \times M$ numbers whose elements belong to closed interval [0,1]. In the Figure 1, *b(i)* is the effect weight of *k*-th classifier to vote for selecting *j*-th class, where *k* is calculated according to the equation 1.

$$k = \lceil i / N \rceil \tag{1}$$

and *j* is calculated according to the equation 2.

$$j = i \bmod N \tag{2}$$

Because of non-normalization of the raw *b* chromosome, we first convert it to a normalized version according to the following equation. We denote this normalized version of chromosome *b* by *nb*.

$$nb(b) == \{b \rightarrow nb \mid b \in [0,1]^{N \times M}, nb \in [0,1]^{N \times M}, nb(b)_i = \frac{b_i}{\sum_{q=1}^{M} b_{(q-1)*k+j}} \tag{3}$$

where *k* and *j* are the same in the equation 1 and 2. The *nb* is employed in calculating confidence of classifier ensemble for each class per each test data item *x*. These confidences are obtained according to the following equation.

$$\begin{aligned} &conf(b,x) = = (conf(b,x)_1, conf(b,x)_2, ..., conf(b,x)_N) \mid b \in [0,1]^c, \\ &conf(b,x)_j = \sum_{i=1}^{M} C_{i,j}(x) * nb(b)_{(i-1)*k+j}\} \end{aligned} \tag{4}$$

where *k* and *j* are the same in the equation 1 and 2, *c* is length of chromosome, i.e. $N \times M$, and $C_{i,j}(x)$ is considered as output of *i*-th classifier for *j*-th class for data item *x*.

Now we define the following terms for the following usage. Normalization of an array of numbers is defined as following equation.

$$\begin{aligned} &normalize(a) = \{(normalize(a)_1, normalize(a)_2, ..., normalize(a)_c) \mid a \in [0,1]^c, \\ &normalize(a) = \frac{a_i}{\sum_{j=1}^{c} a_j}\} \end{aligned} \tag{5}$$

Label is a pre-assigned category of data item ***x***. It is denoted by *l*. $l_i(\boldsymbol{x})$ is a number which is considered as membership of ***x*** to

the class *i*. If ***x*** belongs to *i*-th class, $l_i(\boldsymbol{x})$ is 1 and $l_j(\boldsymbol{x})$ is 0 for all $j \neq i$. The fitness of each chromosome (classifier ensemble) is defined as the amount of its accuracy on the evaluation set. The fitness function of a chromosome is computed as equation 6.

$$fitness(b, DV) = \sum_{x \in DV} || normalize(conf(b, x)) - l(x) || \quad (6)$$

where *DV* is validation dataset, *l* is label function. ||.|| is considered as one of norm function like Euclidean distance.

Generate randomly an initial population of size POP_SIZE
For each chromosome in the population
    Compute fitness of the chromosome (as it will be mentioned below)
For Iteration_Num = 1 .. GENERATION_NUM
    For Chromosome_Num = 1 .. POP_SIZE
        1-Select two parents from the old population
        2-Crossover the two parents to produce two offspring with probability P_{cross}
        3-Mutate each bit of each offspring with probability P_{mut}
        4-Apply weighted majority to each of the offspring
        5-Compute fitness of each offspring (as it will be mentioned below)
    End for
    Replace the original population with the offsprings to form the new population
End for
Select the best chromosome as the resultant ensemble

Figure 2. The GA used in the proposed method

In all experiment, GA parameters are kept fixed. In this study, tournament selection is used for the reproduction phase with tournament size of 5. The crossover operator that has an important role in evolutionary computing, allowing them to explore the problem space by sharing different chromosomes information is set to two-point crossover. The mutation operator, allowing evolutionary computing algorithm to exploiting the problem space, is applied to each entry of the offspring chromosomes with a probability $p_{mut} = 0.01$. Probability of selection for crossover operator is $p_{cross} = 0.8$. In the simulation experiments, the population size is selected as 200. It means that 200 different ensemble candidates evolved simultaneously. Pseudo-code of the GA used in the proposed method for evolving the classifier ensembles is shown in Figure 2.

We use two types of classifier in the ensembles: ANN, DT. The first classifier is ANN with *N* outputs which each of the outputs corresponds to a class.

The accuracy of each classifier ensembles is defined as the number of true estimation on the test data set. In order to evaluate the proposed classifier selection approach, we compare it with weighted and unweighted static classifier selection for Hoda data set. Each chromosome is encoded as a string having *M* entries, one for each classifier with data types real and binary respectively in those two methods. In unweighted static classifier selection which has data type binary, if the value of a gene is 1; this means that the classifier is selected for being used in the corresponding ensemble. It is worthy to note that all the design parameters of the above-mentioned algorithm including population size, number of iterations, crossover and mutation rate etc. are kept fixed. The Figure 3 illustrates the training phase of the proposed method generally.

