Computers & Operations Research 37 (2010) 1395 – 1405

Contents lists available at ScienceDirect

Computers & Operations Research

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

Genetic programming for anticancer therapeutic response prediction using the NCI-60 dataset

Francesco Archetti[a,b], Ilaria Giordani[a], Leonardo Vanneschi[a,*]

[a]Dipartimento di Informatica, Sistemistica e Comunicazione (D.I.S.Co.), University of Milano-Bicocca, 20126 Milan, Italy
[b]Consorzio MilanoRicerche, 20126 Milan, Italy

ARTICLE INFO

Available online 5 March 2009

Keywords:
Genetic programming
Machine learning
Regression
Microarray data
Anticancer therapy
NCI-60

ABSTRACT

Statistical methods, and in particular machine learning, have been increasingly used in the drug development workflow. Among the existing machine learning methods, we have been specifically concerned with genetic programming. We present a genetic programming-based framework for predicting anticancer therapeutic response. We use the NCI-60 microarray dataset and we look for a relationship between gene expressions and responses to oncology drugs Fluorouracil, Fludarabine, Floxuridine and Cytarabine. We aim at identifying, from genomic measurements of biopsies, the likelihood to develop drug resistance. Experimental results, and their comparison with the ones obtained by Linear Regression and Least Square Regression, hint that genetic programming is a promising technique for this kind of application. Moreover, genetic programming output may potentially highlight some relations between genes which could support the identification of biological meaningful pathways. The structures that appear more frequently in the "best" solutions found by genetic programming are presented.

1. Introduction

We investigate the usefulness of genetic programming (GP) [23,43] for understanding the functional relationship between gene expressions[1] and therapeutic response to four clinical agents: Fluorouracil (5-FU), Floxuridine, Fludarabine and Cytarabine. We use the NCI-60 microarray dataset [8,11,38], a panel of 60 cell lines derived from several different cancer types, including leukemias, melanomas, ovarian, renal, prostate, colon, lung and CNS cancers.

GP is an evolutionary approach which extends the genetic model of learning to the space of programs. It is a major variation of genetic algorithms (GAs) [18,15] in which the evolving individuals are themselves computer programs instead of fixed length strings from a limited alphabet of symbols. In the last few years, GP has become popular for biomedical applications. In particular, GP has been recently used to mine large datasets with the goal of correlating the behavior of latent features with some interesting parameters bound to drug activity patterns. For instance, in [44] GP has been used to classify drug-like molecules in terms of their bioavailability. In [1] it has been used for quantitative prediction of drug induced toxicity. In [48] GP has been applied to cancer expression profiling data to select feature genes and build molecular classifiers by mathematical integration of these genes. In [32] the usefulness of GP to attribute selection and classification in human genetics has been discussed. GP can be regarded as an optimization method, which makes no assumption on the objective functions and the data. Furthermore, as pointed out in [1], GP often automatically performs a feature selection, proposing solutions that use subsets of data. Thus, the motivation behind our choice of using GP is twofold:

- Biological data, like gene expression levels, are not independent of each other. Rather, small subsets of genes and molecules work in cohesion [6]. These phenomena lead to high multidependency among the features. Hence, the underlying algorithm should be capable of extracting features from high-dimensional correlated data.
- The dimensionality of the feature space in biomedical datasets is normally much higher than the number of observations available for training. Hence, automatic feature selection as well as other methods to handle overfitting and minimizing the generalization error should be encouraged.

* Corresponding author.
E-mail address: vanneschi@disco.unimib.it (L. Vanneschi).

[1] In our study, experiments have been conducted on cell lines, i.e. external to people. The translation from results in laboratories to people is non-trivial and our work has not accomplished this. For this reason, here we have used the term "gene expressions" and not the term "patients' gene expressions". However, in the continuation of this paper we will sometimes improperly use the term "patients" for simplicity.

0305-0548/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2009.02.015

GP results are compared with the ones returned by linear regression and least square regression.

Section 2 introduces modeling microarray data to therapeutic response, discussing previous and related work. Section 3 presents GP and its use in this work (a more detailed discussion of GP functioning can be found in Appendix A). In Section 4 the machine learning methods used for comparison with GP are discussed. In Section 5 we describe the method employed to build the dataset used in our experiments. In Section 6 we discuss experimental results. Finally, Section 7 concludes the paper and offers hints for future research.

