Information Sciences 369 (2016) 347–367

Contents lists available at ScienceDirect

Information Sciences

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

Evolving genetic programming classifiers with novelty search

Enrique Naredo[a], Leonardo Trujillo[a,*], Pierrick Legrand[b], Sara Silva[c], Luis Muñoz[a]

[a] Tree-Lab, Doctorado en Ciencias de la Ingeniería, Departamento de Ingeniería Eléctrica y Electrónica, Instituto Tecnológico de Tijuana, Blvd. Industrial y Av. ITR Tijuana S/N, Mesa Otay C.P. 22500, Tijuana BC, México
[b] Université de Bordeaux, INRIA Bordeaux Sud-Ouest, CQFD Team, IMB, UMR CNRS 5251, France
[c] BioISI – Biosystems & Integrative Sciences Institute, Faculty of Sciences, University of Lisbon, Portugal

A R T I C L E I N F O

Article history:
Received 23 November 2015
Revised 22 June 2016
Accepted 25 June 2016
Available online 30 June 2016

Keywords:
Novelty search
Behavior-based Search
Supervised classification
Bloat

A B S T R A C T

Novelty Search (NS) is a unique approach towards search and optimization, where an explicit objective function is replaced by a measure of solution novelty. However, NS has been mostly used in evolutionary robotics while its usefulness in classic machine learning problems has not been explored. This work presents a NS-based genetic programming (GP) algorithm for supervised classification. Results show that NS can solve real-world classification tasks, the algorithm is validated on real-world benchmarks for binary and multiclass problems. These results are made possible by using a domain-specific behavior descriptor. Moreover, two new versions of the NS algorithm are proposed, Probabilistic NS (PNS) and a variant of Minimal Criteria NS (MCNS). The former models the behavior of each solution as a random vector and eliminates all of the original NS parameters while reducing the computational overhead of the NS algorithm. The latter uses a standard objective function to constrain and bias the search towards high performance solutions. The paper also discusses the effects of NS on GP search dynamics and code growth. Results show that NS can be used as a realistic alternative for supervised classification, and specifically for binary problems the NS algorithm exhibits an implicit bloat control ability.

1. Introduction

Evolutionary algorithms (EAs) are a broad family of search and optimization algorithms that are based on a simplified model of Neo-Darwinian evolution [9], achieving impressive results in many domains [15]. The bio-inspired origins of EAs suggest a substantial difference with respect to traditional optimization approaches. However, EAs are guided by an objective function and specially designed search operators just like most optimization algorithms [25]. The use of an objective function in standard EAs is a key difference with respect to natural evolution, which is an open-ended process that lacks a predefined purpose.

There is, however, another approach to build EAs, what is normally referred to as open-ended artificial evolution. Open-ended algorithms do not use an objective function to drive the search, at least not an explicit one. An important feature of

* Corresponding author.
E-mail addresses: enriquenaredo@tree-lab.org (E. Naredo), leonardo.trujillo@tectijuana.edu.mx (L. Trujillo), pierrick.legrand@u-bordeaux.fr (P. Legrand), sara@fc.ul.pt (S. Silva), lmunoz@tectijuana.edu.mx (L. Muñoz).
URL: http://www.tree-lab.org (L. Trujillo)

http://dx.doi.org/10.1016/j.ins.2016.06.044

open-ended systems is the continuous emergence of novelty [1]. In fact, some of the earliest EAs were open-ended [6], but they have mostly been used in specialized domains such as artificial life [35] and interactive search [14]. Only recently has open-ended search been proposed to solve mainstream problems, one promising algorithm is Novelty Search (NS) proposed by Lehman and Stanley [19]. NS was conceived to overcome deception in evolutionary robotics (ER) [19,20,22], a common issue in most challenging problems [47].

