When Optimization Is Just an Illusion

Muhammad Marwan Muhammad Fuad

Forskningsparken 3, Institutt for kjemi, NorStruct
The University of Tromsø, The Arctic University of Norway
NO-9037 Tromsø, Norway
mfu008@post.uit.no

Abstract. Bio-inspired optimization algorithms have been successfully applied to solve many problems in engineering, science, and economics. In computer science bio-inspired optimization has different applications in different domains such as software engineering, networks, data mining, and many others. However, some applications may not be appropriate or even correct. In this paper we study this phenomenon through a particular method which applies the genetic algorithms on a time series classification task to set the weights of the similarity measures used in a combination that is used to classify the time series. The weights are supposed to be obtained by applying an optimization process that gives optimal classification accuracy. We show in this work, through examples, discussions, remarks, explanations, and experiments, that the aforementioned method of optimization is not correct and that completely randomly-chosen weights for the similarity measures can give the same classification accuracy.

Keywords: Bio-inspired Optimization, Genetic Algorithms, Similarity Measures, Time Series Data Mining.

1 Introduction

Optimization is a ubiquitous problem that has a broad range of applications in engineering, economics, and others. In computer science optimization has different applications in software engineering, networking, data mining and other domains. Optimization can be defined as the action of finding the best-suited solution of a problem subject to given constraints. These constraints can be in the boundaries of the parameters controlling the optimization problem, or in the function to be optimized. Optimization problems can be classified according to whether they are: discrete/ continuous/hybrid, constrained/unconstrained, single objective/multiobjective, unimodal (one extreme point) /multimodal (several extreme points).

Formally, an optimization task can be defined as follows: Let $\vec{X} = [x_1, x_2, \ldots, x_{nbp}]$ be the candidate solution to the problem for which we are searching an optimal solution. Given a function $f : U \subseteq \mathbf{R}^{nbp} \to \mathbf{R}$ (nbp is the number of parameters), find the solution $\overrightarrow{X^*} = [x_1^*, x_2^*, \ldots, x_{nbp}^*]$ which satisfies $f\left(\overrightarrow{X^*}\right) \le f\left(\vec{X}\right), \forall \vec{X} \in U$. The function f is called the fitness function, the objective function, or the cost function.

While fitness functions can sometimes be expressed analytically using mathematical formulas, in many cases there is no mathematical formula to express this function.

Optimization algorithms can be classified in several ways, one of which is whether they are *single solution –based* algorithms; these use one solution and modify it to get the best solution. The other category is *population-based* algorithms; these use several solutions which exchange information to get the best solution.

Optimization problems can be handled using deterministic algorithms or probabilistic ones. *Metaheuristics* are general approximate optimization algorithms which are applicable to a wide range of optimization problems. Metaheuristics are usually applied when the search space is so large, or when the number of parameters of the optimization problem is very high, or, and this is particularly important, when the relationship between the fitness function and the parameters is not clear.

There are several paradigms to handle optimization problems, one of which is *bio-inspired*, also called *nature-inspired*, optimization algorithms. These optimization algorithms are inspired by natural phenomena or by the collective intelligence of natural agents.

Bio-inspired computation can be classified into two main families; the first is *Evolutionary Algorithms* (EA). This family is probably the largest family of bio-inspired algorithms. EA are population-based algorithms that use the mechanisms of Darwinian evolution such as selection, crossover and mutation.

The *Genetic Algorithm* (GA) is the main member of EA. GA is an optimization and search technique based on the principles of genetics and natural selection [8]. GA has the following elements: a population of individuals, selection according to fitness, crossover to produce new offspring, and random mutation of new offspring [11].

