Electrical Power and Energy Systems 58 (2014) 257–265

Contents lists available at ScienceDirect

Electrical Power and Energy Systems

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

An optimized nearest prototype classifier for power plant fault diagnosis using hybrid particle swarm optimization algorithm

Xiaoxia Wang [a,*], Liangyu Ma [a], Tao Wang [b]

[a] School of Control and Computer Engineering, North China Electric Power University, Baoding, HeBei 071003, PR China
[b] School of Mathematics and Physics, North China Electric Power University, Baoding, HeBei 071003, PR China

ARTICLE INFO

Article history:
Received 8 March 2013
Received in revised form 7 January 2014
Accepted 18 January 2014

Keywords:
Nearest prototype classifier
Particle swarm optimization
Constructive approach
Power plant
Fault diagnosis

ABSTRACT

Correct and rapid fault diagnosis is of great importance for the safe and reliable operation of a large-scale power plant. It is a difficult task, however, due to the structural complexity of a power plant, which needs to deal with hundreds of variables simultaneously in case of fault occurrence. A novel nearest prototype classifier is proposed in this paper to diagnose faults in a power plant. A constructive approach is employed to automatically determine the most appropriate number of prototypes per class, while a hybrid particle swarm optimization (HGLPSO) algorithm is used to optimize the position of the prototypes. The aim is to generate an automatic process for obtaining the number and position of prototypes in the nearest prototype classifier with high classification accuracy and low size. The effectiveness of the HGLPSO classifier is evaluated on eight real world classification problems. Finally, the classifier is applied to diagnose faults of a high-pressure feedwater heater system of a 600-MW coal-fired power unit. The obtained results demonstrate the validity of the proposed approach.

1. Introduction

On-line monitoring and fault diagnosis of a large-scale power plant is of prime importance for process operation and equipment maintenance with the potential advantages of improving safety, reliability, and availability of the power unit [1]. The main target is to monitor the operating state of the plant and identify a fault at its earliest developing stage, by interpreting the different evolutionary patterns of the involved process variables, so as to assist the operator to take fast and corrective actions in due time.

Various approaches have been investigated and successfully applied for fault diagnosis in the past decade, such as artificial neural networks [2–8], fuzzy logic [9], neuro-fuzzy based techniques [10], template matching [11], independent component analysis [12], support vector machines [13] and evolutionary algorithm [14,15]. The Nearest Prototype Classifier (NPC) is perhaps among the simplest and most intuitively motivated classifiers in pattern recognition, which assigns an unknown pattern to the class of the nearest prototype in a set of classified prototypes. Instead of making use of all the data points of a training set, NPC relies on a set of appropriately chosen prototype vectors. This makes the method computationally more efficient, because the number of items which must be stored and to which a new pattern must be compared for classification is considerably less [16]. Developing schemes for obtaining prototypes from examples, however, has proved to be a difficult problem [17].

There are two different approaches to deal with this problem. One approach is to select an appropriate subset from the original training data [18–20], while another approach is to generate a new set of artificial prototypes with lower size and higher classification accuracy [21–23]. Evolutionary algorithm has been successfully used to optimize the position of prototypes with promising results [24]. Artificial immune algorithm [25] and differential evolution (DE) algorithm [26–29] are two effective evolutionary techniques for position adjustment. A good study of the noise tolerance in DE classifier can be found in [30]. Particle swarm optimization (PSO) algorithm has shown an excellent performance in continuous classification problem. Recent advances in PSO for prototype adjustment are PSO(5) [31], Michigan PSO [32] and AMPSO [33]. This work applies PSO algorithm to generate an optimal set of prototypes in order to maximize the classification accuracy and minimize the number of prototypes.

PSO algorithm is a population-based stochastic optimization technique proposed by Kennedy and Eberhart in 1995 [34], which was inspired by the social behavior of animals, such as bird flocking and fish schooling. Compared to GA, PSO takes less time for

* Corresponding author. Address: Department of Computer, School of Control and Computer Engineering, North China Electric Power University, No. 619 Yonghua North Street, Baoding, Hebei 071003, PR China. Tel.: +86 13933225829.
E-mail address: wtwxx@163.com (X. Wang).

http://dx.doi.org/10.1016/j.ijepes.2014.01.016
0142-0615/

each function evaluation as it does not use many of GA operators like mutation, crossover and selection operator. Owing to the simple model, easiness to be implemented and fast convergence, PSO has been successfully applied to many fields [35–38]. Also, it is particularly attractive for fault prototype construction because particle swarms will discover the prototype positions as they fly within the solution space [31]. However, lack of diversity of the swarm, particularly during the latter stages of the optimization, may lead particles to converge to a local optimum prematurely. Recently, many attempts have been made to improve the diversity of the population, considering the behavior of the particles in the swarm during the search. Several proposals are presented on this topic, such as FDR-PSO [39], HPSO-TVAC [40], FIPS [41], CLPSO [42]. Besides, some hybrid PSO algorithms were also proposed, such as those based on genetic algorithm [43], evolutionary algorithm [44] and simulated annealing [45].

In order to improve the performance, we propose a hybrid PSO algorithm, referred to as HGLPSO algorithm combining global search and local search in this paper. In addition, time-varying evolution is also introduced into the algorithm. The major consideration of this modification is to avoid premature convergence and to enhance the global search capability of the particles by providing additional diversity. Based on the HGLPSO algorithm, a novel constructive learning approach is proposed to determine the most appropriate number of prototypes per class and their best position with the double objective of low size and high classification accuracy. After generating an optimal set of prototypes for each class, a new input pattern is classified by the nearest prototype to this pattern. The optimized nearest prototype classifier is referred as HGLPSO classifier in this paper.