The testing phase of the weighted classifier ensemble is depicted in Figure 4. F_i is *i*-th classifier and $f_i(x|l_j)$ is confidence of *i*-th classifier for *j*-th class. w_k is the same as nb_k.

As it is inferred from Figure 4, the final decision for an input example, ***x***, is the class the label with maximum weighted summation of classifiers output.

3. Experimental Study

Hoda data set [7] is a handwritten OCR data set. This data set contains 100000 data points. Some data instances are depicted in

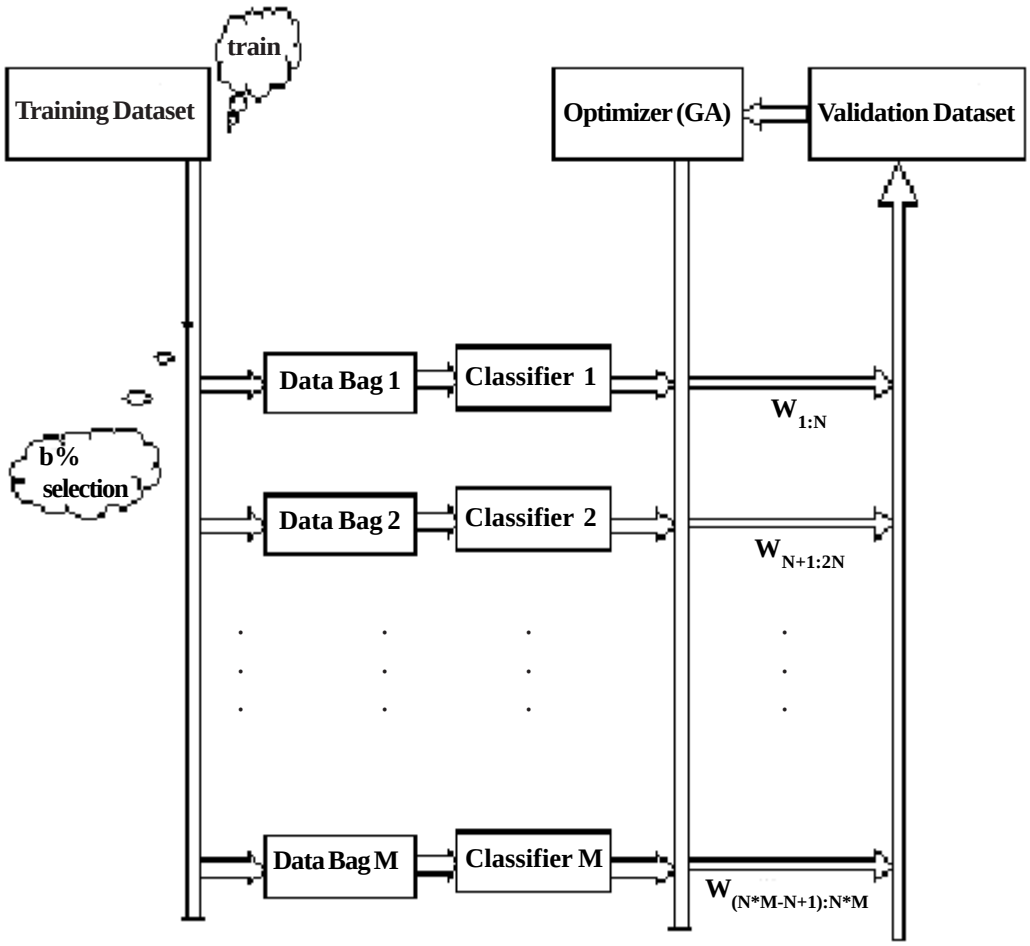

Figure 3. Scheme of the training phase of the weighted classifier ensemble

the Figure 4. HODA dataset is the first dataset of handwritten Farsi digits that has been developed during an MSc. project in Tarbiat Modarres University entitled: Recognizing Farsi Digits and Characters in SANJESH Registration Forms. This project has been carried out in cooperation with Hoda System Corporation. It was finished in summer 2005 under supervision of Prof. Ehsanollah Kabir. Samples of the dataset are handwritten characters extracted from about 12000 registration forms of university entrance examination in Iran. The reader is referred to [7] for more details about the data set such as the process of feature generation.