In the following we present a description of the simple, classical GA. The first step of GA is defining the problem variables and the fitness function. A particular configuration of variables produces a certain value of the fitness function and the objective of GA is to find the configuration that gives the "best" value of the fitness function. GA starts with a collection of individuals, also called *chromosomes*, each of which represents a possible solution to the problem at hand. This collection of randomly chosen chromosomes constitutes a population whose size *popSize* is chosen by the algorithm designer. This step is called *initialization*. In real-valued encoding GA a candidate solution is represented as a real-valued vector in which the dimension of the chromosomes is equal to the dimension of the solution vectors [1]. This dimension is denoted by *nbp*. The fitness function of each chromosome is evaluated. The next step is *selection*, which determines which chromosomes are fit enough to survive and possibly produce offspring. This is decided according to the fitness function of the chromosome. The percentage of chromosomes selected for mating is denoted by *sRate*. *Crossover* is the next step in which offspring of two parents are produced to enrich the population with fitter chromosomes. *Mutation*, which is a random alteration of a certain percentage *mRate* of chromosomes, is the other mechanism which enables the GA to examine unexplored regions in the search space.

Now that a new generation is formed, the fitting function of the offspring is calculated and the above procedures repeat for a number of generations *nGen* or until a stopping criterion terminates the algorithm. □

Data mining is a branch of computer science that handles several tasks, most of which demand extensive computing. As with other fields of research, different papers have proposed applying bio-inspired optimization to data mining tasks [12], [13], [14], [15], [16]. Most of these tasks, however, include non-analytical fitness functions and the relationship between the parameters and fitness function is vague. Under certain circumstances this may result in "pseudo optimization", that is; an optimization algorithm that seems to be functioning normally, but the outcome is unintuitive, or not different from what a completely random choice of values for the parameters can yield.

We will show in this paper the consequences of an inappropriate application of bio-inspired optimization on a particular data mining problem, and we try to explain the reasons for the unintuitive results obtained. In Section 2 we present this application we are referring to, in Section 3 we give our remarks on this application, we show through a counter example in Section 4 that the results of the application, which are supposed to be the outcome of an optimization process, can be obtained through a random solution of the problem, and we try to explain in Section 5 why the application we are referring to did not work, we conclude the paper in Section 6.

2 On Using the Genetic Algorithms to Combine Similarity Measures of Time Series in a Classification Task

A *time series* is a collection of observations at intervals of time points. These observations are measurements of a particular phenomenon. Formally, an n-dimensional time series S is an ordered collection:

$$S = \{(t_1, v_1), (t_2, v_2), \ldots, (t_n, v_n)\} \quad (1)$$

where $t_1 < t_2 < \ldots < t_n$, and where v_i are the values of the observed phenomenon at time points t_i.

Time series data mining handles several tasks such as classification, clustering, similarity search, motif discovery, anomaly detection, and others.

The goal of classification, one of the main tasks of data mining, is to assign an unknown object to one out of a given number of classes, or categories [10]. One of the most popular classification techniques of time series is *Nearest-Neighbor Classification* (NNC). In NNC the time series-query is classified according to the majority of its nearest neighbors [10].

A Classification task is based on another, fundamental task of data mining, which is the *similarity search* problem. In this problem a pattern or a *query* is given and the similarity search task is to retrieve the data objects in the database that are "close" to that query according to some semantics that quantify this closeness. This closeness or similarity is quantified using a principal concept which is the *similarity measure* or its more rigorous concept; the *distance metric*.

There are many similarity measures or distance metrics (in this paper we will use these two terms interchangeably) in the field of time series data mining. The state-of-the-art distance measures [5] include the *Minkowski distance* (L_p) mainly the *Euclidian distance* (L_2) and the *Manhattan distance* (L_1), *Dynamic Time Warping* (DTW), the *Longest Common Subsequence* (LCSS), the *Edit Distance with Real Penalty* (ERP), the *Edit Distance on Real Sequences* (EDR), *Dissimilarity Distance* (DISSIM), *Similarity Search based on Threshold Queries* (TQ), *Spatial Assembling Distance* (SpADe), and *Sequence Weighted Alignment* (Swale).

In [2] the authors propose utilizing a similarity function defined as a weighted combination of several metrics. A similar idea was proposed in [3] where the authors present a retrieval method based on a weighted combination of feature vectors.

