Learning Features for Discriminative Behavior Analysis of Evolutionary Algorithms via Slow Feature Analysis

Chengshan Pang
School of Information Science and Technology, University of Science and Technology of China
Hefei, Anhui, China
nuaapcs@mail.ustc.edu.cn

Mang Wang
School of Information Science and Technology, University of Science and Technology of China
Hefei, Anhui, China
mangwang@mail.ustc.edu.cn

Weiming Liu
School of Information Science and Technology, University of Science and Technology of China
Hefei, Anhui, China
weiming@mail.ustc.edu.cn

Bin Li
School of Information Science and Technology, University of Science and Technology of China
Hefei, Anhui, China
binli@ustc.edu.cn

ABSTRACT

Evolutionary algorithms (EAs) are a kind of stochastic optimization methods, which have been testified to be powerful in solving many real-world hard problems in past decades. But till now, we are still short of effective methods to represent and investigate their collective behaviors in various environments, which are very useful for researchers and engineers in Evolutionary Computation to understand the algorithms better. This paper is a preliminary effort to tackle above issue. We attempt to analyze the generation-wise collective behavior of EAs via an approach called feature learning. An unsupervised feature learning framework based on Slow Feature Analysis (SFA) is presented to extract discriminative features from the generation-wise collective behavior data of several EAs on various fitness landscapes, with the purpose of finding out whether there exist differences between the searching behavior of different EAs running on the same fitness landscape; and whether there are differences between the behavior of one algorithm running on different fitness landscapes. Besides, the relationship between the fitness landscape and the searching behavior of EA is also studied. In the experiments, several typical EAs and classical benchmark functions with typical landscapes are selected as the study subjects. The collective behaviors of various EAs are visualized and compared in the extracted feature space.

GECCO'16 Companion, July 20-24, 2016, Denver, CO, USA

DOI: http://dx.doi.org/10.1145/2908961.2935617

Keywords

Evolutionary algorithms; Collective behavior; Feature learning; Slow feature analysis; Fitness landscape

1. INTRODUCTION

An evolutionary algorithm (EA) is a stochastic optimization technique. EA is initialized with a set of possible solutions, and then gradually updated by multiple iterations to find regions containing optimal solutions. The previous research mainly concentrates on the outcome of EAs, relatively little attention is paid on the searching behavior of EAs. As a result, although EAs have been put to use with success in many fields, it is often hard for users to know how the solutions were discovered.

Over the years, a number of visualization techniques have been proposed to analysis the evolutionary process of EA. In [8], a visualization tool called GAVEL examines how crossover and mutation operations affect the final result in a generational GA. Evolutionary Visual Exploration (EVE) system, presented in [3], is dedicated to analyzing the behavior of EA by visualizing the multidimensional datasets, which are provided by EA in different viewpoints. Additionally, a generic tool named ELICIT is created in [5], it gives an access to the visualization of both phenotypes and genotypes. All of these tools tend to focus on the visualization of population data and the evolution process of fitness values [5]. Nevertheless, the convergence of EA usually go through iterations, and a set of high dimensional population data are produced in each iteration. In the whole process, a large amount of data is generated. Direct observation and analysis of these data is difficult.

In [11], an empirical method is presented to investigate the collective behavior of population-based search algorithm, the key contribution of this work was that some measurements were developed to identify emergent properties of population, i.e. exploitation and exploration, with the purpose to characterize the algorithms. Based on this work, In [12],

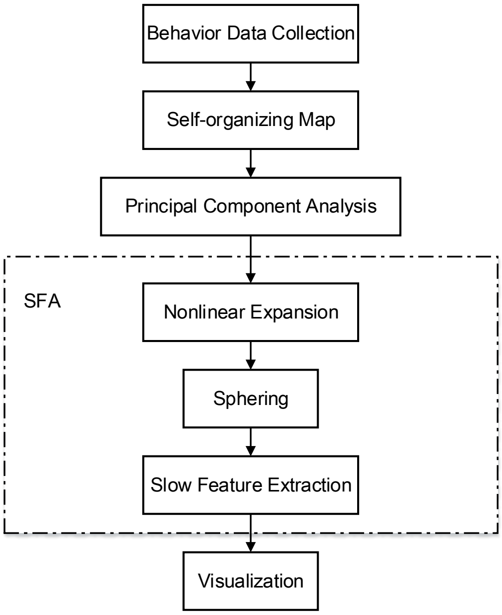

Figure 1: The flowchart of the feature extraction process.

the authors developed a set of features to characterize the collective behavior, and then defined an indicator of the exploitation behavior of an EA.

