ODAC: Hierarchical Clustering of Time Series Data Streams*

Pedro Pereira Rodrigues† João Gama‡ João Pedro Pedroso§

Abstract

This paper presents a time series whole clustering system that incrementally constructs a tree-like hierarchy of clusters, using a top-down strategy. The Online Divisive-Agglomerative Clustering (ODAC) system uses a correlation-based dissimilarity measure between time series over a data stream and possesses an agglomerative phase to enhance a dynamic behavior capable of concept drift detection. Main features include splitting and agglomerative criteria based on the diameters of existing clusters and supported by a significance level. At each new example, only the leaves are updated, reducing computation of unneeded dissimilarities and speeding up the process every time the structure grows. Experimental results on artificial and real data suggest competitive performance on clustering time series and show that the system is equivalent to a batch divisive clustering on stationary time series, being also capable of dealing with concept drift. With this work, we assure the possibility and importance of hierarchical incremental time series whole clustering in the data stream paradigm, presenting a valuable and usable option.

1 Introduction

In recent real-world applications, data flows continuously from a *data stream* at high speed, producing examples over time, usually one at a time. Traditional models can't adapt to the high speed arrival of new examples [2]. This way, algorithms have been developed that aim to process data in *real-time*. These algorithms should be capable of, at each given moment, supply a compact data description and process each example in constant time and memory [1]. Among different techniques known in literature, hierarchical models propose better versatility as they do not require an a priori definition of the number of clusters to find. From these, divisive methods seem to be the most appropriate to online procedure, building cluster structure in a top-down strategy. Most of the work in incremental clustering of data streams has been concentrated on example clustering rather than variable clustering. Moreover, variable clustering is usually considered a batch offline procedure. Incremental variable clustering isn't too surveyed yet, so we may find a lot of possibilities for contributions in this area of research, studying time series whole clustering, which is meaningful as we do not try to cluster subsequences of time series [5].

The main objective of this work is to present an incremental system to hierarchically cluster variables, where each variable is a time series and each new example that is fed to the system is the value of an observation of all time series in a particular time step.

In the next section, a review over incremental clustering analysis for data streams is performed. Section 3 presents the proposed system. In section 4, experimental evaluation on artificial and real data is presented, supporting the quality of the system. We finalize the exposition with section 5, where concluding remarks and future work are presented.

2 Related Work

Clustering is usually taken as a batch procedure, statically defining the structure of objects. Nevertheless, incremental methods have been developed, which can cope with data stream analysis. An overview of clustering analysis can be found in [4]. Hierarchical algorithms have three major advantages over partitional methods: a) don't require user-predefined number of target clusters; b) don't make any assumptions about data distribution; c) don't need any explicit representation rather than a pair-wise dissimilarity matrix. They set their basis on dissimilarities among elements of the same group, defining a hierarchy of clusters.

Methods exist to cluster examples over data streams. Apart from other good examples, *COBWEB*, *Single Pass K-Means*, *BIRCH*, *CluStream*, *CURE* and *STREAM* are well-known incremental methods. Unfortunately, incremental methods to perform clustering on variables are hard to find. Moreover, with the advent of data streams, the assumption that examples are generated at random according to some stationary probability distribution is being disregarded, and new methods are being proposed to deal with changes on the concept of the distribution producing the examples, that is, *concept drift* [7].

*Thanks to the Plurianual support attributed to LIACC, projects RETINAE (PRIME/IDEIA/70/00078) and ALES II (POSI/EIA/55340/2004).

