Towards the Full Model Selection in Temporal Databases by Using Micro-Differential Evolution.
An Empirical Study

Nancy Pérez-Castro, Héctor-Gabriel Acosta-Mesa, Efrén Mezura-Montes, Nicandro Cruz-Ramírez
Artificial Intelligence Research Center
University of Veracruz
Sebastián Camacho # 5, Centro, Xalapa, Veracruz, 91000, MÉXICO.
perez.castro.nancy@gmail.com, heacosta@uv.mx, emezura@uv.mx, ncruz@uv.mx

Abstract—In this paper, an empirical comparison of four micro differential evolution variants to solve partial instances of the Full Model Selection (FMS) problem on time-series databases is presented. Two smoothing methods, three time-series representations, and one classifier, all of them with their corresponding parameters, are optimized by the evolutionary search. The study focuses on the effects of the crossover operator and the type of base vector used in the differential mutation. The four variants are tested in three experiments to analyze the final statistical results, the convergence behavior and the type of optimized model obtained. Six temporal databases with different features are solved. The results, which are statistically validated by using non-parametric tests, suggest that the four variants are able to provide competitive results. However, using the best vector in the current population, coupled with a binomial crossover, allow the micro differential evolution to reach better final results, while the opposite (base vector chosen at random and exponential crossover) help to find a good model with less solution evaluations in temporal databases whose models are very expensive to evaluate.

I. INTRODUCTION

Nowadays, the concepts of data mining and modelling data are considered as hot topics and are under fast development [1]. Many practitioners and academics are attracted to use the methods involved in those areas because of the wide range of applications. In several areas (e.g. science, industry, business and others), a huge amount of data is generated and stored in databases, which contain, explicitly or implicitly, temporal information [2]. Time-series, in particular, are a class of temporal data which have been treated in the field of data mining [3], [4]. Time-series can be defined as a collection of chronologically-made observations, which is characterized by its numerical and continuous nature [3]. In a time-series database, each time-series is always considered as a whole, unlike traditional databases (known as snapshot databases) which are usually discrete and each field is considered as an individual value [5]. The specialized literature reports that traditional techniques for data mining may perform poorly in presence of temporal data [2]. Such performance may be due to the fact that most of them tend to treat temporal data as an unordered collection of events, ignoring its temporal information. Thus, new techniques are proposed to deal temporal data in different tasks such as query by content, anomaly detection, motif discovery, clustering, prediction, and classification, being the latter which has recently attracted an important interest [6]. Practitioners in temporal data mining and other disciplines have faced the dilemma to choose an appropriate technique to solve a specific instance (database). This dilemma has been named as the algorithm selection problem [7] or full model selection problem (FMS) [8]. This problem emerged from the machine learning community and has focused mainly on choosing a suitable learning algorithm for classification problems. However it has been generalized to other disciplines [9]. As a result of the literature review, it was observed that the research efforts are focused on two main topics: (1) FMS for temporal databases for forecasting, where parameter optimization is not considered, and (2) FMS for snapshot databases where stochastic optimization algorithms have been adopted and where traditional data mining techniques are employed. Regarding the first topic, the works are focused on finding an appropriate forecasting model for time-series databases through induction rules [10], intelligent decision-support system based on neural network [11] or by using classifiers as J48 [12] to select suitable forecasting models. For the second topic, there are works based on Genetic Algorithms (GA) [13] and also on swarm intelligence algorithms (SIAs) like Particle Swarm Optimization [8]. In these cases the exploration of different groups of traditional data mining techniques is carried out. Furthermore, the optimization of the parameters related to each method was considered. Differential Evolution (DE) is part of the Evolutionary Algorithms (EAs), whose performance on solving optimization problems has been remarkable [14]. DE, as other EAs, is population-based, i.e., a set of solutions, usually randomly generated, are chosen for reproduction based on fitness during a certain number of cycles. In contrast, there are EAs whose populations are very small [15]. Those are known as micro evolutionary algorithms and are indicated by the prefix µ [16]. According to Caraffini et. al [16], those algorithms with small populations, are characterized by carrying out more exploitation than the algorithms with larger populations. Thus, the µ-EAs could be able to quickly achieve improvements at early stages of the optimization process. In this way, they might be a valid option in optimization problems where the fitness value is expensive to be calculated, and this is the case for the FMS in temporal databases. From the literature review, and to the best of the authors’ knowledge, there is scarce research about using EAs or SIAs to deal with the FMS in temporal databases by considering proper data mining techniques for time-series and parameter optimization of the methods adopted. Furthermore,
there are no reports on the usage of $\mu$-EAs in such FSM on temporal databases. Motivated by the above mentioned, this work presents an empirical comparison of four $\mu$-DE variants, where noise reduction (i.e. smoothing), time-series representation/reduction and one classification method with different similarity measures are considered. The optimization process is focused on minimizing the misclassification error. The four $\mu$-DE algorithms vary in the type of crossover and in the criterion to choose the base vector in the differential mutation. A set of six representative temporal databases are used in the experiments. In the remaining of this work the term model selection (MS) is adopted instead of FMS, because there are elements of the FMS which are not considered in this work (e.g. post-processing methods) and their inclusion is outside the scope of this paper.

