Information Sciences 278 (2014) 545–554

Contents lists available at ScienceDirect

Information Sciences

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

A comparative study of suboptimal branch and bound algorithms

Songyot Nakariyakul *

Department of Electrical and Computer Engineering, Thammasat University, 99 Moo 18 Phaholyothin Rd., Amphur Klongluang, Pathumthani 12120, Thailand

ARTICLE INFO

Article history:
Received 24 July 2009
Received in revised form 5 April 2011
Accepted 8 March 2014
Available online 22 March 2014

Keywords:
Branch and bound algorithm
Dimensionality reduction
Feature selection
High dimensionality
Look-ahead search strategy
Suboptimal solution

ABSTRACT

The branch and bound algorithm is widely known as an efficient approach for selecting optimal feature subsets. If the optimality of the solution is allowed to be compromised, it is possible to further improve the search speed of the branch and bound algorithm. This paper studies the look-ahead search strategy which can eliminate many solutions deemed to be suboptimal early in the branch and bound search. We propose ways to incorporate the look-ahead search scheme into four major branch and bound algorithms, namely the basic branch and bound algorithm, the ordered branch and bound algorithm, the fast branch and bound algorithm, and the adaptive branch and bound algorithm. A comparative study of the look-ahead scheme in terms of the computational cost and the solution quality on these suboptimal branch and bound algorithms is carried out on real data sets. Furthermore, we test the feasible use of suboptimal branch and bound algorithms on a high-dimensional data set and compare its performance with other well-known suboptimal feature selection algorithms.

1. Introduction

Feature selection [4,8,9,11] is to select a representative subset of features from an original feature set. It is an essential pre-processing step in many pattern recognition applications because not only does it reduce the cost of collecting irrelevant and redundant features, but it also improves the prediction performance of the predictors. For high-dimensional data sets that contain more than hundreds of features [22,27], use of feature selection is mandatory. Thus, feature subset selection is an important subject in machine learning problems, and many feature selection algorithms have been proposed in the literature.

To compare the performance of different feature selection algorithms, a criterion function J is used to assess the quality of the selected feature subset. Feature selection techniques in the literature can thus be categorized into two types: suboptimal and optimal methods. Suboptimal feature selection techniques select a subset with a good (not necessary the best) J value, whereas optimal search algorithms find the subset that yields the best (optimal) J value. Suboptimal methods include sequential search [13,17,21] and randomized search [10,20,23] algorithms. These methods are relatively fast and suitable for very high-dimensional problems. Optimal feature selection algorithms include exhaustive search and the branch and bound (BB) algorithm [18] (and its variants). Exhaustive search is a brute-force approach that evaluates a given criterion function J for all possible combinations of subsets with desired dimensions and chooses the subset with the best J value.

* Tel.: +66 2564 3001 9x3148; fax: +66 2564 3001 9x3071.
E-mail address: nsongyot@engr.tu.ac.th

http://dx.doi.org/10.1016/j.ins.2014.03.072
0020-0255/

Exhaustive search is only practical for low-dimensional problems. The BB algorithm, on the other hand, explores the search space more efficiently than an exhaustive search. It organizes the search to reject many subsets that are guaranteed to be suboptimal without computing their J values and thus runs drastically faster than an exhaustive search. The BB algorithm has been successfully employed in many tasks such as minimizing the total completion time in a job scheduling problem [19], achieving an optimal facility layout in a manufacturing environment [24], and optimizing a reliable optical network design [2].

Although many advanced versions [6,16,25,28] of the BB algorithm have been proposed to improve the search speed of the original BB algorithm, the computational time of the BB algorithm is still very excessive when the dimensionality of the original feature space reaches a few dozens or more. However, if the bound of the BB algorithm is relaxed and the optimality of the solution is allowed to be compromised, the search speed can be greatly accelerated [18], which makes the BB algorithm feasible for high-dimensional problems. This paper studies the look-ahead search strategy which is used to eliminate solutions deemed to be suboptimal early in the branch and bound search.