†LIACC, University of Porto. prodrigues@liacc.up.pt
‡LIACC & FEP, University of Porto. jgama@liacc.up.pt
§LIACC & DCC-FCUP, University of Porto. jpp@ncc.up.pt

3 Online Divisive-Agglomerative Clustering

The ODAC *(Online Divisive-Agglomerative Clustering)* system is a variable clustering algorithm that constructs a hierarchical tree-shaped structure of clusters using a top-down strategy. The leaves are the resulting clusters, with a set of variables at each leaf. The union of the leaves is the complete set of variables. The intersection of leaves is the empty set. The system encloses an incremental distance measure and executes procedures for expansion and aggregation of the tree-based structure, based on the diameters of the clusters. The main setting of our system is the monitoring of existing clusters' diameters. In a divisive hierarchical structure of clusters, considering stationary data streams, the overall intra-cluster dissimilarity should decrease with each split. For each existing cluster, the system finds the two variables defining the diameter of that cluster. If a given heuristic condition is met on this diameter, the system splits the cluster and assigns each of the chosen variables to one of the new clusters, becoming this the *pivot* variable for that cluster. Afterwards, all remaining variables on the old cluster are assigned to the new cluster which has the closest pivot. New leaves start new statistics, assuming that only forthcoming information will be useful to decide whether or not this cluster should be split. This feature increases the system's ability to cope with changing concepts as, later on, a test is performed such that if the diameters of the children leaves approach the parent's diameter, then the previously taken decision may no longer reflect the structure of data, so the system re-aggregates the leaves on the parent node, restarting statistics. The forthcoming sections describe the inner core of the system.

3.1 Incremental Dissimilarity Measure We use Pearson's correlation coefficient between time series as *similarity* measure. Deriving from the correlation between two time series a and b calculated in [6], the factors used to compute the correlation can be updated incrementally, achieving an exact incremental expression for the correlation:

$$corr(a,b) = \frac{P - \frac{AB}{n}}{\sqrt{A_2 - \frac{A^2}{n}}\sqrt{B_2 - \frac{B^2}{n}}} \tag{3.1}$$

The *sufficient statistics* needed to compute the correlation are easily updated at each time step: $A = \sum a_i,\ B = \sum b_i,\ A_2 = \sum a_i^2,\ B_2 = \sum b_i^2,\ P = \sum a_i b_i$. In ODAC, the dissimilarity between variables a and b is given by an appropriate metric, the *Rooted Normalized One-Minus-Correlation* given by

$$rnomc(a,b) = \sqrt{\frac{1 - corr(a,b)}{2}} \tag{3.2}$$

with range $[0,1]$. We consider the cluster's *diameter* to be the highest dissimilarity between two time series belonging to the same cluster, or the variable variance in the case of clusters with single variables.

3.2 Growing the Hierarchy The main procedure of the ODAC system is to grow a tree-shaped structure that represents the hierarchy of the clusters present in the data. One problem that usually arises with this sort of models is the definition of a minimum number of observations necessary to assure convergence. A common way of doing this includes a user-defined parameter; after a leaf has received at least n_{min} examples it is considered ready to be tested for splitting. Another approach is to apply techniques based on the Hoeffding bound to solve this problem. The Hoeffding bound has the advantage of being independent of the probability distribution generating the observations [2], stating that after n independent observations of a real-valued random variable r with range R, and with confidence $1-\delta$, the true mean of r is at least $\overline{r} - \epsilon$, where $\overline{r}$ is the observed mean of the samples and

$$\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}} \tag{3.3}$$

As each leaf is fed with a different number of examples, each leaf C_k will possess a different value for ϵ, designated ϵ_k. Let $d(a,b)$ be the heuristic measure used to choose the pair of time series representing the diameter (we use real value of the distance measure), and $D_k = \{(x_i, x_j) \mid x_i, x_j \in C_k, i < j\}$ be the set of pairs of variables included in a specific leaf C_k. After seeing n samples at the leaf, let $(x_1, y_1) \in \{(x,y) \in D_k \mid d(x,y) \geq d(a,b), \forall (a,b) \in D_k\}$ be the pair of variables with maximum dissimilarity within the cluster C_k, $D'_k = D_k \backslash \{(x_1, y_1)\}$ and $(x_2, y_2) \in \{(x,y) \in D'_k \mid d(x,y) \geq d(a,b), \forall (a,b) \in D'_k\}$, $d_1 = d(x_1, y_1)$ and $d_2 = d(x_2, y_2)$. Let $\Delta d = d_1 - d_2$ be a new random variable, the difference between the observed values. Applying the Hoeffding bound to Δd, if $\Delta d > \epsilon_k$, we can confidently say that, with probability $1-\delta$, the difference between d_1 and d_2 is larger than zero, and select (x_1, y_1) as the pair of variables representing the diameter of the cluster. That is,