The rest of this paper is organized as follows. Section II describes the main concepts of DE and the $\mu$-DE as well. In Section III, the adapted $\mu$-DE to solve the MS problem in temporal databases and the four variants are detailed. In Section IV, the experiments and the obtained results, statistically validated, are presented and discussed, respectively. Finally, in Section V, the conclusions are shown and the future work is outlined.

II. DIFFERENTIAL EVOLUTION

Differential Evolution (DE), proposed by Storn and Price [17] is a simple yet powerful population-based stochastic search algorithm. DE mainly consists on three operators, i.e., mutation, crossover and selection. Moreover, the scaling factor ($F$), the crossover rate ($CR$) and the population size ($NP$) are three important control parameters involved in DE. According with Storn and Price, the classical DE can be outlined as follows [17]:

**Creating an initial population.** An initial population of vectors $NP_g = \{\vec{x}_{i,1,g}, \vec{x}_{i,2,g}, ..., \vec{x}_{i,D,g}\}$ at generation $g$ is created, where each $\vec{x}_{i,g} = [x_{i,1,g}, x_{i,2,g}, ..., x_{i,D,g}]^T$ has $D$ elements which are randomly determined between predefined lower and upper search limits $L_i \leq x_{i,g} \leq U_i$.

**Mutation Operation.** For each vector $\vec{x}_{i,g}$ in the population (known as target vector), a new vector called mutant vector $\vec{u}_{i,g}$ is generated by Equation 1.

$$\vec{u}_{i,g} = \vec{x}_{r_3,g} + F \cdot (\vec{x}_{r_1,g} - \vec{x}_{r_2,g})$$ (1)

Where three vectors referred as $r_1, r_2$ and $r_3$ are randomly selected from the current population such as $r_1 \neq r_2 \neq r_3 \neq i$. $\vec{x}_{r_3,g}$ is the base vector, whose position defines the starting location to define a search direction. The factor $F > 0$ is a real number which scales the difference vector $(\vec{x}_{r_1,g} - \vec{x}_{r_2,g})$ which precisely defines the search direction starting from the base vector.

**Crossover operation.** After mutation, the trial vector $\vec{u}_{i,g}$ is created by recombining the target vector $\vec{x}_{i,g}$ and its mutant vector $\vec{v}_{i,g}$ through Equation 2.

$$\vec{u}_{i,j,g} = \begin{cases} 
\vec{v}_{i,j,g} & \text{if } rand_j(0,1) \leq CR \
\vec{x}_{i,j,g} & \text{otherwise}
\end{cases} \quad (2)$$

where $j = 1, ..., D, CR \in [0,1]$ is a user-defined value that controls how similar the trial vector will be with respect to the mutant, $rand_j$ generates a random number between 0 and 1 with an uniform distribution and $j_{rand}$ is a randomly chosen integer $\in [1, \ldots, D]$ to ensure no target vector duplicates.