Lehman and Stanley relate deception with problem hardness, stating that “[a] deceptive problem is one in which a reasonable EA will not reach the desired objective in a reasonable amount of time” [22] (p.193). The core idea behind NS is that using an objective function to determine fitness in challenging problems may mislead the search and prevent it from reaching a global optimum. Therefore, the proposal of NS is to abandon the objective function as the source of selective pressure, and instead determine selective pressure based on the novelty or “uniqueness” of each individual by considering a description of the behavior each individual exhibits. From the NS perspective, a behavior refers to a description of the interaction between a candidate solution and its domain-specific context.

NS has achieved promising results in different areas of ER [48], such as navigation [11,19–22,44], morphology design [23] and gait control [22]. Despite the growing evidence that NS can be used as an alternative to traditional objective-based search (OS), we conjecture that it is not yet widely used for the following reasons. First, most work on NS has been limited to ER, providing little insight regarding the competence of NS in other areas, particularly in common machine learning problems. Second, NS introduces several additional algorithm parameters that must be heuristically tuned. Third, NS relies on a kernel method to estimate the uniqueness of each new solution based on its dissimilarity with previously generated solutions. Such an approach leads to a high computational overhead, which is normally solved with additional heuristics. Finally, NS has been shown to struggle when behavior space is large [13], the search for specific behaviors in these cases can become very slow while the algorithm explores many uninteresting solutions. To address this problem, Lehman and Stanley proposed an extension to NS called Minimal Criteria NS (MCNS) [21], where a solution is considered to be novel only if it is unique and satisfies some domain-specific minimal criteria, thus reducing the portion of behavior space that is explored.

The present work builds on previous contributions to extend the NS paradigm. Firstly, we apply NS on supervised classification with genetic programming (GP) and propose a behavior descriptor for evolved GP classifiers, whereas previous works on NS have focused mainly on ER. The NS approach is tested on twelve real-world datasets, considering binary and multi-class problems and using two different GP-based classifiers. Secondly, an extension to the basic NS algorithm is proposed, where the novelty of a solution is estimated probabilistically by modelling each behavior as a random vector. The proposed strategy is called probabilistic novelty search (PNS), which reduces the computational cost of the original NS algorithm, and all the parameters introduced by NS are eliminated. Thirdly, several NS variants are extensively tested and compared, including NS, MCNS and PNS. Results show that NS-based GP can perform competitively relative to a standard OS, while endowing the search with implicit bloat control in some cases. Preliminary results of this research were presented in [26,31,32,41,43]; however, those works only studied the general applicability of the original NS algorithm on synthetic pattern recognition problems without considering any algorithmic improvements or real-world scenarios. Nonetheless, those works served as a proof-of-concept for the proposed approach, which is fully explored and evaluated in the current paper. In summary, the work presented here will help establish NS as a viable alternative for GP-based machine learning.

The remainder of this paper is organized as follows. Section 2 provides the required background for this work, an overview of GP is given and the concept of behaviors in GP is introduced, discussing how it relates to objective-based fitness and semantics as understood within GP literature. Section 3 describes the NS algorithm and the proposed MCNS variant. Section 4 presents our basic approach towards applying NS with a GP-based classifier. Afterwards, the proposed PNS is described in Section 5. The experimental setup and results are presented in Section 6. Finally, Section 7 contains a summary, conclusions and future work.

2. Background

This section introduces GP, analyzes the search spaces used by GP, introduces the concept of behavior in GP, and discusses how it can be related with an open-ended search algorithm such as NS.

2.1. GP search

One of the central challenges of computer science was stated in the 1950s by Arthur Lee Samuel [16]; “how can computers be made to do what needs to be done, without being told exactly how to do it?”. The evolutionary computation paradigm called GP is a generalization of genetic algorithms (GA) that provides a noteworthy proposal to address this challenge. It is able to create computer programs from a high-level problem statement, a process which is also called program synthesis or automatic program induction [16].

GP is a domain-independent evolutionary method intended to solve problems without requiring the user to know or specify the form or structure of the solution in advance. GP is inspired in natural genetic operations, applying similar operations to computer programs, namely crossover (sexual recombination) and mutation. Computer programs can be represented in several ways, but syntax trees are the traditional representation used in GP literature [16]. Other representations include linear GP [2] and Cartesian GP [36].