This paper tries to represent and analyze the collective behavior of EAs via an approach called feature learning. Different from the works presented in [11] and [12], in which the features were defined by the authors [11]. The purpose of this paper is to develop method to extract discriminative features of the collective behavior of EAs via automatic learning, other than human defining. An unsupervised feature learning framework based on SFA is presented to meet this purpose. Four classical EAs [4]: classical evolutionary programming (CEP) [15], differential evolution (DE) [10], evolution strategy (ES) [2] and genetic algorithm (GA) [7], are selected to generate collective behavior data for study. Meanwhile, six functions are used as the test bed of various fitness landscapes [9]: Ackley's function, the Elliptic function, Rastrigin's function, Rosenbrock's function, Schwefel's problem 1.2 and the Sphere function. In the experiments, data produced by different algorithms from the same current population of solutions on the same fitness landscape, and data produced by the same algorithm from the same current population of solutions on different fitness landscape, are analyzed with a method based-on Slow Feature Analysis (SFA). Obvious aggregation and stable similarity relationship of collective behavior data of four EAs are observed in the feature space constructed with the slow features extracted by the presented method.

The rest of the paper is organized as follows: Section II introduces the framework of feature learning based on slow feature analysis. Section III outlines the experiments setup, and presents the experiment results and analysis in detail. Finally, conclusions are given in section VI.

2. COLLECTIVE BEHAVIOR FEATURE LEARNING BASED ON SFA

An Evolutionary Algorithm can be treated as a input-output system working in the solution space of a specific problem. Input of the system is a set of solutions with their fitness, and the output is a set of new solutions. We can study the discrepancies in the search behavior of different EAs by analyzing the one generation offspring, which were generated from the same initial population according to different evolutionary rules. At the same time, the search behavior of each EA on different landscapes can also be studied from such one generation offspring. As an unsupervised feature learning algorithm, slow feature analysis (SFA) can learn slowly varying signals from multiple dimensional input data. The learned slow features to some extent reveal the essential features of the input-output system, here the EA.

The procedure of EA feature learning is illustrated in Figure 1. Its key components are explained below.

2.1 Behavior Data Collection

The features of EAs' behavior are to be learned from the offspring population generated by EAs from the current population. Firstly, select a small area and randomly initialize a population within the area as the parent population. Then produce offspring with EAs on different fitness landscapes defined by the six test functions. Note that each algorithm with one landscape only performs one single iteration. For each algorithm with each landscape, the operation is repeated 500 times and 500 groups of offspring are produced. Given the stochastic nature of EA operation, the 500 groups of offspring should not be identical, but should contain same information connected to the inner mechanism of the EA.

2.2 Self-organizing Map (SOM) for Representation Normalization

It is the distribution of the population in the solution space, not the permutation of individuals in the population, that is of the most concern in behavior analysis of EAs. Given that, when represented by chaining the individuals sequentially, the representation of the population may change when the individuals change their positions in the chain, while the distribution of the population will not change actually. Therefore, the representation of the population need to be normalized to be invariant under the position change of individuals in the representation.

In this paper, Self-Organizing Map (SOM) [6] is adopted to implement the representation normalization of EA's behavior data. After being well trained, the SOM can map the individuals in original solution space onto a 2 dimensional grid representation, in which, the hitting times of each node is recorded and used to represent the value of the node, and the neighborhood relationship between nodes are kept from high dimensional solution space to the low dimensional SOM representation space.

In this study, a 2-D SOM grid is trained in advance from a large number of data sampled evenly randomly from the solution space, the size of the SOM grid is set to 100*100, and the dimension of the sample vector is consistent with the test problems adopted, that is, 30. To guarantee training quality of SOM, t-SNE [13] is adopted as a pre-training technique to get the well-ordered initialization of SOM.

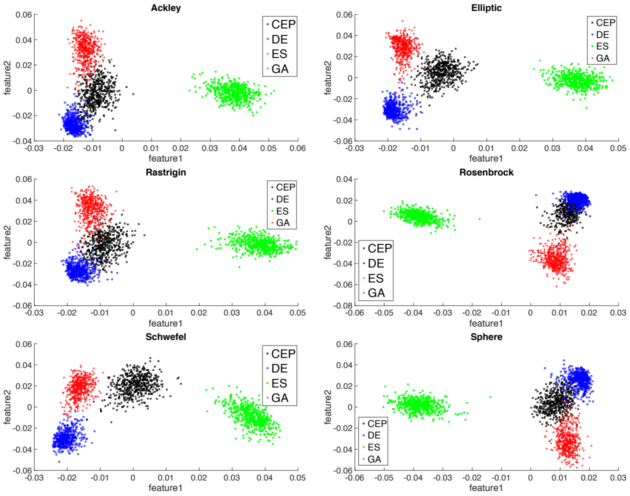

Figure 2: Sample distribution of 4 EAs on the same fitness landscape of 6 benchmark functions in the feature space constructed by slow features.

