Decision Support Systems 61 (2014) 79–92

Contents lists available at ScienceDirect

Decision Support Systems

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

Towards high dimensional instance selection: An evolutionary approach

Chih-Fong Tsai *, Zong-Yao Chen

Department of Information Management, National Central University, Taiwan

ARTICLE INFO

Article history:
Received 28 March 2013
Received in revised form 23 December 2013
Accepted 28 January 2014
Available online 5 February 2014

Keywords:
Data reduction
Instance selection
Data mining
Machine learning
Genetic algorithms
High dimensional data

ABSTRACT

Data reduction is an important data pre-processing step in the KDD process. It can be approached by the application of some instance selection algorithms to filter out unrepresentative or noisy data from a given (training) dataset. However, the performance of instance selection over very high dimensional data has not yet been fully examined. In this paper, we introduce a novel efficient genetic algorithm (EGA), which fits "biological evolution" into the evolutionary process. In other words, after long-term evolution, individuals find the most efficient way to allocate resources and evolve. The experimental study is based on four very high dimensional datasets ranging from 200 to 18,236 dimensions. In addition, four state-of-the-art algorithms including IB3, DROP3, ICF, and GA are compared with EGA. The experimental results show that EGA allows the *k*-NN and SVM classifiers to provide the most comparable classification performance with the baseline classifiers without instance selection. Particularly, EGA outperforms the four algorithms in terms of average classification accuracy. Moreover, EGA can produce the largest reduction rates (the same as GA) and it requires relatively less computational time than the other four algorithms.

1. Introduction

Nowadays, the datasets collected for some specific domains tend to be very large (containing a huge amount of data) and complex (containing a very large number of variables or features), which leads to the so-called 'big data' problem [17]. Consequently, data reduction, as a data pre-processing task has became one of the most important steps in data mining or knowledge discovery in databases (KDD). Data reduction is aimed at shrinking the size of the collected dataset to facilitate the later mining analysis step, i.e., model development.

In other words, if the chosen dataset contains too many instances (i.e., data samples) it can result in large memory requirements, slow execution speed, and over-sensitivity to noise. In addition, one problem with using the original data points is that there may not be any located at the precise points that would make for the most accurate and concise concept description [27].

To this end, instance selection can be utilized to filter out noisy (or unrepresentative) data, which are likely to degrade the data mining performance, from a given dataset [20,30]. The size of the dataset is reduced when some noisy data are removed by a specific instance selection algorithm. After this, data mining algorithms can be applied to the reduced dataset, to achieve sufficient results if the selection strategy is appropriate [28].

Let us now discuss instance selection. Given a dataset D composed of training set T and testing set U, let $S \subset T$ be the subset of selected instances resulting from the execution of an instance selection algorithm. Then, U is used to test a classification technique trained by S [4,6].

A better algorithm can select better quality data from T, which in turn would make the classifier trained by the reduced dataset S perform better than one trained by T alone, or other reduced datasets containing lower quality data produced by other algorithms. The number of selected instances in the reduced dataset (selected by a better (or effective) algorithm) would not necessarily be smaller than the ones selected by other algorithms. In other words, algorithms that produce larger reduction rates are not necessarily effective because over selection may occur, which would filter out many representative instances of T. Reduced datasets containing many lower quality data can directly affect the final classification accuracy.

There are a number of related studies proposing instance selection methods for obtaining better mining quality in the literature. Specifically, Pradhan and Wu [26] and Jankowski and Grochowski [16] surveyed several relevant selection techniques, which can be divided into three application-type groups: noise filters, condensation algorithms, and prototype searching algorithms. In addition, Wilson and Martinez [30] and Brighton and Mellish [3] conducted some comparative experiments. They found the Iterative Case Filtering (ICF) and Decremental Reduction Optimization Procedure 3 (DROP3) to be cutting-edge instance selection algorithms, which make the *k*-NN classifier provide better performances over other instance selection methods.

The genetic algorithms (GA), one of the most widely used techniques for instance selection, have also been used to improve the performance of data mining algorithms [6]. In particular, Cano et al. [4]

* Corresponding author. Tel.: +886 3 422 7151; fax: +886 3 4254604.
E-mail address: cftsai@mgt.ncu.edu.tw (C.-F. Tsai).

0167-9236/$ – see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.dss.2014.01.012

showed that better results can be obtained with a GA technique than many traditional and non-evolutionary instance selection methods in terms of better data reduction rates and higher classification accuracy. Similarly, Nanni and Lumini [23] demonstrated the superiority of GA over other instance selection methods, such as fuzzy clustering and particle swarm optimization (PSO).

Recently, García et al. [12] conducted an extensive study of comparing fifty related instance selection algorithms including the above mentioned algorithms over 58 different datasets. In these works, however, the performance of the instance selection algorithms was assessed using datasets from the UCI Machine Learning Repository[1], where the dimensionalities are very low, less than 100. However, many real world problems contain very high dimensional data, composed of several hundred to several thousands of features. For example, the dataset for text classification will usually contain at least several thousand to over ten thousand representative terms as features, while image classification is based on several hundred low-level features, such as color, texture, and/or shape features. A biological dataset can contain thousands of genes coded for proteins and their locations in various parts of the cells, and so on.