This paper has two main objectives. The first objective is to investigate the performance of the BB algorithm when the optimality of the algorithm is not retained. We study a tradeoff between the solutions of selected feature subsets and the reduced execution time. We analyze the suboptimal solutions of four major BB algorithms in the literature, namely the basic BB algorithm [18], the ordered BB algorithm [6], the fast BB algorithm [25], and the adaptive BB algorithm [16]. We summarize the advantages and disadvantages of each version of the BB algorithm using experimental results for real data sets. The second objective of this paper is to explore the possibility of using the suboptimal BB algorithm on high-dimensional data, for which use of the optimal BB algorithm is prohibitive. We compare the performance of a suboptimal BB algorithm with other well-known suboptimal feature selection algorithms on a data set with 60 features.

This paper is organized into five sections. Section 2 presents an overview of the basic BB algorithm and prior improvements to it. We review the look-ahead search scheme to incorporate the branch and bound algorithm for suboptimal solutions in Section 3. Experimental results for real data sets are given in Section 4, and conclusions are advanced in Section 5.

2. The branch and bound algorithm and its improvements

In this section, the BB algorithm and prior improvements to it [6,16,18,25,28] are discussed. Without loss of generality, we assume that a subset with a larger value of criterion function J is better than one with a smaller J value. The original BB algorithm was first proposed by Narendra and Fukunaga in 1977 [18] and is referred to as the basic BB algorithm in this paper. The BB algorithm requires that the criterion function J satisfy the monotonicity condition, i.e., adding a new feature to a feature subset does not decrease the criterion function J value. That is, assume that $\{y_1, y_2, \ldots, y_i\}$ is a subset of $\{y_1, y_2, \ldots, y_j\}$ for $i < j$, the monotonicity property of the criterion function requires that the J values of the two sets, $Y\backslash\{y_1, y_2, \ldots, y_i\}$ and $Y\backslash\{y_1, y_2, \ldots, y_j\}$, fulfill

$$J(Y \setminus \{y_1, y_2, \ldots, y_i\}) \geqslant J(Y \setminus \{y_1, y_2, \ldots, y_j\}), \tag{1}$$

where $Y\backslash\{y_1, y_2, \ldots, y_j\}$ denotes removing the subset of j features $y_1, y_2, \ldots, y_j$ from the feature set Y.

To find the optimal subset of d features from an original set of D features, the basic BB algorithm selects $D - d$ features to be *discarded* by constructing a solution tree. An example of the solution tree corresponding to selecting the best $d = 2$ features from $D = 5$ total features is shown in Fig. 1. The numbers in parenthesis next to each node in Fig. 1 refer to the set of features remaining at that node. The root of the tree (the top) corresponds to the set of all five features, and the leaves at the bottom of the tree correspond to all ten possible subsets of two features. The number on a branch refers to the number of the feature removed as the search traverses that branch. The level number denotes the total number of features that have been omitted from the root at that level. The problem is to find the best node (leaf) at the bottom of the tree (level $D - d$) with the largest J value by exploring the tree with the smallest number of J calculations.

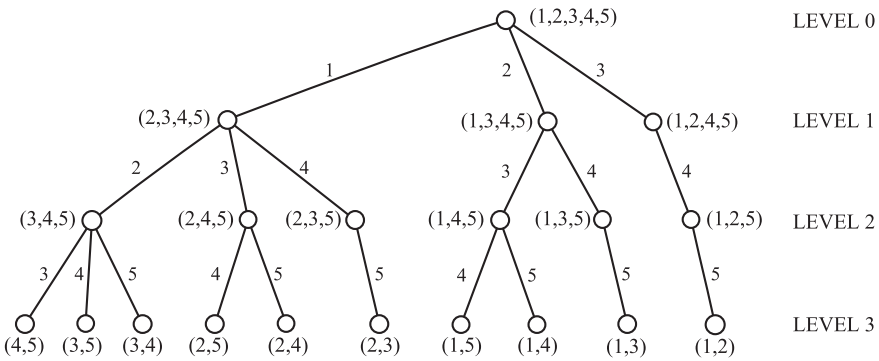

Fig. 1. The solution tree for the basic BB algorithm when $d = 2$ and $D = 5$.