**Selection.** Finally, the target $\vec{x}_{i,g}$ and trial $\vec{u}_{i,g}$ vectors are evaluated and compared based on their fitness values. The vector with the best fitness value is selected for the next generation, see Equation 3, where minimization is assumed.

$$\vec{x}_{i,g+1} = \begin{cases} 
\vec{u}_{i,g} & \text{if } f(\vec{u}_{i,g}) \leq f(\vec{x}_{i,g}) \
\vec{x}_{i,g} & \text{otherwise}
\end{cases} \quad (3)$$

Unlike traditional DE, $\mu$-DE is characterized by using a very small population and a restart mechanism to refresh the population which fastly converges. The structure and operations remain the same as in classical DE. A population of size 6 can be considered a reasonable choice for $\mu$-DE and the restart mechanism requires a number of vectors to be replaced by random ones and also requires the generations interval for the restart mechanism to be applied [15]. Algorithm 1 shows the pseudocode of the $\mu$-DE proposed in [15], which is the base for the contribution of this paper.

**Algorithm 1 $\mu$-DE**

Require: $NP \in [3, 6]$ (Population size), $CR \in [0, 1]$ (Crossover rate), $F > 0$ (Scale factor), $N \in \mathbb{N}$ (Number of restart solutions), $R \in \mathbb{N}$ (Replacement generation).
1. Set $G = MaxEval/NP \%$ Number of generations
2. Set $Count = 1$
3. Randomly generate an initial population of vectors $NP_0 = [\vec{x}_{1,0}, \vec{x}_{2,0}, ..., \vec{x}_{NP,0}]$
4. for $q = 1$ to $G$ do
5. if $Count==R$ then
6. Reinitialization of $N$ worst individuals
7. $Count = 1$
8. end if
9. for $i=1$ to $NP$ do
10. Set $r_1 \neq r_2 \neq r_3, i \in [1, NP], j_{rand} \in [1, D]$ randomly
11. $u_{i,j} = x_{i,j} + F \cdot (x_{r_1,j} - x_{r_2,j})$
12. if ($rand_i(0,1) < CR$ or $j == j_{rand}$) then
13. $\vec{u}_{i,j,g} = x_{r_3,j} + F \cdot (x_{r_1,j} - x_{r_2,j})$
14. else
15. $\vec{u}_{i,j,g} = x_{i,j}$
16. end if
17. end for
18. if ($f(\vec{u}_{i,j,g}) < f(\vec{x}_{i,j,})$) then
19. $\vec{x}_{i,j+1} = \vec{u}_{i,j,g}$
20. else
21. $\vec{x}_{i,j+1} = \vec{x}_{i,j}$
22. end if
23. end for
24. $Count = Count + 1$
25. end for

III. $\mu$-DE MODEL SELECTION ($\mu$-DEMS)

The so-called $\mu$-DEMS, is the $\mu$-DE adapted to solve the MS problem for temporal databases, where the input is a training time-series dataset for classification task, and the output is a model that involves a highly competitive combination of methods for noise reduction, dimensionality reduction and classification. In addition, $\mu$-DEMS chooses the appropriate parameter values for the selected methods. Recalling from Section I, this study is focused on analyzing the effects of the crossover and the base vector in the FMS for temporal databases. Therefore, the four DE variants: rand/1/bin, rand/1/exp, best/1/bin, and best/1/exp (described in Table I), were implemented in $\mu$-DEMS. Unlike the bin crossover, the exp crossover copies values from the mutant vector to the trial
vector until some criteria is satisfied. After that, the remaining values of the trial vector are copied from the target vector. All four DE variants are based on Algorithm 1. The only steps that vary are those of the crossover and the type of base vector used in the mutant vector calculation (steps 12-16 in Algorithm 1), as detailed in Table I.