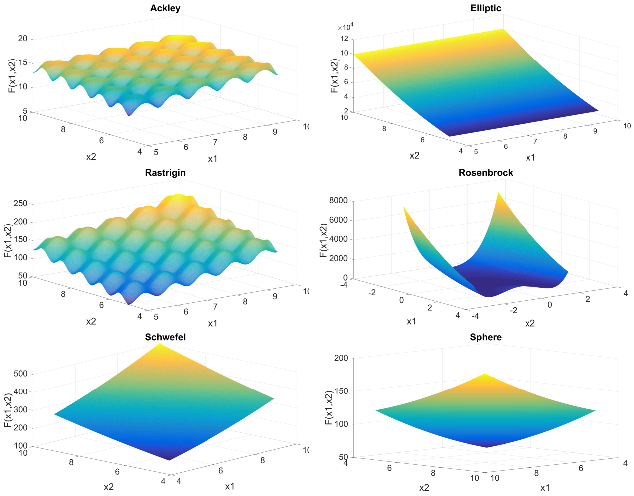

Figure 3: Six benchmark functions in 2-D space $[5,10]^2$.

2.3 Principal Component Analysis (PCA) for Discriminative Feature Extraction

Given the 2-D representation of one generation offspring of EAs, and 500 sampled offspring populations for each of four EAs on each of six test fitness landscapes, PCA is adopted to extract linear discriminative features from sample sequences of all experiment configurations. The extracted features contain both the stable information related to the inner mechanism of the EA, and random information caused by the random nature of the EA. To obtain the former information, Slow feature analysis is applied to the new sample sequences represented in PCA features to extract slow features corresponding to the stable part in the sample sequences.

2.4 Slow Feature Analysis (SFA) for Stable Feature Extraction

SFA was originally designed for extracting slowly varying or invariant components from the input time series. The slow features learned by SFA are believed to be able to represent the stable part of the subject under investigation, and be useful for classification [14]. When being applied to the analysis of EAs, SFA plays a role to filter out the stochastic part of EAs, and the stable part of EAs are retained, which, in a sense, can be regarded as the feature of EAs.

The fundamental of SFA can be described like this [1]: Given an input signal $x(t) = [x_1(t), x_2(t), \ldots, x_I(t)]^T$, find the input-output function $g(t) = [g_1(t), g_2(t), \ldots, g_J(t)]^T$, so that the output signal $y_j(t) = g_j(x(t)), j \in \{1, 2, \ldots, J\}$. $g_j(x)$ can be expressed as a weighted sum over a set of K nonlinear functions $h_k(x)$, namely $g_j(x) = \sum_{k=1}^{K} w_{jk} h_k(x)$. Thus, $g_j(x(t)) = w_j^T h(x(t)), j \in \{1, 2, \ldots, J\}$, the nonlinear expansion turn the problem into a linear one. Denote $z(t) = h(x(t))$, then

$$\text{Minimize } \Delta(y_j) = \langle \dot{y}_j^2 \rangle = w_j^T \langle \dot{z}\dot{z}^T \rangle w_j \tag{1}$$

Under the constraints:

$$\langle y_j \rangle = w_j^T \langle z \rangle = 0 \tag{2}$$

$$\langle y_j^2 \rangle = w_j^T \langle zz^T \rangle w_j = 1 \tag{3}$$

$$\forall j' < j : \langle y_{j'} y_j \rangle = w_j^T \langle zz^T \rangle w_j = 0 \tag{4}$$

Constraints (2) and (3) aim at avoiding the trivial solution $g_j(x) = 0$, and constraint (4) guarantees that different signal components to be uncorrelated.

The constraint (3) is integrated in the objective function (1), then we obtain:

$$\text{Minimize } \Delta(y_j) = \frac{w_j^T \langle \dot{z}\dot{z}^T \rangle w_j}{w_j^T \langle zz^T \rangle w_j} = \frac{w_j^T A w_j}{w_j^T B w_j} \tag{5}$$

The weight vector w_j that minimize this equation corresponds to the eigenvector of A [16].

In this paper, the input sequence is constituted by the samples represented in PCA features. The samples from same EA are put together in the sequence. Reconstruction of the sequence and reformulation of the objective function are as follows:

$$\text{Minimize } \Delta(y_j) = a \cdot \sum_{m=1}^{C} \sum_{k,l=1}^{P_m} (g_j(x_k^{(m)}) - g_j(x_l^{(m)}))^2 \tag{6}$$