3.1 Parameter Setting

In this paper, MLP and DT are used as base primary classifier. We use an MLPs with 2 hidden layers including respectively 10 and 5 neurons in the hidden layer 1 and 2, as the base Multiclass classifier. Confusion matrix is obtained from its output. Also DT's measure of decision is taken as Gini measure. The classifiers' parameters are kept fixed during all of their experiments. It is important to take a note that all classifiers in the algorithm are kept unchanged. It means that all classifiers are considered as MLP in the first experiments. After that the same experiments are taken by substituting all MLPs whit DTs.

3.2 Experimental Results

The first row in the Table 1 stands for the accuracy of ensemble of all 201 classifiers without classifier weighting or selection. The majority vote is employed for making final decision in the unweighted full ensemble.

The method in rows 2 in Table 1, unweighted static classifier selection, only focuses on the selected classifiers which are allowed to vote unweightedly based on majority vote mechanism. Indeed this method uses a binary chromosome with the length size which is equaled to the number of classifiers. *i*-th bit of the chromosome stands for being-absent/participating *i*-th classifier. For

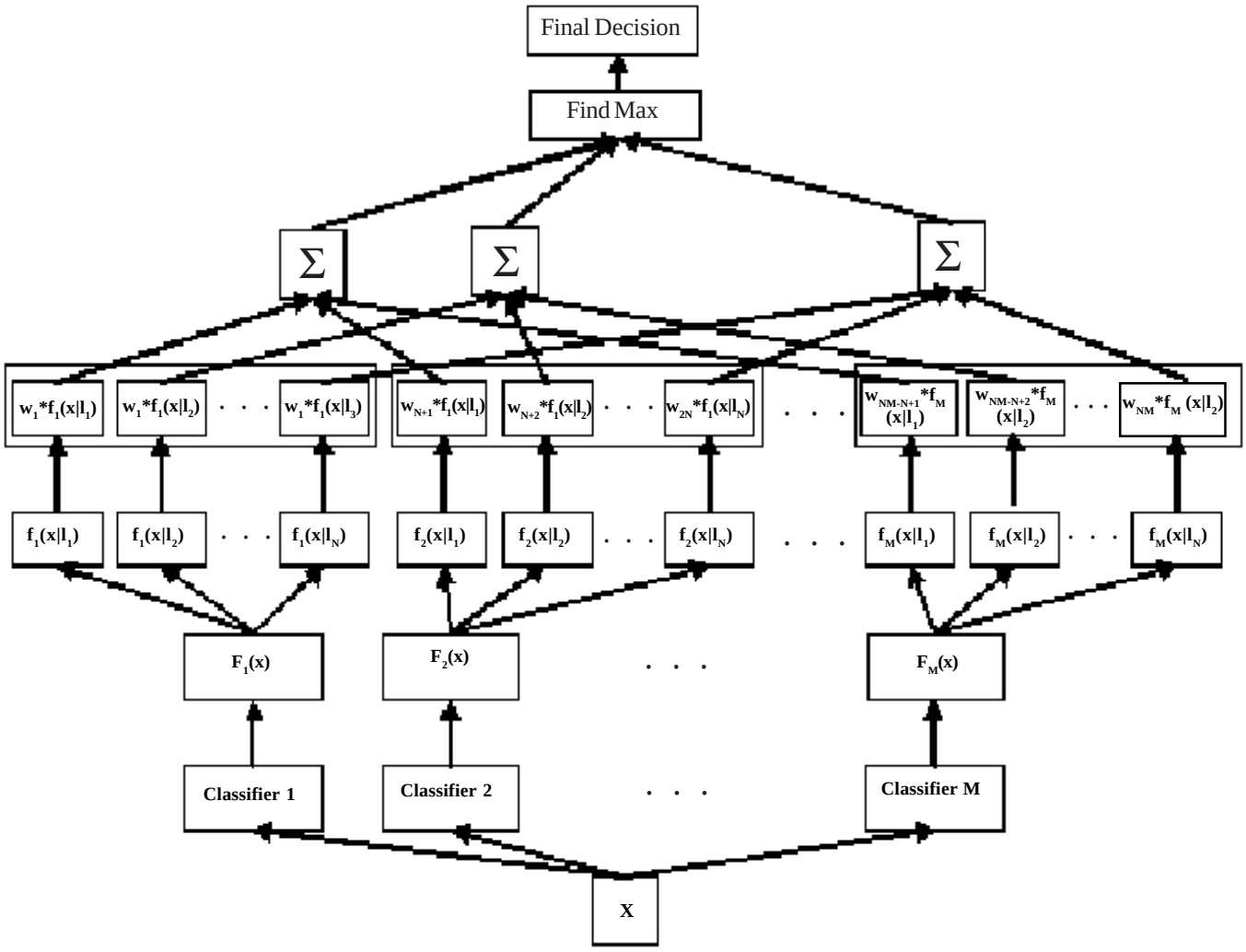

Figure 4. Scheme of the testing phase of the weighted classifier ensemble

example, value 1 for *i*-th bit of the chromosome means that the classifier number *i* must participate in the ensemble, else it must be absent in the ensemble and it is prevented for voting.