The method we are discussing in this work [6] is called “Combination of Similarity Measures for Time Series Classification Using Genetic Algorithms”. We will refer to this method from now on as CSM-GA. The authors of CSM-GA propose the same idea of a weighted combination of similarity measures. As we can see, the idea is not new, although they describe their method as “novel”. They mention however that the closest work to theirs is that of [18], which also uses a weighted combination of distances on a text classification task, and where the optimized weights are also computed by applying the genetic algorithms, so we are not sure what the novelty of CSM-GA is.

It is important to mention here that in our paper we are not going to discuss the validity of [18], since text classification is not our field of expertise (besides the application details are not presented in [18]), neither will we discuss the similarity measures proposed in that paper. We will not discuss either the correctness of applying the genetic algorithms, or any other optimization technique, to determine the weights of the similarity measures in a combination of similarity measures as a method to enhance the retrieval process in any kind of multimedia search in general. We are only discussing the validity of CSM-GA as described in [6].

As mentioned earlier, the idea of CSM-GA is to use a weighted combination of similarity measures for time series, where the weights are determined using the genetic algorithms (GA), and the authors of CSM-GA apply it on a classification task. This is expressed mathematically as:

$$S_{new} = \sum_{i=1}^{n} \omega_i . s_i \tag{2}$$

where S_{new} is the new similarity measure, $(s_1, s_2, \ldots, s_n)$ are the n combined similarity measures, ω_i are the associated weights, and where $0 \leq \omega_i \leq 1$.

CSM-GA uses a classical GA (described in Section 1) as an optimization technique. The fitness function to be optimized is the classification accuracy.

As indicated earlier, the objective of CSM-GA is to find the optimal values of ω_i which yield the highest classification accuracy.

The authors of CSM-GA use 8 similarity measures in the combination of similarity measures of relation (2). The similarity measures used in CSM-GA are the following:

i- Euclidean Distance (L_2): defined between time series *p* and *q* as:

$$s_1(p,q)=\sqrt{\sum_{i=1}^{n}(p_i-q_i)^2}$$

ii- Manhattan Distance (L_1): defined as:

$$s_2(p,q)=\left|\sum_{i=1}^{n}(p_i-q_i)\right|$$

iii- Maximum Distance (L_∞):

$$s_3(p,q)=max(|p_1-q_1|,|p_2-q_2|,\ldots,|p_n-q_n|)$$

iv- Mean Dissimilarity:

$$s_4(p,q)=\frac{1}{n}\sum_{i=1}^{n}disim(p_i,q_i),\qquad \text{where: } disim(p_i,q_i)=\frac{|p_i-q_i|}{|p_i|-|q_i|}$$

v- Root Mean Square Dissimilarity:

$$s_5(p,q)=\sqrt{\frac{1}{n}\sum_{i=1}^{n}disim(p_i,q_i)^2}$$

vi- Peak Dissimilarity:

$$s_6(p,q)=\frac{1}{n}\sum_{i=1}^{n}peakdisim(p_i,q_i),\qquad \text{where: } peakdisim(p_i,q_i)=\frac{|p_i-q_i|}{2\,max(|p_i|,|q_i|)}$$

vii- Cosine Distance:

$$s_7(p,q)=1-cos(\theta),\qquad \text{where: } cos(\theta)=\frac{p.q}{\|p\|\cdot\|q\|}$$

viii- Dynamic Time Warping Distance (DTW): DTW is an algorithm to find the optimal path through a matrix of points representing possible time alignments between two time series $p=\{p_1, p_2,\ldots, p_n\}$ and $q=\{q_1, q_2,\ldots, q_m\}$. The optimal alignment can be efficiently calculated via dynamic programming [7]. The dynamic time warping between the two time series is defined as:

$$s_8=DTW(i,j)=d(i,j)+min\begin{cases}DTW(i,j-1)\\DTW(i-1,j)\\DTW(i-1,j-1)\end{cases}\quad \text{, where } 1\le i\le n,\ 1\le j\le m \qquad \square$$

The authors of CSM-GA test their method on a *1*-NN classification task of time series data. In *k*-NN each time series is assigned a class label. Then a "leave one out" prediction mechanism is applied to each time series in turn; i.e. the class label of the chosen time series is predicted to be the class label of its nearest neighbor, defined based on the tested distance function. If the prediction is correct, then it is a hit; otherwise, it is a miss.