Where C means the number of experimental configurations (EA, fitness landscape), and there are P_m samples $x_1^{(m)}, x_2^{(m)}, \cdots, x_{P_m}^{(m)}$ for a certain experimental configuration. Accordingly, the constraints are reformulated to:

$$\frac{1}{P} \sum_{m=1}^{C} \sum_{k=1}^{P_m} g_j(x_k^{(m)}) = 0 \tag{7}$$

$$\frac{1}{P} \sum_{m=1}^{C} \sum_{k=1}^{P_m} g_j(x_k^{(m)})^2 = 1 \tag{8}$$

$$\forall i < j, \frac{1}{P} \sum_{m=1}^{C} \sum_{k=1}^{P_m} g_i(x_k^{(m)}) g_j(x_k^{(m)}) = 0 \tag{9}$$

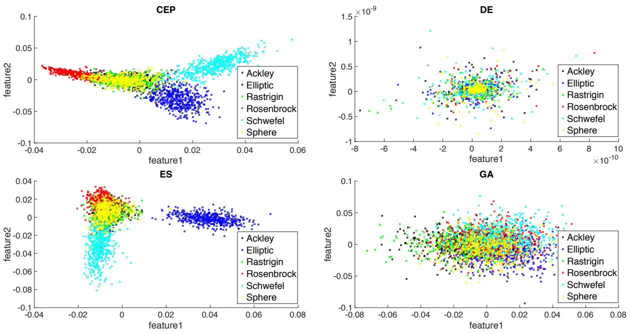

Figure 4: Sample distribution of EA on different fitness landscapes.

Where $P = \sum_{m=1}^{C} P_m$ is the total number of samples.

2.4.1 Nonlinear Expansion

Assume $x = [x_1, x_2, \cdots, x_n]^T$ is the sample sequence represented in PCA features, then the expanded signal is:

$$z = h(x) = [x_1, x_2, \cdots, x_n, x_1x_1, x_1x_2, \cdots, x_nx_n]^T \qquad (10)$$

And the dimension of z is $(n + n(n+1)/2)$.

2.4.2 Sphering

The expanded signal needs to be sphered to satisfy the constraints (7), (8), (9). Let $E = [e_1, e_2, \cdots, e_k]$ be the matrix whose columns are the unit-norm eigenvectors of the covariance matrix $C_z = E\{zz^T\}$, meanwhile, let $D = diag(d_1, d_2, \cdots, d_k)$ be the diagonal matrix of the eigenvalues of C_z, thus a transform matrix $V = D^{-\frac{1}{2}}E^T$ is calculated. Finally, compute the sphered matrix:

$$s = [s_1, s_2, \cdots, s_k]^T = Vz = D^{-\frac{1}{2}}E^T z \qquad (11)$$

2.4.3 Slow Feature Extraction

The constraint (8) can also be integrated in the objective function (6) as the same as that in equation (5), then the matrix A referred to in equation (5) can be calculated by:

$$A = a \cdot \sum_{m=1}^{C} \sum_{k,l=1}^{P_m} (x_k^{(m)} - x_l^{(m)})(x_k^{(m)} - x_l^{(m)})^T \qquad (12)$$

The input-output functions $g(x)$ can be derived and sorted according to corresponding eigenvalues of A, then the first two or three slow features can be driven from the corresponding input-output functions.

3. EXPERIMENT RESULTS AND ANALYSIS

Four classical EAs (GA, ES, CEP, DE) are chosen as the exemplar algorithms to investigate the ability and legality of the method presented. Detailed information of these algorithms is provided in the Appendix.

Six classical benchmark functions [9] are adopted to provide various fitness landscapes for investigation. These six functions represent different types of problem with different fitness landscapes. The dimensionality of all the functions is set to 30. And the global optimum is the solution that has the minimum function value. Detailed information of the benchmark functions is provided in the Appendix.

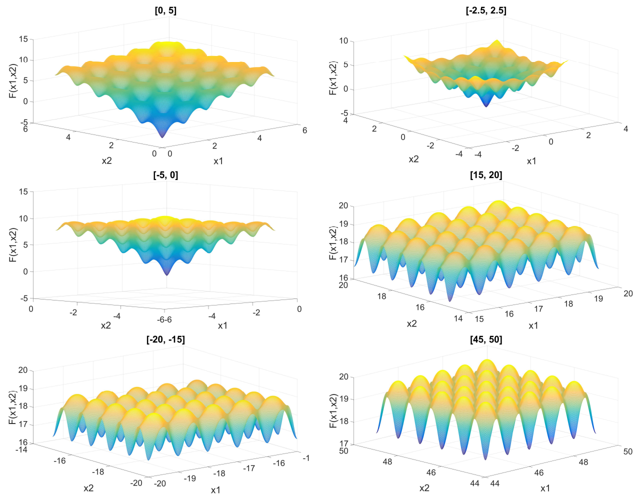

Figure 5: Fitness landscape of Ackley's Function on different areas in 2-D.