The differences between the HGLPSO classifier and MPSO/AMPSO approach are in the following aspects.

1) The HGLPSO classifier follows a constructive scheme in which an incremental approach is employed to automatically find the smallest reduced set of prototypes, while the HGLPSO algorithm is used to adjust the position of the prototypes. In MPSO, a fixed population of particles is limited. AMPSO adjusts the number of prototypes and their classes by particle reproduction.
2) HGLPSO allows the particles to profit not only from their own discoveries and the discoveries of the swarm as a whole, but also from the discoveries of their neighborhood in each generation. Particles use attraction and repulsion rules in MPSO/AMPSO.

In experiments on 8 well-known real world classification problems with different properties, the classification accuracy and the number of prototypes in the solution of the HGLPSO classifier are investigated and its performance is compared with recent PSO-based classifiers and other classical classifiers. Finally, a high-pressure feedwater heater system of a 600-MW coal-fired power unit is taken as a target system for investigation. Several faults are simulated in a power plant simulator and diagnosed by the HGLPSO classifier. The obtained results demonstrate the validity of the proposed approach.

The remainder of this paper is organized as follows. Section 2 describes the original PSO algorithm and the proposed HGLPSO algorithm. In Section 3, the detail of the proposed HGLPSO classifier is reported. Using a set of 8 real world classification problems, comparisons of the HGLPSO classifier with other classifier are given in Section 4. Section 5 presents how the proposed approach is used for fault diagnosis in the feedwater heater system of a 600-MW coal-fired power plant and the experimental results are reported in this section. Finally, conclusions are drawn in Section 6.

2. HGLPSO algorithm

2.1. PSO algorithm

In PSO algorithm, the system initially has a swarm of random solutions. Each potential solution, called a particle, is given a random initial velocity and is flown through the problem space. These particles find the global best position by competition as well as cooperation among themselves after some iterations. Supposing the dimension of a searching space is D, the total number of particles is S, the position of the ith particle is expressed as $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$; the velocity of the ith particle is represented as $\mathbf{v}_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$. The best previous position (which possesses the best fitness value) of the ith particle is denoted as $\mathbf{p}_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, which is also called *pbest*. The index of the best *pbest* among all the particles is represented by the symbol g and the position $\mathbf{p}_g = (p_{g1}, p_{g2}, \ldots, p_{gD})$ is also called *gbest*. Therefore the particle updates its velocity and position according to the following equations:

$$\mathbf{v}_i(k+1) = w\mathbf{v}_i(k) + c_1 r_1(\mathbf{p}_i(k) - \mathbf{x}_i(k)) + c_2 r_2(\mathbf{p}_g(k) - \mathbf{x}_i(k)) \quad (1)$$

$$\mathbf{x}_i(k+1) = \mathbf{x}_i(k) + \mathbf{v}_i(k+1) \quad (2)$$

where k is the indices of iteration number; c_1 and c_2 are the cognitive and social acceleration constants; r_1 and r_2 are uniformly distributed random numbers in the range $[0,1]$. In addition, velocities of particles on each dimension can be clamped by a user defined maximum velocity $V_{\max}$. The inertia weight w is employed to control the impact of the previous history of velocities on the current velocity. A larger inertia weight facilitates global exploration, while a smaller inertia weight tends to facilitate local exploration to fine-tune the current search area. Suitable selection of the inertia weight can provide a balance between global exploration and local exploitation abilities, and thus requires fewer iterations on average to find the optimum [46,47]. Although research on the inertia weight is still in progress, it appears that a good general approach is to decrease the inertia weight linearly from a relatively large value to a small one through the course of the PSO run [47].

2.2. HGLPSO algorithm

In the PSO domain, there are two main variants: global version PSO (GPSO) and local version PSO (LPSO). The algorithm described above is the global version. In GPSO, a particle learns from the personal best position *pbest* and the best position *gbest* achieved so far by the whole population. In LPSO, a particle's velocity is adjusted according to its personal best position *pbest* and the best position *lbest* achieved so far within its neighborhood, instead of *gbest*. The GPSO and the LPSO have advantages and disadvantages. The former converges fast but may easily be trapped in a local optimum while the latter shows slow convergence but can easily jump out of local optimum. In order to take advantage of the complementary property of GPSO and LPSO, the HGLPSO algorithm is proposed to combine global search and local search, which shares information of *pbest*, *gbest* and *lbest*. The mathematical representation is given as shown in Eq. (2) and

$$\begin{aligned}\mathbf{v}_i(k+1) = {} & w(k)\mathbf{v}_i(k) + c_1(k) r_1(\mathbf{p}_i(k) - \mathbf{x}_i(k)) \\ & + c_2(k) r_2\left[\alpha^k(\mathbf{p}_l(k) - \mathbf{x}_i(k)) + (1-\alpha^k)(\mathbf{p}_g(k) - \mathbf{x}_i(k))\right]\end{aligned} \quad (3)$$

$$w(k) = w_{initial} + (w_{final} - w_{initial})\frac{k}{k_{\max}} \quad (4)$$

$$c_1(k) = c_{1initial} + (c_{1final} - c_{1initial})\frac{k}{k_{\max}} \quad (5)$$