Table 1. The datasets used in the experiments of CSM-GA

Dataset	Size of Training Set	Size of Validation Set	Size of Test Set
Control Chart	180	120	300
Coffee	18	10	28
Beef	18	12	30
OliveOil	18	12	30
Lightning2	40	20	61
Lightning7	43	27	73
Trace	62	38	100
ECG200	67	33	100

The classification error rate is defined as the ratio of the number of misses to the total number of time series [4]. *1*-NN is a special case of *k*-NN, where *k=1*.

The data sets chosen in the experimental part of CSM-GA are obtained from [9]. This archive contains 47 different datasets, where each dataset is divided into a training set, on which the algorithm for the problem at hand is trained, and a testing set, to which the outcome of the training set is applied [17]. To test CSM-GA the authors apply a different protocol where they split the original training datasets in [9] further into what they call *training set* and *validation set* (in other words, training set + validation set in CSM-GA = training set in [9]), so they have three sets in their experiments as they report in Table 1 taken from their paper (we omit two columns which show the number of classes in the datasets and the size of the time series).

The authors of CSM-GA do not explain the reason for choosing this protocol, neither do they explain the exact role of their training set and validation set. However, the authors report the results they obtain after running the genetic algorithms for 10 generations on the datasets presented in Table 1 to obtain the optimal weights ω_i of the distance measures s_i in relation (2). We show these optimal weights in Table 2 (which is also taken from their paper).

Then the weights in Table 2 are combined to form the new similarity measure which, as the authors explain in their paper, is used to classify the test data. The results are shown in Table 3, which also contains the results of applying each similarity measure separately, also on a *1*-NN classification task, which the authors report for comparison reasons to show the merits of using a combination of similarity measures. We present Table 3 exactly as it was presented in CSM-GA.

Table 2. Weights assigned to each similarity measure after 10 generations of CSM-GA

Dataset	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8
Control Chart	0.72	0.29	0.33	0.18	0.12	0.61	0.31	0.82
Coffee	0.74	0.9	0.9	0.1	0.03	0.03	0.06	0.70
Beef	0.95	0.09	0	0.48	0	0.62	0.58	0.73
OliveOil	0.7	0	0.79	0	0	0	0.58	0.67
Lightning2	0.95	0.38	0	0.78	0.42	0.81	0.49	0.59
Lightning7	0	0	0	0.23	0.84	0.56	0	0.39
Trace	0.62	0.08	0.28	0.39	0.14	0.47	0.23	0.98
ECG200	0.052	0	0.21	0	0	0.98	0.90	0

Table 3. Comparison of classification accuracy using CSM-GA and other similarity measures

Dataset	Size	CSM-GA mean ± std	L_2	L_1	L_∞	disim	root disim	peak disim	cosine	DTW
Control Chart	600	99.07± 0.37	88	88	81.33	58	53	77	80.67	99.33
Coffee	56	87.50± 2.06	75	53.57	89.28	75	75	75	53.57	82.14
Beef	60	54.45± 1.92	53.33	50	53.33	46.67	50	46.67	20	50
OliveOil	121	82.67± 4.37	86.67	80.33	83.33	63.33	60	63.33	16.67	86.67
Lightning2	121	87.54± 1.47	74.2	81.9	68.85	55.75	50.81	83.60	63.93	85.25
Lightning7	143	69.28± 2.97	71.23	71.23	45.21	34.24	28.76	61.64	53.42	72.6
Trace	200	100.0± 0.00	76	100	69	65	57	75	53	100
ECG200	200	90.00± 1.15	88	89	87	79	79	91	81	77

The authors of CSM-GA do not explain how they obtained the standard deviation. The size of the data sets in Table 3, which is the sum of the training sets, the validation sets, and the test sets, would suggest that they tested the weights on the training sets+ validation sets+ testing sets all mixed together.

The authors of CSM-GA conclude that the results obtained by their approach are "considerably better" and that their method "is guaranteed to yield better results".