Four experiments are designed to analyze the collective behavior of EAs : A) The behavior of different EAs on the same fitness landscape; B) The behavior of same EA on fitness landscape of different benchmark functions; C) The behavior of EA on the different fitness landscapes of same benchmark function; D) The behavior of different EAs on different benchmark functions. In this section, the experiment results are given in detail.

3.1 The behavior of different EAs on the same fitness landscape

In this experiment, the initial population is randomly sampled in area $[5, 10]^{30}$. The one-generation offspring generated by four algorithms with the same initial population on the same benchmark function are collected as the sample data of collective behavior of EAs, with the purpose to find out whether there exist differences between the collective behaviors of different evolutionary algorithms when executed on the same condition. There are $500(\text{offspring populations}) * 4(\text{EAs}) = 2000$ samples in the input sequence, they are mapped in the 2-D feature space extracted via the process presented in Section 2, as shown in Figure 2.

In Figure 2, it can be observed that the sample distribution of four algorithms exhibit very clear aggregative and discriminative nature in the feature space constructed by slow features. Beyond that, for all of the fitness landscapes of six benchmark functions, the one-generation behavior of four EAs exhibits a sort of stable similarity relationship, that is the behavior of CEP, DE and GA are more similar compared to that of ES.

3.2 The behavior of EA on different fitness landscapes

In this experiment, the initial population is also randomly sampled in $[5, 10]^{30}$, and the input is constituted by the offspring created by one algorithm under different benchmark functions. That is to say, there are $500(\text{offspring populations}) * 6(\text{benchmark functions}) = 3000$ samples in the input sequence for SFA. Different benchmark functions correspond-

