Supplementary material for the paper titled "Reliable early classification of time series based on discriminating the classes over time"

Analysis of runtimes of early classification methods ECDIRE, RelClass, EDSC and ECTS

Usue Mori, Alexander Mendiburu, Eamonn Keogh and Jose A. Lozano

In this document, we show the execution times of the four early classification methods compared in the main manuscript.

1 Run-time measurements

Particularly, execution times have been measured separately for two stages: first, the learning phase, in which the model is created, using the time series stored in the training set. Second, the prediction phase, in which we calculate the maximum time the model (created in the learning phase) needs to provide a class value or abstain at each timestamp. Note that this answer must be fast, since it must be obtained before the next data point arrives.

All the codes have been executed using the code provided by the authors directly, but in the case of ECDIRE the cross-validation has been executed in a sequential fashion, in order to be fair. However, we must emphasize that the execution times should be analyzed carefully, because the methods have been coded using different programming languages and by different programmers (with different skills, presumably). In addition, we ignore the effort made by each of the authors to optimize the code, which could make the comparison even more unfair.

Experiments have been conducted in a desktop computer with Ubuntu server 14 LTS operating system, an Intel(R) Core(TM) i5-2400S CPU @ 2.50GHz processor, and 4GB RAM. Additional software was needed to run the different methods:: Matlab 2012b (RelClass) , gnu g++ compiler 4.8.4 (ECTS, EDSC, and some functions of the other methods), and R 3.2.2 (ECDIRE).

2 Results

Tables 1 and 2 respectively show the execution times for the learning phase and the prediction step for each of the four methods. On the one hand, the learning phase execution time is the time needed to construct the entire model. On the other hand, the prediction time has been measured as the largest (worst case) time required by each model to provide a class value or abstain, taking into account every possible length of the incomplete incoming testing time series. Prediction times have been measured separately for each time series included in the testing set of the database. Next, mean and deviation values have been calculated.

It can be seen that, with regards to the training time, our method is the most expensive. This is mainly due to the 10x5-cross validation process included in the first step of the learning phase, which takes the largest part of the execution time. In this context, if the training database is very large, the training of the method could become prohibitive. However, the runtimes can be reduced drastically by applying a simple 5-fold cross validation process instead of a 10x5 cross validation as recommended by [1], or the cross-validation process could be executed in parallel, as we have done in our experimentation, which reduces the running time almost linearly with respect to the number of computing CPU cores available.

Additionally, in the context of early classification, the most crucial step is the prediction phase, in which the model should be able to provide an answer as soon as possible, before the next data point arrives. As seen in Table 2, most of the times the methods are able to make a decision in mili-seconds (or even in micro-seconds), which can be considered a fast response. However, the adequacy of these values is conditioned by the real scenario the models are applied to.

	RelClass	**ECTS**	**EDSC**	**ECDIRE**
SonyAiboRobotSurface	1.7795	0.0003	0.1125	67.9060
Gunpoint	1.1835	0.0044	5.0026	196.0030
Trace	0.5501	0.0115	136.8690	5314.2240
Yoga	1.6231	0.0673	16012.4000	2989.7700
Faceall	4.4999	0.1422	484.6060	93031.1890
UwaveGestureLibrary_z	7.8624	0.2264	18785.1000	96111.5970

Table 1: Execution times in seconds for completing the learning phase. Four early classification methods: RelClass, ECTS, EDSC, and ECDIRE.

	RelClass		**ECTS**		**EDSC**		**ECDIRE**	
	Avg	*Std*	*Avg*	*Std*	*Avg*	*Std*	*Avg*	*Std*
SonyAiboRobotSurface	13.3452	2.3429	0.0037	0.0018	0.0004	0.0005	2.3278	0.5514
Gunpoint	20.3896	1.4701	0.0089	0.0014	0.0009	0.0005	5.1467	2.2267
Trace	66.1225	3.0368	0.0393	0.0038	0.0027	0.0004	10.9600	0.7375
Yoga	707.1048	7.6525	0.1215	0.0068	0.0338	0.0049	28.6457	3.1094
Faceall	301.9426	9.7631	0.0721	0.0047	0.0177	0.0015	56.4982	5.8985
UwaveGestureLibrary_z	2292.1204	5.8210	0.2661	0.0113	0.1340	0.0470	108.3780	7.0843

Table 2: Execution times in mili-seconds (worst case) required by each model to provide a class value or abstain. Four early classification methods: RelClass, ECTS, EDSC, and ECDIRE.

Finally, as commented previously, comparing the times of the different methods is not entirely fair because they have been coded by different programmers and are based on different languages. In this context, we also provide the theoretical complexities of the prediction phase, which is a much more relevant measure: $O(\max\{N \cdot M, N^2\})$ for ECDIRE, $O(N \cdot M)$ for ECTS, $O(S \cdot L)$ for EDSC, $O(\max\{N \cdot D, D^2\})$ for RelClass using dimensionality reduction and $O(\max\{N \cdot M, M^2\})$ for RelClass without dimensionality reduction. N is the number of series in the training set, M is the length of the series in the training set (we suppose, for the sake of simplicity, that all series are of the same length), S is the number of shapelets that the EDSC method includes into its library, L is the length of the largest shapelet in this library, and D is the dimension of the time series after LDG dimensionality reduction in RelClass. Note that the complexities do not differ much from method to method and they are in correspondence with the values obtained in the table, as expected.

References

[1] Rodríguez, J. D., Pérez, A., Lozano, J. A., 2013. A general framework for the statistical analysis of the sources of variance for classification error estimators. Pattern Recognition 46 (3), 855–864.