Pattern Recognition 44 (2011) 97–106

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Sparse ensembles using weighted combination methods based on linear programming

Li Zhang *, Wei-Da Zhou

Institute of Intelligent Information Processing, Xidian University, Xi'an 710071, China

ARTICLE INFO

Article history:
Received 11 March 2010
Received in revised form
2 July 2010
Accepted 18 July 2010

Keywords:
Classifier ensemble
Linear weighted combination
Linear programming
Sparse ensembles
k nearest neighbor

ABSTRACT

An ensemble of multiple classifiers is widely considered to be an effective technique for improving accuracy and stability of a single classifier. This paper proposes a framework of sparse ensembles and deals with new linear weighted combination methods for sparse ensembles. Sparse ensemble is to sparsely combine the outputs of multiple classifiers by using a sparse weight vector. When the continuous outputs of multiple classifiers are provided in our methods, the problem of solving sparse weight vector can be formulated as linear programming problems in which the hinge loss or/and the 1-norm regularization are exploited. Both the hinge loss and the 1-norm regularization are techniques inducing sparsity used in machine learning. We only ensemble classifiers with nonzero weight coefficients. In these LP-based methods, the ensemble training error is minimized while the weight vector of ensemble learning is controlled, which can be thought as implementing the structure risk minimization rule and naturally explains good performance of these methods. The promising experimental results over UCI data sets and the radar high-resolution range profile data are presented.

1. Introduction

Recently, combining multiple classifiers has been a very active research technique. It is widely accepted that combining multiple classifiers can achieve better classification performance than a single (best) classifier, supported by experimental results [1–3]. An ensemble means combining multiple versions of a single classifier or multiple various classifiers. One classifier used in an ensemble is called an individual or component classifier. There are two important issues in combining multiple classifiers. One is that an ensemble of classifiers must be both diverse and accurate in order to get better performance. Diversity can ensure that all the individual classifiers make uncorrelated errors. If classifiers get the same errors which will be propagated to the ensemble, no improvement can be achieved in combining multiple classifiers. In ensemble learning, there are two schemes to implement diversity [4]. One scheme is to seek diversity explicitly (i.e., to define a diversity measure and optimize it), and the other is to seek diversity implicitly. Here we consider the scheme of seeking diversity implicitly. One common way is to train individual classifiers by using different (randomly selected) training sets [5–7]. Bagging [5] and Boosting [6] are well known examples of successfully iterative methods for reducing a generalization error. The other way is to train multiple classifiers by using different feature sets [8,9]. In addition, accuracy of individual classifiers is also important, since too many poor classifiers can suppress correct predictions of good classifiers.

The other issue is about combination rules or fusion rules, which is regarding how to combine the outputs of individual classifiers. So far, many combination rules have been proposed [10–16]. If the labels are available, a simple (majority) voting (SV) rule can be used [10]. If the continuous outputs like posteriori probabilities are supplied, an average, linear or nonlinear combination rules can be employed [10,12,16]. Linear weighted voting is the most frequently used rule [11,12,15]. Work on weighted voting have addressed the problem of weights estimation, in a regression setting [11,14,15], or in a classification setting [12,17,18]. A linear weighted voting based on the minimum classification error (WV-MCE) criterion is presented in [12], which is solved by using gradient descent methods. In [17], a genetic algorithm (GA) is used to select the best subset of classifiers and the corresponding weight coefficients in neural network ensembles. Grove et al. [18] suggest that we should make the minimum margin of learned ensembles as large as possible by minimizing training set error. They propose the LP-Adaboost method to find the sparse weight vector.

The LP-Adaboost method in [18] and the GA-based method in [17] are the beginning of sparse ensembles. By sparse ensembles, we mean combining the outputs of all classifiers by a sparse weight vector. Each classifier model has its own weight value, zero or nonzero. Only classifiers corresponding to nonzero coefficients play a role in the ensemble. As it is known, a sparse

* Corresponding author.
E-mail address: lizhang.ml@gmail.com (L. Zhang).

0031-3203/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2010.07.021

model representation in machine learning is expected to improve the generalization performance and computational efficiency [19–21]. The mechanism to maximize the sparseness of a model representation can be thought of as an approximative form of the minimizing description length principle which can be used to improve the generalization performance [7]. The sparsity in machine learning can be measured by the number of nonzero coefficients in a decision function.

The above combination rules except LP-Adaboost and GA-based methods are to try to combine all classifiers in an ensemble. In general classifier ensembles, it is necessary to combine all individual classifiers to ensure good performance. It results in a large memory requirement and a slow classification speed [22]. Selective ensembles, also called pruned ensembles, are designed to remedy the drawbacks of general classifier ensembles. Only a fraction of individual classifiers is selected and combined by using simple or weighted voting in selective ensembles. In [22], some methods are introduced for selecting a subset of individual classifiers, and the performance of these methods are compared in several benchmark classification tasks. The problem of selecting the optimal subset of classifiers is a combinatorial search assuming that the generalization performance can be estimated in terms of some quantity measured on the training set [22]. Recently, global optimization methods, e.g., GA [23] and semi-definite programming [24] are used to solve the combinatorial search problem. Since the global methods cost a lot, some suboptimal ensemble pruning methods based on ordered aggregation are proposed, including reduce-error pruning [25], margin distance minimization (MDM) [26], orientation ordering [27], boosting-based ordering [28], expectation propagation [29], and so on. Among the pruning techniques, MDM and boosting-based ordering methods provide similar or better classification performance [22]. Actually the concept of pruned ensembles is identical with that of sparse ensembles. In pruned ensemble, the coefficients of selected classifiers are nonzero, and unselected are zero, which generates a sparse weight vector. Generally, pruned ensembles use simple voting or weighted voting. The nonzero coefficients take the value one in simple voting [22], and a value proportional to the classification accuracy of the corresponding classifier [30,31], or found by some optimization methods [23,24,29] in weighted voting.