A. Solution encoding

In μ-DEMS, each solution in the population represents a model that is codified by a vector of eleven variables as in Figure 1. Such figure shows the pool of smoothing, representation and classification methods which are adopted in this work. For each method, its parameters and their respective predefined ranges are outlined in Table II, where some parameters consider the time-series length. The different lengths of time-series are shown in Table III.

![Diagram of solution encoding](image)

Figure 1. Solution encoding.

Decision variable $x_1$ in the solution vector can only take two values (1 or 2) and identifies the smoothing method (ID_SM). Variables $x_2$ to $x_4$ codify the parameters for the selected smoothing method ($x_2$ and $x_3$ belong to ID_SM = 1, while $x_4$ belongs to ID_SM = 2). Variable $x_5$ can take three values (1, 2, and 3) and depicts the identifier of the time-series representation method (ID_RM), while $x_6$, $x_7$, and $x_8$ are the respective parameters for each representation method. $x_6$ is shared between methods with ID_RM = 1 and ID_RM = 2, because it has the same valid range for both of them. Variable $x_9$ represents the classifier selected. However, in this work only the K-Nearest Neighbours (KNN) classifier is considered because it is one of the most popular, simple and competitive methods on time series classification problems whose performance has been remarkable depending on a suitable similarity measure [18]. Four different similarity measures such as Euclidean Distance (ED), Dynamic Time Warping (DTW), Manhattan Distance (MD) and Chebyshev Distance (CD) are used. Thus, ID_CM can take integer values between 1 and 4. Finally, variables $x_{10}$ and $x_{11}$ represent the parameters of the classifier.

It is worth to mention that the restriction of selecting a particular classifier with a specific similarity measure according with the chosen representation method is considered before evaluating the solution and the valid choice is adopted. Such valid combinations are summarized in Figure 2. The four μ-DEMS variants, as in the original DE, work with a real representation and a rounding process to obtain the corresponding integer values is carried out (if applicable). If odd integers are required, the ceil and floor functions are used to obtain them.

B. Fitness function

Each μ-DEMS variant looks for a competitive model based on the Error Rate (ER), this is described in Equation 4.

$$ ER = \frac{a}{b} \quad (4) $$

where $a$ represents the instances of the temporal database that were incorrectly classified and $b$ is the total number of instances in such database. To avoid overfitting [19]–[21] k-folds Cross Validation (CV) method [22] is adopted to compute the ER, as it is described in Equation 5, where $k$ defines the number of stratified subsamples (folds) chosen randomly but with roughly equal size. In this work a value of 5-folds was considered, based on [19].

$$ CVER = \frac{1}{k} \sum_{i=1}^{k} ER_i \quad (5) $$

IV. EXPERIMENTS AND RESULTS

A. Experimental setup

This section presents the experimental setup used to compare the performance of the four μ-DEMS variants under similar conditions in a set of six representative time-series databases of a well-known benchmark [23]. The detailed description of each temporal database can be found in Table III. The parameter settings used by the four μ-DEMS variants are shown in Table IV. This configuration is based on the values suggested in [24]. The exception was the replacement generation, whose suggested value was 100 and in this work it was modified to 10.

Based on the suggestions made in [19], five independent runs per each μ-DEMS variant and per each time-series database were computed. The termination condition on each run was 1,500 evaluations. Three experiments were designed: (1) a comparison of the final statistical results, (2) the convergence behavior, and (3) an analysis of the best final solution obtained.

<table>
<thead>
<tr>
<th>SAX</th>
<th>KNN-ED (MinDist ED-based)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAA</td>
<td>KNN-ED (Reduced ED to PAA)</td>
</tr>
<tr>
<td>PCA</td>
<td>KNN-DTW (Reduced DTW to PAA)</td>
</tr>
<tr>
<td></td>
<td>KNN-MD</td>
</tr>
<tr>
<td></td>
<td>KNN-CD</td>
</tr>
<tr>
<td></td>
<td>KNN-ED (Classical ED)</td>
</tr>
<tr>
<td></td>
<td>KNN-DTW (DTW with Lower Bound)</td>
</tr>
<tr>
<td></td>
<td>KNN-MD</td>
</tr>
<tr>
<td></td>
<td>KNN-CD</td>
</tr>
</tbody>
</table>