2. State of the art

2.1. DNA microarrays

DNA microarrays have dramatically accelerated many types of investigations in many fields of medicine, bioinformatics and systems biology [29,2]. The advantages in microarrays technology and the growing availability of biological measurements performed at molecular level have intensified the role of machine learning methods for effective cancer prediction and classification. These measurements are represented by the expression levels of thousands of genes exhibited in different kind of tissues under the same experimental conditions. The collection of gene expression data usually results in high-dimensional datasets, composed of a huge number of features (genes) and a relative few number of tissues. The use of a collection of distinct DNAs in arrays for expression profiling was first described in [24]. The use of miniaturized microarrays for gene expression profiling was first reported in [40] and a complete eukaryotic genome (*Saccharomyces cerevisiae*) on a microarray was published in [27]. In the last decade, many contributions have appeared using DNA-microarrays for studying the molecular mechanism underlying several cancer types. In [8], for instance, the classification of a set of cell lines used in the National Cancer Institute (NCI)'s screen for anticancer drugs revealed a correspondence to the ostensible origins of the tumors from which the cell lines themselves were derived. Comparison of gene expression patterns in the cell lines to those observed in normal tissues or in tumor specimens revealed features of the expression patterns in tumors that had recognizable counterparts in specific cell lines. In the same year, Scherf and coworkers [11] used cDNA microarrays to assess gene expression profiles in 60 human cancer cell lines used in a drug discovery screen by the NCI. Using these data, they linked bioinformatics and chemoinformatics by correlating gene expression and drug activity patterns. They used the NCI-60 [38] dataset, which provides a large amount of data, and thus an excellent opportunity for modeling pharmacogenomics. This dataset has been used also in this work, as described in Section 6.

2.2. Machine learning for DNA microarrays

Modeling the relationship between genomic features and therapeutic responses is of central interest in pharmacogenomics [33,12]. To identify genes that might be associated with chemosensitivity, a cDNA microarray representing 23,040 genes to analyze expression profiles in a set of 85 cancer tissues derived from nine human organs was used in [9]. In [10], authors used cDNA microarrays to compare gene expression profiles of colorectal biopsies from 25 sick tissues and 13 healthy ones. In [7] tumor colon samples from 21 patients with advanced colorectal cancer were analyzed for gene expression profiling. A predictor classifier was constructed using support vector machines. An attempt to identify the genetic components contributing to drug sensitiveness using the NCI-60 dataset has been done in [6], using partial least squares. Feature selection methods have been applied with success to genomic microarray data in [47,16,42,45] and evolutionary algorithms have recently been employed in [20,28]. A novel rank correlation-based multiobjective evolutionary biclustering method to extract simple gene interaction networks from microarray data is proposed in [5]. A survey about the role of different soft computing paradigms, like fuzzy sets, artificial neural networks, evolutionary computation, rough sets, and support vector machines in bioinformatics can be found in [31].

3. Genetic programming

GP [23,43] is an evolutionary approach which extends GAs [18,15] to the space of programs. Like any other evolutionary algorithm, GP works by defining a goal in the form of a quality criterion (or *fitness*) and then using this criterion to *evolve* a set (also called population) of solution candidates (also called individuals) by mimic the basic principles of Darwin evolution theory [4]. The most common version of GP, and also the one used here, considers individuals as LISP-like tree structures that can be built recursively from a set of function symbols $\mathscr{F}=\{f_1,f_2,\ldots,f_n\}$ (used to label internal tree nodes) and a set of terminal symbols $\mathscr{T}=\{t_1,t_2,\ldots,t_m\}$ (used to label tree leaves). GP breeds these solutions to solve problems by executing an iterative process involving the probabilistic selection of the fittest solutions and their variation by means of a set of genetic operators, usually crossover and mutation. A detailed introduction to GP and its functioning and motivations can be found in Appendix A.

We have used a tree-based GP configuration for regression problems inspired by Koza [23], Keijzer [21,22]. Each feature in the dataset has been represented as a floating point number. Potential solutions (GP individuals) have been built by means of the set of functions $\mathscr{F}=\{+,*,-,/,\log,\sin,\cos,\exp,sqrt\}$.[2] The same technique as in [21] has been used to avoid individuals containing a division with the denominator equal to zero. The set of terminals $\mathscr{T}$ was composed of $M-1$ floating point variables (where M is the number of columns in the dataset). The fitness function we have used is the root mean squared error (*RMSE*) on the training set.

Example. Suppose the names of the floating point variables contained into the $\mathscr{T}$ set are $x_1,x_2,\ldots,x_{M-1}$. Then, a candidate solution found by GP, expressed in infix notation, may for instance be

$$F(x_1,x_2,\ldots,x_{M-1})=\exp(x_5)+x_{12}-x_{34}$$

A tree representation of this expression is given in Fig. 1. Its fitness is obtained by calculating the value of $F(x_1,x_2,\ldots,x_{M-1})$ on all the

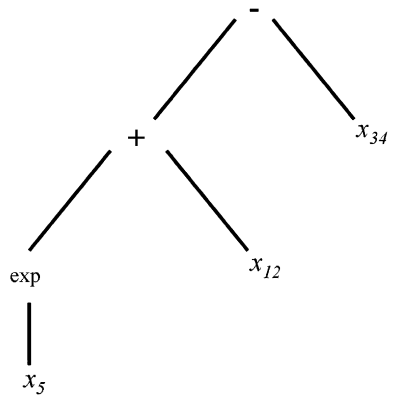

Fig. 1. An example of a simple GP individual.

[2] This set of functional symbols have been chosen after a large amount of simulations and are the ones that have allowed GP to find the most performing solutions.