A New Method for Constructing Classifier Ensembles

Zahra Rezaei and Sajad Parvin

Department of Computer Engineering, Nourabad Mamasani Branch, Islamic Azad University, Nourabad Mamasani, Iran

Received: 2 Jan. 2014, Revised: 7 Oct. 2014, Accepted: 10 Oct. 2014
Published online: 1 Jan. 2015

Abstract: Usage of recognition systems has found many applications in almost all fields. 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 Ensemble, Combination of Multiple Classifiers, Support Vector Machine.

1 Introduction

Usage of recognition systems has found many applications in almost all fields. However, Most of classification algorithms have obtained good performance for specific problems; they have not enough robustness for other problems. Therefore, recent researches are directed to the combinational methods which have more power, robustness, resistance, accuracy and generality. Combination of multiple Classifiers (aMC) can be considered as a general solution method for pattern recognition problems. Inputs of CMC are results or separate classifiers and output of CMC is near combined decisions according to [1] and [2].

These methods train multiple base classifiers and then combine their predictions. Since the generalization ability of an ensemble could be significantly better than a single classifier, combinational methods have been a hot topic during the past years [2], [3]. It was established firmly as a practical and effective solution for difficult problems [4]. It appeared under numerous names: hybrid methods, decision combination, multiale experts, mixture of experts, classifier ensemble, cooperative agents, opinion pool, decision forest, classifier fusion, combinational systems and so on. Combinational methods usually result in the improvement of classification, because classifiers with different features and methodologies can complete each other [4]-[6]. Kuncheva in [7] using Condorcet Jury theorem [8], has shown that combination of classifiers can usually operate better than single classifier provided that its components are independent. It means if more diverse classifiers are used in the ensemble, then error of them can considerably be reduced. In general, theoretical and empirical works showed that a good ensemble is one where the individual classifiers have both accuracy and diversity. In other words, the kndividual classifiers make their errors on difference parts of the input space [9], [10]. Many approaches have been proposed to construct such ensembles. One group of these methods obtains diverse individuals by training accurate classifiers on different training set, such as bagging, boosting, cross validation and using artificial training examples [10]-[13]. Another group of these methods adopts different topologies, initial weight setting, parameter setting and training algorithm to obtain individuals. For example, Rosen sn [14] adjusted the training algorithm of the network by introducing a penalty term to encourage individual networks to be decorrelated. For more convergence on ensemble method readers are referred to [7] and [15].

In section 2 we will briefly overview combining classifier levels. We will try in section 3 to obtain really independent and diverse classifier using manipulation of data set. And finally in section 4 we will conclude.
2 combining Classifiers

In general, creation of combinational classifier may be in four steps [7]. It means combining of classifiers may happen in four levels. Figure 1 depicts these four steps. In step four, we try to create different subsets of data in order to make independent classifiers. Bagging and boosting are examples of this method [11], [16]. Some other methods create independent classifiers trained on manipulated data by relabeling data [17]. In these examples, we use different subset of data instead of all data for training. In step three, we use subset of features for obtaining diversity in ensemble. In this method, each classifier is trained on different subset of features [15], [18]-[19]. In step two, we can use different kind of classifiers for creating the ensemble [15]. Finally, in the step one, method of combining (fusion) is considered.

In the combining of classifiers, we intend to increase the performance of classification. There are several ways for combining classifiers. The simplest way is to find best classifier. Then we use it as map classifier. This method is offline CMC. Another method that is named onlin CMC is all classifier in ensemble. For examiln, this work is done using voting. We also use arg voting majority method in this paper.

3 Proposed Method

3.1 Background

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 all the outputs of classifiers to generate a single valid prediction 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, end 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 error data points on validation 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 to the above two classes and then using a pairwise classifier we approximate probability of error occurrence. This pairwise classifier indeed works on error detectrs. 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 this step, we label validation data points according to the two above classes and then using a pairwise classifier we approximate probability of the error occurrence. This pairwise classifier indeed works on error detectrs. 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.

Pseudo code of the proposed algorithm is shown in Figure 2.

It can be said about time order of this algorithm that the method just multiplies a constant multiplier 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 (f(n,c)), also assume taw in the worst case the Ome order of training pairwise classifier on that data set to be O(g(n,m)) and also m to be the number of max iteration (or that predefined number). Then the time order of this method is Ω(3*m*f(n,c)). Consequently the time order of the method will be Ω(m*f(n,c)). This shoes the time order of the algorithm relevant to just a constant factor is reduced, that is the waste of time is completely tolerable against important achieved accuracy.

4 Experimental Results

The experiments were performed on three data sets: “iris”, “Wine” and “Bupa”. A summary of these data set characteristics is depicted in Table 1. Here, the training set, test set and validation set contain 60%, 15% and 25% of the entire data set respectively. Proposed Algorithm (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=compute_error_on_each_class(classifier, validation set);
  nor i=1 to number_of_classes
    if error(i)>error_threshold
      data_of_class_validation(i)=extract_data_of_each_class(validation data);
    end if
  end for
  train(classifier, training data, validation set);
  error=compute_error_on_each_class(classifier, validation set);
  nor i=1 to number_of_classes
    if error(i)>error_threshold
      data_of_class_validation(i)=extract_data_of_each_class(validation data);
    end if
  end for
end for
Figure 2. The pseudocode of the proposed combinational algorithm

<table>
<thead>
<tr>
<th>No. of Classes</th>
<th>No. of Features</th>
<th>No. of Patterns</th>
<th>Patterns per class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wine</td>
<td>3</td>
<td>13</td>
<td>178</td>
</tr>
<tr>
<td>Bupa</td>
<td>2</td>
<td>6</td>
<td>345</td>
</tr>
<tr>
<td>Iris</td>
<td>3</td>
<td>4</td>
<td>150</td>
</tr>
</tbody>
</table>

Table 1. A summary of our data sets characteristics

4.1 Data Sets

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.