Figure 2. Variants for each similarity measure.
Table I. DE VARIANTS USED IN THE EXPERIMENTS, \( r_{\text{rand}} \) IS A RANDOM REAL NUMBER \( \in [0,1] \), \( j_{\text{rand}} \) IS A RANDOM INTEGER NUMBER \( \in [1, D] \). \( \vec{x}_{\text{best},g} \) IS THE BEST VECTOR AT GENERATION \( g \).

<table>
<thead>
<tr>
<th>Nomenclature</th>
<th>Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>rand/1/bin</td>
<td>( u_{i,g}^j = \frac{x_{i,j}^g + F\cdot (x_{i,j}^g - x_{i,j}^g)}{x_{i,j}^g} ) if ( r_{\text{rand}}[0,1] &lt; CR ) or ( j = j_{\text{rand}} ) otherwise</td>
</tr>
<tr>
<td>rand/1/exp</td>
<td>( u_{i,g}^j = \frac{x_{i,j}^g + F\cdot (x_{i,j}^g - x_{i,j}^g)}{x_{i,j}^g} ) while ( r_{\text{rand}}[0,1] &lt; CR ) or ( j = j_{\text{rand}} ) otherwise</td>
</tr>
<tr>
<td>best/1/bin</td>
<td>( u_{i,g}^j = \frac{x_{\text{best},j}^g + F\cdot (x_{\text{best},j}^g - x_{i,j}^g)}{x_{i,j}^g} ) if ( r_{\text{rand}}[0,1] &lt; CR ) or ( j = j_{\text{rand}} ) otherwise</td>
</tr>
<tr>
<td>best/1/exp</td>
<td>( u_{i,g}^j = \frac{x_{\text{best},j}^g + F\cdot (x_{\text{best},j}^g - x_{i,j}^g)}{x_{i,j}^g} ) while ( r_{\text{rand}}[0,1] &lt; CR ) or ( j = j_{\text{rand}} ) otherwise</td>
</tr>
</tbody>
</table>

Table II. DESCRIPTION OF THE METHODS USED AND THEIR PARAMETERS. TLS MEANS TIME-SERIES LENGTH.

<table>
<thead>
<tr>
<th>ID</th>
<th>Method</th>
<th>Type</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Spoly</td>
<td>SM</td>
<td>( k \in [1, 15] )</td>
<td>Polynomial order.</td>
</tr>
<tr>
<td>2</td>
<td>Moving Average</td>
<td>SM</td>
<td>( \mu )</td>
<td>Span size.</td>
</tr>
<tr>
<td>3</td>
<td>SAX</td>
<td>RM</td>
<td>( w \in [0.1, 7.5] )</td>
<td># of segments.</td>
</tr>
<tr>
<td>4</td>
<td>PSA</td>
<td>RM</td>
<td>( \sigma )</td>
<td>Alphabet size.</td>
</tr>
<tr>
<td>5</td>
<td>PCA</td>
<td>RM</td>
<td>( \rho \in [0, 1] )</td>
<td>SDE reduction.</td>
</tr>
<tr>
<td>6</td>
<td>KNN-ED</td>
<td>CM</td>
<td>( k \in [1, 15] )</td>
<td># of nearest neighbors.</td>
</tr>
<tr>
<td>7</td>
<td>KNN-DT</td>
<td>CM</td>
<td>( r \in [0.1, 0.6] )</td>
<td>Size reduction.</td>
</tr>
<tr>
<td>8</td>
<td>KNN-ED</td>
<td>CM</td>
<td>( k \in [1, 15] )</td>
<td># of nearest neighbors.</td>
</tr>
<tr>
<td>9</td>
<td>KNN-DT</td>
<td>CM</td>
<td>( k \in [1, 15] )</td>
<td># of nearest neighbors.</td>
</tr>
</tbody>
</table>

Table III. TIME-SERIES DATABASES USED IN THE EXPERIMENTS.