The method in rows 3 in Table 1, weighted static classifier selection, uses a real chromosome again with the length size which is equaled to the number of classifiers. *i*-th bit of the chromosome stands for amount or weight of being-absent/participating *i*-th classifier.

Because of unbalanced accuracy of classifiers in the ensemble, generally, the static classifier selection can give better results than the simple full ensemble. Usually the weighted approaches are doing better than unweighted ones. However, the results of weighted and unweighted approaches are close to each other, the weighted method slightly outperforms the unweighted one. It improves the result achieved by the full ensemble. Even though, a classifier is not able to achieve good accuracy in all classes; it may obtain a good accuracy on one special class. So, the method which is introduced in [13] has a good result.

It is also worthy to mention that using the absolute error as fitness function rather than equation 6 of the neural networks available in the ensemble, results in less improvement as in the row 4.

Another aspect of the proposed approach is that its computational cost is very low. Although we can train just one MLP to reach to a good accuracy, it consumes many days for large data sets like Hoda. We need to train an MLP for some weeks to reach the accuracy approximately 98% on Hoda data set. The weak learners can converge to a good accuracy very soon, but the

Classification Scheme	Accuracy with ANN base classifier	Accuracy with DT base classifier
Unweighted Full Ensemble	98.11	98.22
Unweighted Static Classifier Selection	98.15	98.13
Weighted Static Classifier Selection	98.21	98.34
The Method in [19]	98.27	98.41
The Method in [13] using Soft Error	98.51	98.45
The proposed Method	98.99	99.03

Table 1. The results of proposed ensemble method

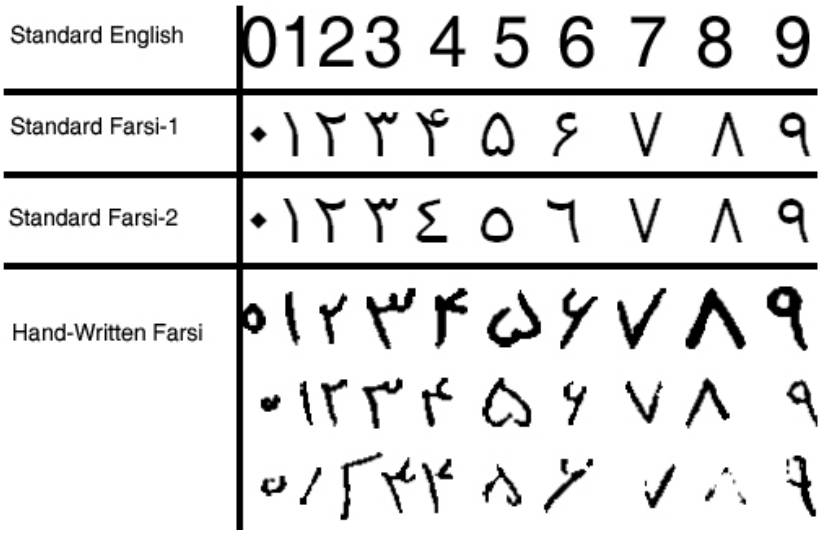

Figure 4. Some instances of Farsi OCR data set, with different qualities

subsequent small improvements are very slow. In this approach, we have some weak base classifiers that are under-trained but ensemble of them is not under-trained. We provided 201 individual weak base MLPs or DTs as members in the ensemble. Also it is notable that both ensembles of MLPs and DTs are comparable, with fairly superior of DTs ensemble.

It is inferred from Table 1 that the proposed framework causes a significant improvement in the classification precision specially when employing DT as base classifier. Taking a look at Table 1 shows that using DT as base classifier in ensemble almost always produces a better performing classification. It may be due to inherent instability of DT. It means that because a DT is unstable classifier, so it is better to use it as a base classifier in an ensemble. A stable classifier is the one converge to an identical classifier apart from its training initialization. It means the 2 consecutive trainings of the classifier with identical initializations, results in two classifiers with the same performance. This is not valid for the DT classifier. Although MLP is not a stable classifier, it is more stable than DT. So it is also expected that using DT classifier as base classifier has the most impact in improving the recognition ratio.