Strategy pattern: When making the cache server capable of choosing either caching method, we used strategy pattern to separate the basic algorithm for replacement of objects and implemented it nicely so that does not depend on the other parts of the program and could be extended easily.

4.2 Results

The predefined number of max_iteration in the algorithm is experimentally assumed to be 3 here. All classifiers used in the ensemble are support vector machines (SVM). Here, the training set, test set and validation set are coide to contain 60%, 15% and 25% of entire data set respectively. The results are reported in Table 2-4.

As it is inferred from Tables 2 to 4, 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 accuracy and more important is the fact that it almost outperforms initial classifier and anytime it is not worse than the finest. Indeed the first classifier (classifier in the iteration 1) is simple classifier that was just compared 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 corresponding to column number. The
Table 2. A summary of seven independent runs of algorithm over "Iris" data sets

<table>
<thead>
<tr>
<th>&quot;Iris&quot;</th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>Iteration 3</th>
<th>Ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run 1</td>
<td>0.93333</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Run 2</td>
<td>0.9</td>
<td>0.9</td>
<td>0.96667</td>
<td>0.9</td>
</tr>
<tr>
<td>Run 3</td>
<td>0.96667</td>
<td>0.96667</td>
<td>0.8</td>
<td>0.96667</td>
</tr>
<tr>
<td>Run 4</td>
<td>0.93333</td>
<td>0.93333</td>
<td>0.96667</td>
<td>0.96667</td>
</tr>
<tr>
<td>Run 5</td>
<td>0.96667</td>
<td>0.96667</td>
<td>0.8</td>
<td>0.96667</td>
</tr>
<tr>
<td>Run 6</td>
<td>0.9</td>
<td>0.93333</td>
<td>0.26667</td>
<td>0.93333</td>
</tr>
<tr>
<td>Run 7</td>
<td>0.9222</td>
<td>0.93333</td>
<td>0.7222</td>
<td>0.95</td>
</tr>
</tbody>
</table>

Table 3. A summary of seven independent runs of algorithm over "Wine" data sets

<table>
<thead>
<tr>
<th>&quot;Wine&quot;</th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>Iteration 3</th>
<th>Ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run 1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Run 2</td>
<td>0.97222</td>
<td>0.97222</td>
<td>0.97222</td>
<td>0.97222</td>
</tr>
<tr>
<td>Run 3</td>
<td>0.94444</td>
<td>0.94444</td>
<td>0.94444</td>
<td>0.94444</td>
</tr>
<tr>
<td>Run 4</td>
<td>0.91418</td>
<td>0.91418</td>
<td>0.91418</td>
<td>0.91418</td>
</tr>
<tr>
<td>Run 5</td>
<td>0.98148</td>
<td>0.98148</td>
<td>0.98148</td>
<td>0.98148</td>
</tr>
<tr>
<td>Run 6</td>
<td>0.94444</td>
<td>0.94444</td>
<td>0.94444</td>
<td>0.94444</td>
</tr>
<tr>
<td>Run 7</td>
<td>0.98148</td>
<td>0.98148</td>
<td>0.98148</td>
<td>0.98148</td>
</tr>
</tbody>
</table>

Table 4. A summary of seven independent runs of algorithm over "Bupa" data sets

<table>
<thead>
<tr>
<th>&quot;Bupa&quot;</th>
<th>Iteration 1</th>
<th>Iteration 2</th>
<th>Iteration 3</th>
<th>Ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run 1</td>
<td>0.61765</td>
<td>0.69118</td>
<td>0.48529</td>
<td>0.67647</td>
</tr>
<tr>
<td>Run 2</td>
<td>0.67647</td>
<td>0.66176</td>
<td>0.73529</td>
<td>0.67647</td>
</tr>
<tr>
<td>Run 3</td>
<td>0.72059</td>
<td>0.75</td>
<td>0.70588</td>
<td>0.75</td>
</tr>
<tr>
<td>Run 4</td>
<td>0.66176</td>
<td>0.57353</td>
<td>0.64706</td>
<td>0.66176</td>
</tr>
<tr>
<td>Run 5</td>
<td>0.66176</td>
<td>0.66176</td>
<td>0.67647</td>
<td>0.69118</td>
</tr>
<tr>
<td>Run 6</td>
<td>0.63235</td>
<td>0.60294</td>
<td>0.66176</td>
<td>0.64706</td>
</tr>
<tr>
<td>Run 7</td>
<td>0.66176</td>
<td>0.65686</td>
<td>0.65196</td>
<td>0.68137</td>
</tr>
</tbody>
</table>

5 Conclusion and Discussion

It was shown that the necessary diversity of an ensemble can be achieved by this algorithm. The method was explained on the example above and the result over some real data sets proves the correctness of our claim. Although the ensemble created by the algorithm may not always outperform all classifiers existing in all iterations, it always possesses the diversity needed for creation of ensemble, and consequently it always outperforms the first or the simplest 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 a different way different with previous methods such as bagging and boosting.

6 Acknowledgement

This work is partially supported by Data and Text Mining Research Group Computer Research Center of Islamic Sciences (CRCIS), NOOR co. P.O. Box 37185-3857, Qom, Iran.

References