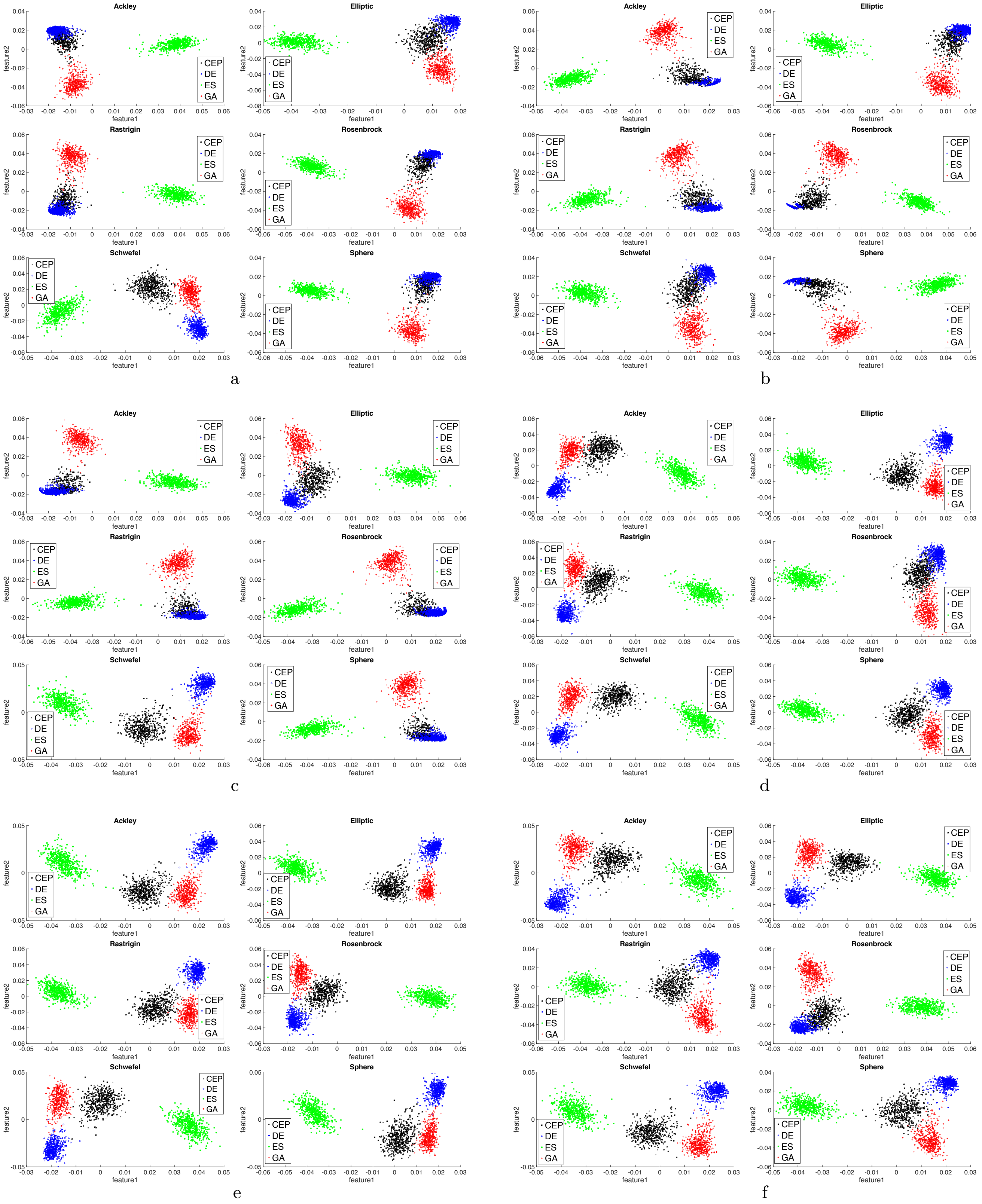

Figure 6: Sample distribution of EAs on different landscapes of different benchmark functions in the obtained feature space. The landscapes corresponding to different areas in the solution space of different benchmark functions, the bounds are: a. [0, 5]; b. [-2.5, 2.5]; c. [-5, 0]; d. [15, 20]; e. [45, 50]; f. [-20, -15].

ing to different landscapes, so the behavior differences of one algorithm on different landscapes can be learned. Figure 3 illustrates the fitness landscape of 6 benchmark functions on the 2-D solution area $[5, 10]^2$. The sample distribution of four EAs are displayed in Figure 4.

In Figure 4, it can be observed that CEP and ES exhibit discrimination on the landscape of Elliptic function and Schwefel's Problem 1.2 compared with the other four landscapes. While no discrimination is observed for DE and GA on all fitness landscapes under test. It may imply that the behavior of DE and GA are not sensitive to the six fitness landscapes, while the behavior of CEP and ES are somehow sensitive to the fitness landscapes. In order to further investigate the relationship between the search behavior of EA and the fitness landscape, the third experiment is conducted.

3.3 The behavior of EA on different fitness landscapes of one benchmark function

In experiment A) and B), the range of initialization is $[5, 10]$. In order to find the relationship of search behavior and the fitness landscapes, the initial landscape of the parent group need to be changed in this experiment. To focus the investigation on the fitness landscape other than the solution distribution, instate to change the sampling area of solution space, the benchmark function is shifted in the solution space accordingly. For example, instate of changing sampling area from $[5, 10]$ to $[0, 5]$, an offset -5 is added to all coordinates in benchmark functions. In this way, several different landscapes are obtained. Take the Ackley's Function as an example, the different landscapes of it are illustrated in Figure 5.

In this experiment, the same configuration as Experiment A) is adopted but is repeated six times, with each time different area of solution space is investigated. All the results are showed in Figure 6.

From Figure 6, it can be observed again that, on different fitness landscapes, in the feature space constructed by slow features extracted by the method presented in this paper, the sample distribution of four algorithms exhibit clear aggregative and discriminative nature. Additionally, for all of the fitness landscapes, the one-generation behavior of four EAs exhibits a sort of stable similarity relationship.

3.4 The behavior of different EAs under different benchmark functions

In previous experiments, benchmark functions or EA are controlled respectively to investigate the search behavior of different EAs on same landscape or same EA on different landscapes. In this section, the offspring of four EAs on all landscapes are put together for analysis. That is to say, there are 500(offspring populations)$*$4(EAs)$*$6(benchmark functions)=12000 samples in the input sequence for SFA. The experiment configuration is repeated six times with six different areas of solution space are investigated respectively. The sample distribution in the feature space constructed by the first three slowest features are showed in Figure 7.

Since in each figure, samples of EAs on all benchmark functions are included, and the samples of same EA on different benchmark functions are hard to separate, in this experiment, we just mark samples of different EAs with different colors, regardless of their difference between benchmark functions.

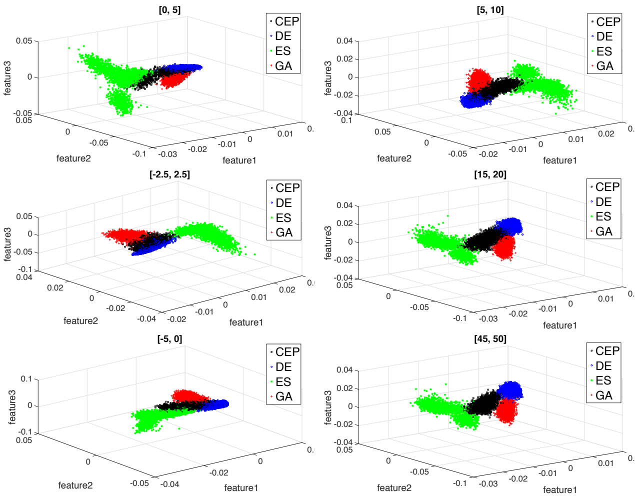

Figure 7: Sample distribution of EAs on different landscapes of all benchmark functions in the obtained feature space, the samples of different EAs are represented by different colors and shapes of mark.

