Applied Soft Computing 21 (2014) 382–406

Contents lists available at ScienceDirect

Applied Soft Computing

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

Differential Evolution algorithms applied to Neural Network training suffer from stagnation

Adam P. Piotrowski*

Institute of Geophysics, Polish Academy of Sciences, Ks. Janusza 64, 01-452 Warsaw, Poland

ARTICLE INFO

Article history:
Received 29 November 2012
Received in revised form 17 March 2014
Accepted 19 March 2014
Available online 12 April 2014

Keywords:
Differential Evolution
Artificial Neural Networks
Stagnation
Global and Local optimization
Benchmark problems
Algorithm population size

ABSTRACT

Large number of population-based Differential Evolution algorithms has been proposed in the literature. Their good performance is often reported for benchmark problems. However, when applied to Neural Networks training for regression, these methods usually perform poorer than classical Levenberg–Marquardt algorithm. The major aim of the present paper is to clarify, why? In this research, in which Neural Networks are used for a real-world regression problem, it is empirically shown that various Differential Evolution algorithms are falling into stagnation during Neural Network training. It means that after some time the individuals stop improving, or improve very occasionally, although the population diversity remains high. Similar behavior of Differential Evolution algorithms is observed for some, but not the majority of, benchmark problems. In the paper the impact of Differential Evolution population size, the initialization range and bounds on Neural Networks performance is also discussed.

Among tested algorithms only the Differential Evolution with Global and Local neighborhood-based mutation operators performs better than the Levenberg–Marquardt algorithm for Neural Networks training. This version of Differential Evolution also shows the symptoms of stagnation, but much weaker than the other tested variants. To enhance exploitation in the final stage of Neural Networks training, it is proposed to merge the Differential Evolution with Global and Local neighborhood-based mutation operators algorithm with the Trigonometric mutation operator. This method does not rule out the stagnation problem, but slightly improves the performance of trained Neural Networks.

1. Introduction

During the recent decade Differential Evolution (DE) [54,60,61], one of the population-based Evolutionary Computation methods, becomes a very popular tool for solving continuous optimization problems. There are probably two reasons of such popularity. Firstly, the good performance of DE for solving benchmark, engineering and real-world problems is widely acclaimed in the literature [15,34,39,50,54]. Secondly, comparing with many other recently developed heuristics, the basic DE is very simple and hence it is easily understood, encoded and implemented even by non-specialists.

However, despite the frequent claims of successful applications, the basic DE is not free from drawbacks. It suffers from the limited number of available steps, and hence the possibility of falling into stagnation. The choice of DE control parameters, that is required from the user, is a difficult task that may significantly affect the final performance. The basic DE algorithm is also blamed for slow or premature convergence [15,65]. For fifteen years overcoming the drawbacks of the basic DE method has motivated researchers to propose various improved DE versions. Today a large family of DE algorithms exists; their overview may be found in Refs. [15,42,54]. In Ref. [15] nine DE algorithms developed for single-objective unconstrained continuous optimization problems are granted an “important variant” status, but this is just a tip of the iceberg. Although since the publication of “No Free Lunch Theorems” [69] it has been widely accepted that no single “best” global optimization method can be developed, most novel DE algorithms are empirically shown to outperform the basic DE on many benchmark and real-world problems. However, due to the profusion of methods the choice of proper DE variant for the particular problem is a difficult task.

DE algorithms have been applied to a number of scientific problems [7,16,35,58,74], including Artificial Neural Networks (ANN) training [3,6,17,18,20,23,30,46,49]. Although ANN training aims at a bit different goal than classical optimization, as the ANN parameter values that are searched for should allow good generalization capabilities of the model, various metaheuristics have been widely

* Tel.: +48 22 6915 858; fax: +48 22 6915 915.
E-mail address: adampp@igf.edu.pl

http://dx.doi.org/10.1016/j.asoc.2014.03.039
1568-4946/© 2014 Elsevier B.V. All rights reserved.

used for such task for many years, as may be found in historical reviews published by Whitley et al. [67] and Yao [72] in 1990 and 1999. In Ref. [29] ANN training was even used together with benchmark problems to validate a good performance of a novel optimization algorithm. The metaheuristics, including DE methods, are usually claimed to be needed for ANN training due to two reasons: the gradient-based algorithms may stick in a local optimum, and some objective functions are not-differentiable. Although DE methods are frequently used for ANN training, the question arises if they are really efficient and successful.

In Ref. [30] it was shown that the basic DE method [60] is not suitable for training the Multi-Layer Perceptron ANN (MLP), probably the most popular type of ANN, due to slow convergence and inability of finding "good" optima. However, comparing with gradient-based algorithms, much slower convergence during ANN training is frequently observed when various kinds of Evolutionary Algorithms are used. This is a cost of exploration capabilities. Hence, for example, for ANN training by means of Evolution Strategies Mandischer [40] allowed much larger number of function calls than in cases when gradient-based algorithms were used. However, the slow convergence was not the only disadvantage of the basic DE. Its poor performance and inability to find reasonable ANN parameters were also claimed in Refs. [3,23,46].