In other words, the major limitation of the afore-mentioned studies is the lack of high dimensional data reduction. The mining performance when performing instance selection over high dimensional datasets has not been fully examined. In this study, four very high dimensional datasets are used, with data dimensionalities of over 200.

GAs basically search for optimal solutions using generation succession. However, reproduction mechanisms, crossover, and mutation calculation can cause optimal chromosomes to disappear over successive generations, thereby making it impossible to fully use previous search experience. Meanwhile, a lack of diversity in chromosome populations produces premature convergence, which limits the search for a local optimum. In addition, to reach the optimal solution, an exhaustive search over the entire solution space must be carried out, which, for many complex problems, is computationally intractable.

Therefore, in this paper, a novel instance selection method for high dimensional data reduction, which we call the efficient genetic algorithm (EGA), is introduced. This method is designed to minimize the computational burden and improve the optimal solution, i.e., the instance selection result. EGA simulates the biological evolutionary process and natural rules where, after long-term evolution, individuals find the most efficient way to allocate resources and evolve [2]. Inspired by nature, EGA is constructed as an efficient and effective problem solving method for instance selection.

The major contribution of this study is to introduce a novel evolutionary-based instance selection algorithm, EGA. It is an extension of the genetic algorithm based on biological evolution. Moreover, the results of EGA assessment over high dimensional datasets are compared with those from four well-known and representative instance selection algorithms. The experimental results demonstrate that on average EGA outperforms the chosen baseline instance selection methods, allowing it to provide the highest classification accuracy, the least storage requirement, and the lowest computational complexity.

The rest of this paper is organized as follows. Section 2 describes the concept of GA and its application to instance selection. Section 3 introduces the proposed EGA method for instance selection. In Section 4, the experimental results, based on high dimensional datasets containing various domain problems, are presented. Finally, conclusions are given in Section 5.

2. Literature review

The main idea of the evolutionary algorithm (EA) is derived from Darwin's theory of evolution or natural selection, of which the genetic algorithm (GA) is one example [9,13]. The basic idea of a GA is that you have a population of strings (called chromosomes), which encode candidate solutions (called individuals) in an optimization problem. In general, the genetic information (i.e., chromosome) is represented by a bit string (such as binary strings of 0 s and 1 s) and sets of bits encode the solution. Genetic operators are then applied to the individuals in the population for the next generation (i.e., a new population of individuals). There are two main genetic operators: crossover and mutation. Crossover creates two strings of offspring from two parent strings by copying selected bits from each parent, whereas mutation randomly changes the value of a single bit (with small probability). In addition, a fitness function is used to measure the quality of an individual in order to increase the probability that the single bit can survive throughout the evolutionary process.

Basically, using GA for instance selection contains the following steps [15]. First, as the initialization step, a number of individuals are randomly generated, and the length of each individual is m, which is the total number of the training set. Second, as the genetic operation step, it follows the process of the original genetic algorithm, which includes 'selection' (i.e. to randomly select a pair of strings from the current population), 'mating' (i.e. to select a pair of strings to generate two offspring) and 'mutation' (i.e. to randomly change the bit value from 1 to 0 or from 0 to 1). Finally, as the termination step, in this step, if a pre-specified stopping condition is not satisfied, the instance selection process returns to the second step; otherwise the best string with the largest fitness value is produced, where the reduced set is determined. An example of performing GA for instance selection is shown in Fig. 5.

3. The efficient genetic algorithm

3.1. The basic concept

While GAs have demonstrated success for a diverse set of problems, they are only able to handle simple concepts. Basically, the idea is that if resources are limited, the "individuals" will follow the most reasonable and simplest rules — allowing for a more effective use of resources, or "reproduction of species" [2,24]. By using simple rules, individuals that maximize the "savings cost" will become more efficient. While reasonable rules help with this approach, if we can fit the concept of "biological evolution" into the evolutionary process, where the most streamlined process also complies with reasonable rules, we can not only closely simulate the natural evolution of an algorithm, but also the algorithm will be both efficient and effective.

In other words, the algorithm only pursues the simplest evolutionary process. While in general this is reasonable, as it is able to solve problems fairly efficiently, a number of other elements are discarded in the pursuit of efficiency in the evolutionary process. This could result in a degradation of performance, and could also cause it to fall into the local optimal solution. To avoid this problem, we introduce a novel algorithm, called the efficient genetic algorithm (EGA).

In EGA, we use the individual/organism to represent a solution. In particular, the individual is similar to a gene in GA or a particle in particle swarm optimization (PSO). In addition, the '*Kings*' represent the best group, containing the top K individuals from each iteration. After the evaluation, EGA will randomly select/choose some individuals for the mating pool. These individuals are able to perform crossover and mutation processes and thus create new individuals. Specifically, each iteration of EGA will generate a new generation, which is composed of many individuals including the *Kings* and other individuals. Migration is a mechanism which can help EGA jump out from the local optima, just like the migration of organisms.

A flow chart of the EGA process is shown in Fig. 1, in which the white boxes represent the classical GA components and processes, and the gray boxes indicate the additional processes and components of EGA.

[1] http://archive.ics.uci.edu/ml/.