3 Legitimate Remarks

In the following we present a few remarks on CSM-GA as proposed and implemented in [6]:

1- Some of the results presented in the experiments of CSM-GA are very unintuitive; e.g. L_∞ gave better classification accuracy on (Coffee) than the combination itself. This means that after going through the costly optimization process, CSM-GA proposed an output which is not even as good as using a single similarity measure (whose complexity is low) and which does not require any optimization process. This was also the case with DTW on (Chart Control), L_1 , L_∞, and DTW on (OliveOil) , L_1 , L_2 and DTW on (Lightning7), L_1 and DTW on (Trace), *peakdism* on (ECG200). In fact for all the datasets reported in Table 3 we see that one or even several similarity measures give very close, or even better, classification accuracy than the combination itself, which requires a costly optimization process. To give an idea, the total time of training the two smallest datasets in Table 1 for 10 generations (run on Intel Core 2 Duo CPU with 3G memory) was 29 hours 43 minutes (Beef) and 6 hours 23 minutes (Coffee), yet the datasets used in the experiments of CSM-GA are among the smallest datasets in [9]. As we can see, the optimization process is very time consuming, may be even impossible to implement for some datasets in [9] whose size is of the magnitude of 1800 (for comparison, the sizes of Beef and Coffee are 30 and 28, respectively). Yet this optimization process proposed in CSM-GA is of very little benefit, if any. These surprising remarks make the motivation of applying CSM-GA seriously questionable.

2- The choice of some of the similarity measures in the experimental part of CSM-GA seems redundant. Some of these distances are of the same family (L_1, L_2, L_∞), and (*dism*, *rootdisim*). They are even related numerically ($L_1 \geq L_2 \geq L_\infty$), (*dism* $\leq$, *rootdisim*). Strange enough, although the authors present in the background of their paper the same similarity measures we mentioned in Section 2 as the state-of-the-art similarity measures, they use only 3 of these measures in their experiments.

3- The outcome of the training phase in some cases is strange and unexplainable; for instance, Table 3 shows that the weight of s_3 on dataset (Beef) is 0, which means that this similarity measure is completely useless in classifying (Beef), yet Table 3 shows that the performance of s_3 as a standalone similarity measure on (Beef) is not worse than any other similarity measure of the others (on the contrary, it seems to be the best). This is also the case with s_2, s_4, s_5, s_6 on (OliveOil) and s_1, s_2, s_3, s_7 on (Lighting7) , and this was particularly strange with s_2 on (Trace) whose value on the training set was very small; 0.08 whereas its performance as a standalone similarity measure is completely accurate (100%). In fact, this result itself refutes the basis of the whole method CSM-GA; if s_2 (which is L_1) gives and accuracy of 100% on (Trace) then why do we have to go through all this long and complicated optimization process to get the same results (which are not even intuitive since the solution obtained gives a very small weight to this distance, almost 0)?

4- The most astonishing remark to us was the abnormal existence of all these zeros in Table 2, and in 10 generations only. This point needs particular attention; in genetic algorithms the only possibility to produce a certain value is either during initialization or through mutation (or a mixture of both). As for initialization, CSM-GA uses a population of 10 chromosomes each with 8 parameters (the similarity measures) we find it very hard to understand how a random generator of real numbers between 0 and 1 (with 2 or 3 decimal digits) would generate the value zero 15 times out of 80 (=10x8) . As for mutation, again, this is also a mystery; the authors of CSM-GA do not state the mutation rate they used, they only mention that it is small. However, most genetic algorithms use a mutation rate of 0.2 [8], so the number of mutations of CSM-GA is 10 (=population size) x 8 (=number of parameters) x 10 (=number of generations) x 0.2 (=mutation rat) =160 random real numbers between 0 and 1 (with 2 or 3 decimal digits). Again we can not understand how 0 was generated 15 times. This remark strongly suggests that there was an error in the optimization process of CSM-GA.