Some suggestions how to improve DE performance on ANN training have been given in the literature. For example Fan and Lampinen [23] proposed a novel Trigonometric mutation operator to be used within the basic DE framework. Authors of Ref. [17] claimed that the more advanced self-adaptive DE variant proposed in Ref. [8] outperforms the basic DE version [60] on ANN training. Somehow contrary, in Ref. [44] it was found that self-adaptive DE variants do not perform better than the basic DE with control parameters tuned by means of off-line meta-optimization. In Ref. [3] hybridization of self-adaptive DE [8] with conjugant-gradient algorithm was proposed; the hybrid algorithm outperformed the basic DE, but its superiority over simple multiple-restart conjugant gradient was disputable. It must be noted that the large disadvantage of similar memetic approaches is that they cannot be used when objective function is not-differentiable, what is the main reason of searching for proper metaheuristics for ANN training. Authors of Ref. [20] proposed distributed DE algorithm for Pi-Sigma Higher-Order ANN training and found that its performance is only comparable with the back-propagation algorithm. In Ref. [46] six DE algorithms, namely: basic DE [60], Distributed DE with Explorative–Exploitative Population Families [65], Self-Adaptive DE (SADE) [55], DE with Global and Local neighborhood-based mutation operators (DEGL) [13], Grouping DE [45] and JADE [76] were compared with two Particle Swarm Optimization versions and the gradient-based Levenberg–Marquardt method (LM) [27,53] on MLP training for regression problem. The performance of all tested heuristics, with the exception of DEGL, turned out poorer than the performance of LM algorithm, a classic method for ANN training [27,75].

There is no doubt that since the publication of Ref. [30] significant improvement of DE methods has been achieved [15] and the novel algorithms outperform the basic DE version on various benchmark and real-world optimization problems. Unfortunately, from the literature survey presented above one may note that when applied to ANN training, where the purpose is to find the model parameters which allow good generalization capabilities, the new DE variants do outperform the basic DE, but not necessarily the gradient-based algorithms.

The major goal of the present paper is to clarify why the performance of popular DE algorithms is frequently disappointing when such methods are used to ANN training. This requires some insight into the behavior of a few popular and relatively new DE variants. Such DE variants are applied to ANN training for regression problem, namely daily river runoff forecasting based on large set of hydro-meteorological data, and to optimization of selected popular benchmark functions with the same dimensionality and maximum number of function calls. During algorithms' run the lowest, median and the largest Euclidean distances between the individuals in the decision space (such distances represent the available magnitudes of difference vectors that may be used by DE mutation operators) and the maximum, minimum and median fitness of all individuals in the current population are monitored. The importance of both features may be explained as follows.

DE algorithms are population-based. The population is composed of individuals, which create children in other positions in the decision space (the Euclidean distance between the position of a parent and a child may be termed a step size). The behavior of DE population has been explained in Refs. [24,65], where it has been noted that DE is an atypical Evolutionary Algorithm – most Evolutionary Algorithms require maintaining high population diversity during the whole search process, but for the proper functioning of DE the diversity loss is required. The common feature of DE algorithms is that during the generation the step sizes depend primarily on the difference vectors $\left\|\mathbf{x}_i - \mathbf{x}_k\right\|$, where $\mathbf{x}_i$, $\mathbf{x}_k \in \mathbf{R}^D$ are two individuals from the current population (or points in the decision space) and $\| \ \|$ represents the Euclidean norm. Although the scaling factors and crossover values also affect the step sizes, the distances between individuals are of major importance. When individuals are initially randomly generated in the decision space, distances between them are large, and the probability of finding some individuals close to each other is very low (at least in case of multi-dimensional space, see Ref. [59]). At this stage large exploratory steps prevail. As DE algorithms follow greedy selection (only the better of the parent-offspring pair survives to the next generation), when algorithm proceeds the individuals that survive are located in "better" parts of the decision space. In case of most benchmark problems this results in clustering of individuals. The distances between individuals within a cluster become small, but difference vectors of large magnitude are still easily obtained when the chosen individuals belong to different clusters, hence both large explorative and small exploitative steps are possible at that stage. As the time proceeds, the individuals are expected to concentrate around a few clusters and the exploitation steps become more frequent. Finally, if all individuals find their way to a single cluster located close to the most luring optimum, the possible magnitudes of difference vectors (i.e. distances between individuals) diminish and only exploitation is performed.

The question arises, what if the population remains scattered in the decision space and the distances between individuals remain large during the whole run. This may happen if fitness landscape is very "rough" and each individual finds its own "niche". The latter stages described above are not achieved and DE algorithms waist time on exploratory steps. Due to greedy selection all individuals in the current population are better than all its parents and offspring produced by such parents in the past, hence after long time the probability of successful large exploratory steps becomes very small, at least when local optima are not distributed regularly. The regular distribution of local optima is frequent in many benchmark problems, but when real-world data are considered, the regular distribution of local minima is rather uncommon. Montgomery [41] showed empirically that, at least in case of the basic DE, during later part of the run almost exclusively small exploitation steps are successful. Hence, the undesired effect of the lack of difference vectors of small magnitude may be the stagnation of DE algorithms [15,32,65], in other words the situation when the population stops proceeding toward the optimum, although the population diversity remains high. Even if some individuals occasionally generate better children that enter the population, both the average fitness of the population and the fitness of the best found solution do not