From Figure 7, same conclusion as that obtained from Figure 6 can be obtained.

4. CONCLUSIONS

This paper presents a feature learning method for analyzing the discriminative collective behavior of EAs. The key component of the method is an unsupervised learning method named Slow Feature Analysis (SFA). Experiments based on four EAs and six benchmark functions are designed to investigate the ability of the presented method. Experiment results show that 1) the presented method can extract discriminative features of one-generation collective behavior of different EAs; 2) In the feature space constructed by the extracted slow features, the aggregative and discriminative nature of the samples are observed clearly; 3) In the feature space, the similarity relationship between different EAs are somehow stable given the variation of fitness landscapes; 4) The one-generation collective behaviors of DE (or GA) on different fitness landscapes adopted in this paper are not distinguishable, but this should not lead to the conclusion that the performance of DE (or GA) on different problems (fitness landscapes) are same.

5. ACKNOWLEDGMENT

The work is supported by the National Natural Science Foundation of China under grand No. 61473271 and No. 61331015.

6. REFERENCES

[1] P. Berkes. Pattern recognition with slow feature analysis. *Computational Neuroscience*, 2005.

[2] H.-G. Beyer and H.-P. Schwefel. Evolution strategies–a comprehensive introduction. *Natural Computing*, 1(1):3–52, March 2002.

[3] W. Cancino, N. Boukhelifa, A. Bezerianos, and E. Lutton. Evolutionary visual exploration: experimental analysis of algorithm behaviour. In

Proceedings of the 15th annual conference companion on Genetic and evolutionary computation, pages 1373–1380. ACM, July 2013.

[4] T. Collins. *The Application of Software Visualization Technology to Evolutionary Computation: A case study in genetic algorithms*. PhD thesis, The Open University, May 1998.

[5] A. Cruz, P. Machado, F. Assunção, and A. Leitão. Elicit: Evolutionary computation visualization. In *Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference*, pages 949–956. ACM, July 2015.

[6] W. Duch and A. Naud. Multidimensional scaling and kohonen's self-organizing maps. In *Proc. 2nd Conf. Neural Networks and their Applications, Szczyrk, Poland*, pages 138–143, April 1996.

[7] D. E. Goldberg et al. *Genetic algorithms in search optimization and machine learning*. Addison-wesley Company, Reading, Menlo Park, 1989.

[8] E. Hart and P. Ross. Gavel-a new tool for genetic algorithm visualization. *Evolutionary Computation, IEEE Transactions on*, 5(4):335–348, August 2001.

[9] X. Li, K. Tang, M. N. Omidvar, Z. Yang, K. Qin, and H. China. Benchmark functions for the cec 2013 special session and competition on large-scale global optimization. *Gene*, 7(33):8, December 2013.

[10] R. Storn and K. Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, December 1997.

[11] M. Turkey and R. Poli. An empirical tool for analysing the collective behaviour of population-based algorithms. In *Applications of Evolutionary Computation*, pages 103–113. Springer, April 2012.

[12] M. Turkey and R. Poli. A model for analysing the collective dynamic behaviour and characterising the exploitation of population-based algorithms. *Evolutionary Computation*, 22(1):159–188, February 2014.

[13] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, November 2008.

[14] L. Wiskott and T. J. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, April 2002.

[15] X. Yao, Y. Liu, and G. Lin. Evolutionary programming made faster. *Evolutionary Computation, IEEE Transactions on*, 3(2):82–102, August 1999.

APPENDIX

Evolutionary algorithms and benchmark functions for analysis in this paper are introduced in this part.

A. EVOLUTIONARY ALGORITHMS

Four classical EAs are chosen to investigate the ability and legality of the method presented above. The introduction of the four algorithms are described here, and the parameters of EAs are listed in TABLE 1:

A.1 Classical EP (CEP)

CEP [15] generates the offspring by mutation operation, one of the significant characteristics in CEP is that the recombination operator is inexistent. The mutation is realized by adding random value to old individuals.