4 A Counter Example

In this section we will show that the conclusion of CSM-GA; that there is an optimal combination of weighted similarity measures which gives an optimal classification accuracy of a time series *1*-NN classification task, is unfounded, and that any, completely randomly-chosen combination of the weighted similarity measures presented in CSM-GA on the same datasets can give very close classification accuracy.

In our example we will conduct a *1*-NN classification task on the same time series presented in Section 2, using the same datasets, and the same similarity measures, except that the weights of the similarity measures we use are chosen randomly and without any optimization process.

To prove that the weights we choose are random and do not result from any hidden optimization process we choose weights resulting from a known irrational number; the number π=3.141592653589793238462…. Since in the experiments of CSM-GA there were 8 weights, each between 0 and 1, with two digits after the decimal point, we will remove the decimal point of π and assign each two successive digits, after dividing them by 100 (to scale them in the range [0, 1]; the range of ω_i), to the successive similarity measures, so the weights will be: ω_1=0.31, ω_2=0.41, ω_3=0.59, ω_4=0.26, ω_5=0.53, ω_6=0.58, ω_7=0.97, ω_8=0.93. As we can see, the weights do not result from any optimization process whatsoever. Then we construct a combination of the similarity measures according to relation (2) with the above weights, so we get:

$$S_\pi = 0.31s_1 + 0.41s_2 + 0.59s_3 + 0.26s_4 + 0.53s_5 + 0.58s_6 + 0.97s_7 + 0.93s_8$$

We also wanted to test another randomly chosen number, so we took the square root of 2 ; $\sqrt{2} = 1.4142135623730950488...$, which is also an irrational number, and we proceeded in the same manner as we did with π, i.e.: we remove the decimal point and we assign each two successive digits, after dividing them by 100, to the successive similarity measures in relation (2), so we get the following weights in this case; ω_1=0.14, ω_2=0.14, ω_3=0.21, ω_4=0.35, ω_5=0.62, ω_6=0.37, ω_7=0.30, ω_8=0.95 , and the combination in this case is:

$$S_{\sqrt{2}} = 0.14s_1 + 0.14s_2 + 0.21s_3 + 0.35s_4 + 0.62s_5 + 0.37s_6 + 0.30s_7 + 0.95s_8$$

Now we test S_π and $S_{\sqrt{2}}$ on the same datasets on which CSM-GA was tested. Since in our experiment we are not applying any optimization process we do not have any training phase (which is used in CSM-GA to get the optimal weights), so we apply S_π and $S_{\sqrt{2}}$ directly on the test sets. In Table 4 we present the results we obtained (together with those of CSM-GA for comparison). As we can see, the results are very close in general and we do not see any significant difference in classification accuracy between CSM-GA, which is supposed to be the outcome of an optimization process, and two other combinations with weights chosen completely at random. In other words, any combination of weights could produce the same classification accuracy. We experimented with other randomly chosen combinations of weights and we got similar results, and we urge the reader to test any combination of weights (between 0 and 1) and he/she will draw the same conclusion that we did that there is no optimal combination of weights.

Table 4. Comparison of the classification accuracy using CSM-GA and two randomly chosen combinations of weights w_i

Dataset	**CSM-GA** **mean ± std**	S_π	$S_{\sqrt{2}}$
Control Chart	99.07± 0.37	96	98.33
Coffee	87.50± 2.06	92.85	89.28
Beef	54.45± 1.92	56.66	53.33
OliveOil	82.67± 4.37	83.33	80
Lightning2	87.54± 1.47	83.61	88.53
Lightning7	69.28± 2.97	80.82	76.71
Trace	100.0± 0.00	96	95
ECG200	90.00± 1.15	90	90

It is also worth mentioning that [9] urges all those who use that archive to test on all the datasets to avoid "cherry picking" (presenting results that work well/ better with certain datasets). The authors of CSM-GA do not mention why they did not test their method on the other datasets in [9], or why they chose these datasets in particular. However, in this example we meant to test our random combinations on these datasets in particular because these are the ones on which CSM-GA was tested.