This paper gives a framework of sparse ensemble learning, and proposes new weighted combination methods for sparse ensembles. The key problem in sparse ensembles is to find a sparse weight vector. Grove and Schuurmans use a linear programming method to find a sparse weight vector. The objective function of LP-Adaboost is to minimize maximum margin in [18]. Here, our goal is to find a sparse weight vector by minimizing the ensemble training error and simultaneously controlling the weight vector of ensemble learning, which can be taken as implementing the structural risk minimization rule from the view of machine learning. In our methods, the continuous outputs (estimated posteriori probabilities or discriminant function values) of individual classifier are required. This learning problem can also be formulated as linear programming problems in which sparseness techniques the hinge loss or/and the 1-norm regularization are used. In our experiments, we consider the k NN classifier as an individual classifier and apply the new linear weighted combination rule to combine the multiple k NN classifiers.

The rest of this paper is organized as follows. In Section 2, we propose the framework of sparse ensembles and review the related work including some classical combination rules. Section 3 presents new linear weighted voting based on LP. We compare our methods with the single k NN classifier and the k NN ensemble classifiers based on other seven combination rules on the UCI data sets and the radar high-resolution range profile (HRRP) data in Section 4. Section 5 concludes this paper.

2. Sparse ensembles and other related work

In this section, we firstly propose the framework of classifier sparse ensembles and then introduce some other combination methods used in our experiments.

2.1. Framework of sparse ensembles

Sparse ensembles mean that we combine the outputs of all classifiers using a sparse weight vector. Each classifier model has its own weight value, zero or nonzero. Only classifiers corresponding to nonzero coefficients play a role in the ensemble. To reduce memory demand and improve test speed, it is required to select an optimal sub-ensemble (or a subset of classifiers) in pruned (or selective) ensembles [22,30–32]. Actually the concept of pruned ensembles is identical with that of sparse ensembles. In pruned ensemble, the coefficients of selected classifiers are nonzero, and unselected are zero, which creates a sparse weight vector.

Now consider a multi-class classification problem. Let a training sample set be $X=\{(\mathbf{x}_i,y_i)|\mathbf{x}_i\in\mathbb{R}^D,\ y_i\in\ \{1,2,\ldots,c\},\ i=1,2,\ldots,\ell\}$, where y_i are labels of $\mathbf{x}_i$, D is the dimensionality of the sample space (or the number of sample features), c is the number of classes, and ℓ is the total number of training samples. Hereafter we use ω_m to denote class m, $m=1,\ldots,c$. If $\mathbf{x}_i\in\omega_m$, then $y_i=m$. The framework of sparse ensembles is shown in Fig. 1. The whole ensemble process is divided into two phases: training phase and test phase. In training phase, $X_1,X_2,\ldots,X_N$ are the training sets of N individual classifiers, respectively. In this phase, we need to find the sparse weight vector $\boldsymbol{\alpha}=[\alpha_1,\alpha_2,\ldots,\alpha_N]^T\in\mathbb{R}^N$ by using some methods, such as LP-Adaboost. In the test phase, the goal is to estimate the label of a given test sample $\mathbf{x}$. Assume the j-th classifier would generate an output vector $\mathbf{f}_j=[f_{j1}(\mathbf{x}),f_{j2}(\mathbf{x}),\ldots,f_{jc}(\mathbf{x})]^T\in\mathbb{R}^c$, where $f_{jm}(\mathbf{x})$ are the output of the j-th classifier for the sample $\mathbf{x}$ associated with class ω_m, which could be posteriori probabilities or just only discriminant values normalized to the interval [0,1]. The ensemble output of $\mathbf{x}$ for class ω_m is

$$f_m^*=\sum_{j=1}^{N}\alpha_j f_{jm}(\mathbf{x}) \tag{1}$$

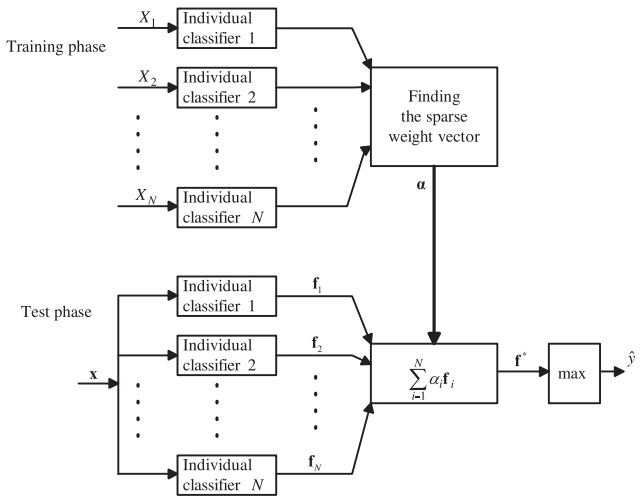

Fig. 1. Framework of classifier ensembles.