Table 1: EAs' parameters

Algorithm	Fix Parameters	Initialization of self-adapted parameters
CEP	$\tau = \frac{1}{\sqrt{2\sqrt{D}}}$ $\tau' = \frac{1}{\sqrt{2D}}$	$\eta_i(j) = 3$
DE	$F = 0.5$ $CR = 0.9$	
ES	$\tau = \frac{1}{\sqrt{2\sqrt{D}}}$ $\tau' = \frac{1}{\sqrt{2D}}$ $\frac{\lambda}{\mu} = 7$	$\eta_i(j) = 3$
GA	$Pc = 0.8$ $Pm = 0.05$ $Re = 0.1$	

a. Mutation:

$$x_i{}'(j) = x_i(j) + \eta_i(j) N_j(0,1) \tag{13}$$

$$\eta_i{}'(j) = \eta_i(j) exp(\tau' N(0,1) + \tau N_j(0,1)) \tag{14}$$

Where $x_i{}'(j)$, $x_i(j)$, $\eta_i{}'(j)$ and $\eta_i(j)$ denote the j-th dimensionality of the vectors $x_i{}'$, x_i, $\eta_i{}'$ and η_i, respectively. Besides, for each j in equation (14), $N_j(0,1)$ represents a random value which subjects to standard normal distribution.

b. Selection: $(\mu + \lambda)$ selection. That is, μ parents are allowed to breed λ offspring, and these parents and offspring populations are merged, then select the best μ individuals from the $(\mu + \lambda)$ individuals.

A.2 DE

DE [10] creates new offspring by adding the weighted difference between two parents to a third parent. The operations are listed below [10]:

a. Mutation:

$$x_i{}' = x_{r1} + F \cdot (x_{r2} - x_{r3}) \tag{15}$$

b. Crossover:

$$x_i{}'' = \begin{cases} x_i & \text{if } rand > CR \\ x_i{}' & \text{if } rand \leq CR \end{cases} \tag{16}$$

c. Selection: greedy criterion. The new generated individual $x_i{}''$ is compared with the original individual x_i. If the fitness of $x_i{}''$ is better than x_i, then $x_i{}''$ is selected as the next generation; otherwise, the old value x_i is retained.

A.3 ES

In this paper, (μ,λ) ES [2] is adopted. Compared to CEP, recombination operation is applied to ES, and the selection operation has a discrepancy.

a. Recombination:

$$x_i{}' = \frac{(x_j + x_k)}{2}, j \neq k \tag{17}$$

b. Mutation:

$$x_i{}''(j) = x_i{}'(j) + \eta_i(j) N_j(0,1) \tag{18}$$

Table 2: Benchmark Functions

Function Name	Definition	Properties
The Ackley's Function	$F_{ackley}(x) = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi x_i)) + 20$	Multimodal Separable
The Elliptic Function	$F_{elliptic}(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$	Unimodal Separable
The Rastrigin's Function	$F_{rastrigin}(x) = \sum_{i=1}^{D} [x_i^2 - 10cos(2\pi x_i) + 10]$	Multimodal Separable
Rosenbrock's Function	$F_{rosenbrock}(x) = \sum_{i=1}^{D-1} [100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$	Multimodal Fully-nonseparable
Schwefel's Problem 1.2	$F_{schwefel}(x) = \sum_{i=1}^{D} \sum_{j=1}^{i} {x_i}^2$	Unimodal Fully-nonseparable
The Sphere Function	$F_{sphere}(x) = \sum_{i=1}^{D} x_i^2$	Unimodal Separable

$$\eta_i(j) = \eta_i(j)exp(\tau' N(0,1) + \tau N_j(0,1)) \tag{19}$$

The parameters in the above equations denote the same thing as in CEP.

c. Selection: (μ,λ) selection. That is, μ parents are allowed to breed λ offspring, then select the best μ individuals from the λ individuals.

A.4 GA

GA [7] is a typical evolutionary algorithm. In GA, the individuals participated in the process of recombination and mutation are chosen from parental population.

a. Selection: roulette wheel selection.

b. Recombination: randomly generated a number between 0 and 1, named α, then

$$x_i{}' = \alpha \cdot x_{j1} + (1-\alpha)x_{j2} \tag{20}$$

c. Mutation: Gaussian mutation.

In Gaussian mutation, the original genetic values are replaced by random values which subject to normal distribution, and these random values can be derived from uniform distributed values according to equation (21).

$$Q = \mu + \sigma \cdot (\sum_{i=1}^{12} rand(i) - 6) \tag{21}$$

Where $rand(i)$ means the random value subject to uniform distribution and Q represents the random value subjects to normal distribution $N(\mu,\sigma^2)$. If the range of the genetic value is $[lb, ub]$, then μ and σ can be replaced by:

$$\mu = \frac{lb + ub}{2} \tag{22}$$

$$\sigma = \frac{ub - lb}{2} \tag{23}$$

Consequently, the new genetic value $x_k{}'$ can be written as below:

$$x_k^{'} = \frac{lb + ub}{2} + \frac{ub - lb}{6} \cdot (\sum_{i=1}^{12} rand(i) - 6) \tag{24}$$

B. BENCHMARK FUNCTIONS

As shown in Table 2, six benchmark functions [9] adopted in this paper are the classical benchmark functions in the area of continuous optimization, and their properties represent several different landscapes. The dimensionality of all the functions are set to 30. And the global optimum is the solution that has the minimum function value.