It is very important to mention here that all the results, remarks and the example we present in this paper are reproducible; the datasets are, as mentioned before, available at [9] where the reader can also find the code for the classification task. As for the 8 similarity measures used (Section 2) they are very easy to code. Besides, many scientific programming forums on the internet have the codes for these similarity measures, so all our results are easily verifiable.

5 What Went Wrong with CSM-GA

In this section we try to answer the question of why CSM-GA did not work. We were able to find two major errors in CSM-GA

1-The most serious error in CSM-GA is that the similarity measures it uses were not normalized before being used in relation (2). These similarity measures have very different numeric values so they should be normalized. To give a rough estimation, we present in Table 5 the distances between the first and the second time series of the test datasets used in the experiments of CSM-GA using each of the 8 similarity measures used with CSM-GA. We believe Table 5 clearly explains the whole point; since all the weights w_i are in the interval [0,1] then for a certain value of the weight the combination in relation (2) will be dominated by the similarity measures with the highest numeric values. We see in (Coffee) for instance that the value of DTW is $6.3\text{x}10^6$ times larger than that of the Cosine Distance, so the influence of DTW on S_{new} in (2) will be far much stronger than that of the Cosine Distance in classifying (Coffee), so normalization is crucial.

Table 5. A rough estimation of the numeric values of the similarity measures used in CSM-GA

Dataset	L_2	L_1	L_∞	disim	root disim	peak disim	cosine	DTW
Control Chart	9.79	59.17	2.94	0.64	0.8	0.48	0.81	21.16
Coffee	187.58	2932.7	18.77	0.23	0.48	0.19	0.001	10567
Beef	0.93	13.8	0.14	0.53	0.73	0.38	1.05	50
OliveOil	0.07	0.84	0.02	0.007	0.08	0.007	0.000	0.004
Lightning2	32.85	358.24	18.36	0.75	0.87	0.50	0.85	294.06
Lightning7	16.55	206.18	8.88	0.59	0.77	0.43	0.43	98.28
Trace	27.57	389.97	5.08	0.76	0.87	0.56	1.39	437.4
ECG200	3.90	28.48	1.53	0.36	0.60	0.26	0.08	6.34

Since the numeric values of L_1, L_2, L_∞, DTW are far much larger than those of the other 4 similarity measures, and since, as we can see from Table 3, the performance of L_1, L_2, L_∞, DTW as standalone similarity measures is better than the other 4 similarity measures, we can easily understand the reason behind what it seems as the "optimal" classification accuracy of S_{new}. It is simply due to the fact that S_{new} is dominated by similarity measures whose performance is superior, and not as a result of an optimization process. This point explains the findings of our example in Section 4 that any combination will likely give similar classification accuracy as that of CSM-GA.

2- The *1*-NN classification task of time series has certain characteristics related to its nature. We will explain this through the following example; the size of (Beef) is 30, thus the number of mismatches on this dataset (whatever algorithm is used) will be *NrMis*= {0,1,2,…,30}, thus the *1*-NN classification accuracy (which is the objective function of CSM-GA), whatever algorithm is used, will take one of these values: *ClassAcc*={30/30,29/30,28/30,…,0/30}. This set *ClassAcc* is finite (and even more it has a small cardinality), while the values of the weights w_i belong to an infinite set (the interval [0, 1]), yet any combination of weights of similarity measures will yield *1*-NN classification accuracy that belongs to the finite (and small) set *ClassAcc*, this means that an infinite number of different combinations will give the same classification accuracy. In optimization language, such an objective function is highly multimodal, so the assumption that there is an optimal combination of similarity measures that gives maximum*1*-NN classification accuracy is incorrect and the search for such an optimal combination is meaningless (and the example we presented in Section 4 leads to the same conclusion).

6 Conclusion

There are far too many "new" bio-inspired optimization applications claiming their superior performance. But a deep investigation of these techniques may show that they are incorrect. Our purpose in this paper was by no means to refute a particular method or to show its incorrectness, but it was rather to shed light on certain cases where the careless, inappropriate, or erroneous application of bio-inspired algorithms may give false and even unintuitive results. While the widespread success of bio-inspired algorithms encourages many researchers to apply these algorithms extensively, it is important to be careful to choose the right application that takes into account the problem at hand, in addition to the correct implementation of the bio-inspired algorithm, especially that these algorithms require employing intensive computing resources, so we have to make sure that the results obtained will be better than those yielded by a random solution of the problem.