<table>
<thead>
<tr>
<th>No.</th>
<th>Name</th>
<th>Classes</th>
<th>Time-series length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Best</td>
<td>15</td>
<td>430</td>
</tr>
<tr>
<td>2</td>
<td>Beef</td>
<td>10</td>
<td>290</td>
</tr>
<tr>
<td>3</td>
<td>ECG200</td>
<td>2</td>
<td>359</td>
</tr>
<tr>
<td>4</td>
<td>OliveOil</td>
<td>4</td>
<td>570</td>
</tr>
<tr>
<td>5</td>
<td>FaceFour</td>
<td>10</td>
<td>570</td>
</tr>
<tr>
<td>6</td>
<td>GunPoint</td>
<td>2</td>
<td>150</td>
</tr>
</tbody>
</table>

Table IV. PARAMETER SETTING USED BY \( \mu \)-DEMS FOR EACH VARIANT.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CR</td>
<td>0.1</td>
</tr>
<tr>
<td>( \psi )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>10</td>
</tr>
</tbody>
</table>

Table V. STATISTICAL RESULTS (B: BEST, M: MEAN, W: WORST, SD: STANDARD DEVIATION, P: P-VALUE) OBTAINED BY EACH \( \mu \)-DEMS VARIANT. VALUES IN BOLDFACE MEAN THE BEST VALUES FOUND.

B. Experiment 1. Final results

The statistical results of the CVER values obtained by the four \( \mu \)-DEMS variants are summarized in Table V. The p-values obtained by the non-parametric 95%-confidence Kruskal-Wallis test applied to the results of the four variants are included as well. Based on such test there were not significant differences among the variants. However, in the samples of runs, lower CVER values were provided by best/1/bin in most databases. The exception was the coffee database, which is the one with the lowest number of instances (see Table III) and where rand/1/bin was the most competitive. Finally, the less competitive overall results were precisely provided by rand/1/bin. To enhance the statistical validation, as suggested in [25], the post-hoc Bonferroni test was computed by using two metrics (best and median values) to compare the performance of four variants considering the full set of instances (databases). Figure 3 shows the results of the Bonferroni test. The x-axis shows the confidence interval of midranges and the y-axis indicates the name of each \( \mu \)-DEMS variant compared. The post-hoc results confirm that observed in Table V, where best/1/bin provided slightly better results and that rand/1/bin was not competitive. From this experiment it can be concluded that the performance of the four \( \mu \)-DEMS variants is quite similar, perhaps due to the limited number of evaluations computed per run due to the high computational cost required to evaluate a single solution (as it will be discussed later). However, best/1/bin provided slightly better results.

C. Experiment 2. Convergence plots

The convergence plots of the run located in the median value of the five independent runs computed by the four variant per each database are shown in Figure 4. From those plots it is interesting to see that best/1/bin was able to clearly reach more competitive results by avoiding local optima in Beef, ECG200, and GunPoint databases. In OliveOil and in FaceFour best/1/bin reaches the same good result with respect
D. Experiment 3. Analysis of the final solutions

Table VI presents the best models suggested by each $\mu$-DEMS variant. The third column details the models with the minimum CVER per variant, while the fourth column shows the corresponding CVER value, and the fifth column indicates the processing time for that single run in minutes. Despite the fact that differences are observed in the solutions, there are important similarities. Regarding the smoothing method, Sgolay was the most preferred, while SAX and PCA were the most popular representation methods. Finally, the KNN classifier with Euclidean Distance appeared as the most suitable. The processing time varies considerably due to the different features of the temporal databases (as shown in Table III). However, such computational time (e.g., more than 4,000 minutes for a single run) is an issue which requires further research. Figure 5 shows a graphical example of a model selection (see suggested model in Table VI) applied to FaceFour database. The example shows the average behavior of the original data which is compared with its respective averaged smoothed data (see Subfigure 5(1)). In Subfigure 5(2) the averaged behavior of the smoothed data is presented again and it is compared with the reduced data (see Subfigure 5(3)).