As another point to be mentioned, reader can infer that using the framework can outperforms Unweighted Full Ensemble, Unweighted Static Classifier Selection and Unweighted Static Classifier Selection methods explained in [9]. This can be in consequence of employing binary classifiers instead of multiclass classifiers.

It is inferred from the Table 1 that the proposed framework affects significantly in improving the classification precision specially when employing DT as base classifier. It is also obvious that using DT classifier as base classifier has more impact in improving the recognition ratio. It is may be due to its inherent instability.

4. Conclusion

Because of their robustness and high performance, classifier ensemble methods are used for difficult problem solving. In this paper, a new ensemble algorithm is proposed, which is designed for building ensembles of bagging classifiers.

The proposed method is a weighted vote-based classifier ensemble like Random Forest method which employs DT and ANN as classifiers.The empirical study on the very large dataset of Persian handwritten digits, Hoda shows that the proposed approach is superior to other combination methods of classifiers, as it is discussed. It effectively improves the accuracy of full ensemble of ANN or DT classifiers. It has been shown that using DT classifier as base classifier has more impact in improving the recognition ratio than using ANN classifier as base classifier. It is may be due to its inherent instability.

References

[1] Bala, J., De Jong, K., Huang, J., Vafaie, H., Wechsler H. (1997). Using learning to facilitate the evolution of features for recognizing visual concepts , *Evolutionary Computation*, Vol 4, No 3.

[2] Bandyopadhyay, S., Muthy, C.A. (1995). Pattern Classification Using Genetic Algorithms, *Pattern Recognition Letters*, 16, p. 801-808.

[3] Breiman, L. (1996). Bagging Predictors, *Journal of Machine Learning*, 24 (2) 123-140.

[4] Guerra-Salcedo, C., Whitley, D. (1999). Feature Selection mechanisms for ensemble creation: a genetic search perspective, AAAI Workshop, p. 13-17.

[5] Efron, B., Tibshirani, R. (1993). An Introduction to the Bootstrap, New York: Chapman & Hall.

[6] Holland, J. (1992). Adaptive in Natural and Artificial Systems, MIT Press, USA.

[7] Khosravi, H., Kabir, E. (2007). Introducing a very large dataset of handwritten Farsi digits and a study on the variety of handwriting styles, *Pattern Recognition Letters*, 28 (10) 1133-1141.

[8] Kuncheva, L. I., Jain, L. C. (2000). Designing Classifier Fusion Systems by Genetic Algorithms, *IEEE Transaction on Evolutionary Computation*, 33. 351-373.

[9] Kuncheva, L. I. (2005). Combining Pattern Classifiers, Methods and Algorithms, Wiley, USA.

[10] Martin-Bautista, M. J., Vila M. A. (1999). A survey of genetic feature selection in mining issues, Congress on Evolutionary Computation (CEC-99), p. 1314-1321.

[11] Minaei-Bidgoli, B., Punch, W. F. (2003). Using Genetic Algorithms for Data Mining Optimization in an Educational Web-based System, GECCO.

[12] Minaei-Bidgoli, B., Kortemeyer, G., Punch, W. F. (2004). Mining Feature Importance: Applying Evolutionary Algorithms within a Web-Based Educational System, *In*: Int. Conf. on Cybernetics and Information Technologies, Systems and Applications, p. 381-386.

[13] Dimililer, N., Varoglu, E., Altýncay, H. (2007). Vote-Based Classifier Selection for Biomedical NER Using Genetic Algorithms, Iberian Conference on Pattern Recognition and Image Analysis, p. 202-209.

[14] Punch, W. F., Pei, M., Chia-Shun, L., Goodman, E. D., Hovland, P., Enbody, R. (1993). Further research on Feature Selection and Classification Using Genetic Algorithms, International Conference on Genetic Algorithm, p 557-564.

[15] Zhou, Z. H., Wu, J. X., Tang, W.(2002). Ensembling Neural Networks: Many Could Be Better Than All, Artificial Intelligence, p. 239-263.

[16] Vafaie, H., De Jong, K. (1993). Robust feature Selection algorithms, Int. Conf on Tools with AI, p. 356-363.

[17] Yang, T. (2006). Computational Verb Decision Trees, *International Journal of Computational Cognition*, p. 34–46.

[18] Zhou, Z. H., Wu, J. X., Jiang, Y. , Chen, S. F. (2001). Genetic Algorithm based Selective Neural Network Ensemble, International Joint Conference on Artificial Intelligence, p. 797-802.

[19] Parvin, H., Alizadeh, H., Minaei-Bidgoli, B. (2008). A New Approach to Improve the Vote-Based Classifier Selection, International Conference on Networked Computing and advanced Information Management, p. 91-95.