References

1. Affenzeller, M., Winkler, S., Wagner, S., Beham, A.: Genetic Algorithms and Genetic Programming Modern Concepts and Practical Applications. Chapman and Hall/CRC (2009)
2. Bustos, B., Skopal, T.: Dynamic Similarity Search in Multi-metric Spaces. In: Proceedings of the ACM Multimedia, MIR Workshop, pp. 137–146. ACM Press, New York (2006)

3. Bustos, B., Keim, D.A., Saupe, D., Schreck, T., Vrani'c, D.: Automatic Selection and Combination of Descriptors for Effective 3D Similarity Search. In: Proceedings of the IEEE International Workshop on Multimedia Content-based Analysis and Retrieval, pp. 514–521. IEEE Computer Society (2004)
4. Chen, L., Ng, R.: On the Marriage of Lp-Norm and Edit Distance. In: Proceedings of 30th International Conference on Very Large Data Base, Toronto, Canada (August 2004)
5. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures. In: Proc of the 34th VLDB (2008)
6. Dohare, D., Devi, V.S.: Combination of Similarity Measures for Time Series Classification using Genetic Algorithms. Congress on Evolutionary Computation, CEC (2011)
7. Guo, A.Y., Siegelmann, H.: Time-warped Longest Common Subsequence Algorithm for Music Retrieval. In: Proc. ISMIR (2004)
8. Haupt, R.L., Haupt, S.E.: Practical Genetic Algorithms with CD-ROM. Wiley-Interscience (2004)
9. Keogh, E., Zhu, Q., Hu, B.: Hao. Y., Xi, X., Wei, L. and Ratanamahatana, C.A.: The UCR Time Series Classification/Clustering, `http://www.cs.ucr.edu/~eamonn/time_series_data/`
10. Last, M., Kandel, A., Bunke, H. (eds.): Data Mining in Time Series Databases. World Scientific (2004)
11. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1996)
12. Muhammad Fuad, M.M.: ABC-SG: A New Artificial Bee Colony Algorithm-Based Distance of Sequential Data Using Sigma Grams. In: The Tenth Australasian Data Mining Conference, AusDM 2012, Sydney, Australia, December 5-7 (2012)
13. Muhammad Fuad, M.M.: Genetic Algorithms-Based Symbolic Aggregate Approximation. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2012. LNCS, vol. 7448, pp. 105–116. Springer, Heidelberg (2012)
14. Muhammad Fuad, M.M.: Particle Swarm Optimization of Information-Content Weighting of Symbolic Aggregate Approximation. In: Zhou, S., Zhang, S., Karypis, G. (eds.) ADMA 2012. LNCS (LNAI), vol. 7713, pp. 443–455. Springer, Heidelberg (2012)
15. Muhammad Fuad, M.M.: Towards Normalizing the Edit Distance Using a Genetic Algorithms–Based Scheme. In: Zhou, S., Zhang, S., Karypis, G. (eds.) ADMA 2012. LNCS (LNAI), vol. 7713, pp. 477–487. Springer, Heidelberg (2012)
16. Muhammad Fuad, M.M.: Using Differential Evolution to Set Weights to Segments with Different Information Content in the Piecewise Aggregate Approximation. In: KES 2012. Frontiers of Artificial Intelligence and Applications (FAIA). IOS Press (2012)
17. Xiao, Y., Chen, D.Y., Ye, X.L.: Hu: Entropy-Based Symbolic Representation for Time Series Classification. Fuzzy Systems and Knowledge Discovery. In: Fourth International Conference on In Fuzzy Systems and Knowledge Discovery (2007)
18. Yamada, T., Yamashita, K., Ishii, N., Iwata, K.: Text Classification by Combining Different Distance Functions with Weights. In: International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, and International Workshop on Self-Assembling Wireless Networks (2006)