V. CONCLUSIONS AND FUTURE WORK

A comparison of four $\mu$-DE variants based on rand/1/bin, rand/1/exp, best/1/bin and best/1/exp to solve an instance of the model selection problem on time-series databases was presented. The comparison was focused on analyzing the effect of the crossover and the type of base vector when dealing with the model selection in temporal databases. Each $\mu$-DEMS variant was tested in six databases through three experiments to analyze the final statistical results, the convergence behavior and the type of final solution reached. The overall assessment indicates that the $\mu$-DE is a viable option to find suitable models. Moreover, it was found that using the best vector in the population as the base vector coupled with the bin crossover is a good option to reach better final results. In contrast, if the temporal database requires significant computational time to evaluate models, changing the base vector with one chosen at random and using the bin crossover may be a good choice to get a competitive result faster. From the convergence behavior, it was noted that in a range of 150 to 200 generations it was possible to achieve competitive results by the rand variants. Furthermore, if the number of generations exceeded 200, best/1/bin was a capable variant to achieve better final results. Finally, the empirical comparison in this work showed that Sgolay, SAX with PCA, and KNN with Euclidean distance, all of them with suitable parameter values found by the $\mu$-DEMS variants, were the most competitive methods for smoothing, representation and classification, respectively. As future work, post-processing methods will be added to the encoding of the $\mu$-DEMS variants. Moreover, comparisons against other $\mu$-EAs will be carried out and other objectives (e.g. complexity of the model) will be considered.

VI. ACKNOWLEDGMENTS

The first author acknowledge support from the Mexican National Council for Science and Technology (CONACyT) through a scholarship number 259655 to pursue graduate...
studies at the University of Veracruz. The third author acknowledges support from CONACyT through project No. 220522.

References


Table VI. Models suggested by each μ-DEMS variant.

<table>
<thead>
<tr>
<th>Database</th>
<th>Variant</th>
<th>Model</th>
<th>CREV</th>
<th>Time (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beef</td>
<td>rand/2/bit</td>
<td>Smoothing: Moving Average (span 168), Representation: PCA(300), Classifier/Similarity Measure: KNN-MD(512)</td>
<td>0.6967</td>
<td>752.55</td>
</tr>
<tr>
<td></td>
<td>best/2/bit</td>
<td>Smoothing: Sgolay, Representation: PCA(300), Classifier/Similarity Measure: KNN-MD(512)</td>
<td>0.5367</td>
<td>1046.23</td>
</tr>
<tr>
<td>Coffee</td>
<td>best/2/exp</td>
<td>Smoothing: Moving Average (span 33), Representation: PAA(4000), Classifier/Similarity Measure: KNN-MD(64)</td>
<td>0.6967</td>
<td>1089.15</td>
</tr>
<tr>
<td>ECG200</td>
<td>best/1/exp</td>
<td>Smoothing: Sgolay, Representation: SAX<a href="138">79</a>, Classifier/Similarity Measure: KNN-DD(512)</td>
<td>0.0000</td>
<td>174.67</td>
</tr>
<tr>
<td>OliveOil</td>
<td>best/1/exp</td>
<td>Smoothing: Moving Average (span 156), Representation: PCA(64), Classifier/Similarity Measure: KNN-MD(512)</td>
<td>0.6667</td>
<td>532.55</td>
</tr>
<tr>
<td>FaceFour</td>
<td>best/1/exp</td>
<td>Smoothing: Sgolay, Representation: PCA(64), Classifier/Similarity Measure: KNN-MD(512)</td>
<td>0.7407</td>
<td>281.29</td>
</tr>
<tr>
<td>GunPoint</td>
<td>best/1/exp</td>
<td>Smoothing: Moving Average (span 11), Representation: PAA(64), Classifier/Similarity Measure: KNN-DD(512,8)</td>
<td>0.6969</td>
<td>532.55</td>
</tr>
</tbody>
</table>