$$d_1 - d_2 > \epsilon_k \Rightarrow diam(C_k) = d_1 \tag{3.4}$$

With this rule, the ODAC system will only apply the splitting test when the true diameter of the cluster is known with statistical confidence given by the Hoeffding bound. This rule has another feature: it supports the decision of splitting or aggregation as it triggers the moment when the leaf has been fed with enough examples to support it.

3.2.1 Feeding the System In the ODAC system, each example is processed only once. The system incrementally updates, at each new example arrival, the sufficient statistics needed to compute the dissimilarity matrix, enabling its application to clustering of data streams. The dissimilarity matrix for each leaf is only computed when it is being tested for splitting or aggregation, after receiving a minimum number of examples. When processing a new example, only the leaves are updated, reducing computation of unneeded dissimilarities; this speeds up the process every time the structure grows. We have decided to model the time series first-order differences in order to reduce the negative effect of autocorrelation on the Hoeffding bound, preventing larger errors. The missing values can be easily treated with a zero value, considering that, when unknown, the time series is constant.

3.2.2 Splitting Criteria Several criteria and heuristics can be used to perform division of a cluster of variables using the previously presented distance measure, supported by the confidence level given by the Hoeffding bound. The splitting criterion should reflect some relation among the distances between variables of the cluster. Given this fact, we can impose a cluster to be split if it includes a high difference between $(d_1 - \overline{d})$ and $(\overline{d} - d_0)$, where d_0 stands for the minimum distance between variables belonging to the cluster and $\overline{d}$ is the average of all distances in the cluster. In our approach, we relate the expression with the global difference $d_1 - d_0$. Our heuristic is the following: for a given cluster C_k, we choose to split this leaf if:

$$(d_1 - d_0)\left|d_1 + d_0 - 2\overline{d}\right| > \epsilon_k \tag{3.5}$$

This criterion has also appeared to be useful on acting as a stopping criterion for the hierarchy growing phase. When a split point is reported, the pivots are variables x_1 and x_2 where $d_1 = d(x_1, x_2)$, and the system assigns each of the remaining variables of the old cluster to the cluster which has the closest pivot.

3.3 Aggregating at Concept Drift Detection If a leaf expresses changes in the relations between variables it represents, then perhaps the split decision that has created this leaf may in fact be outdated. The heuristic that is adopted in this work is the analysis of diameters. This way, no computation is needed between the variables of the two siblings. For each given leaf C_k, we shall test the diameters of C_k, C_k's sibling (C_s) and C_k's parent (C_j), assuming that the sum of the children diameters should not be as large as two times the diameter of the parent. We define a new random variable $\Delta a = 2 \cdot diam(C_j) - (diam(C_k) + diam(C_s))$. Applying the Hoeffding bound to this random variable, if $\Delta a > \epsilon_j$ then the condition is met, so the splitting decision is still a good approach. Given this, we choose to aggregate on C_j if

$$2 \cdot diam(C_j) - (diam(C_k) + diam(C_s)) < \epsilon_j \tag{3.6}$$

supported by the confidence given by the parent's consumed data. The system decreases the number of clusters as previous division no longer reflects the best divisive structure of data. The resulting leaf starts new computations and a concept drift is detected.

3.4 When a Tie Occurs To distinguish between the cases where the cluster has many variables nearly equidistant and the cases where there are two or more highly dissimilar variables, a tweak must be done. Having in mind the application of the system to a data stream with high dimension, possibly with hundreds or thousands of variables, we turn to a heuristic approach. Based on techniques presented in [2], we introduce a parameter to the system, τ, which determines how long will we let the system check for the real diameter until we force the splitting and aggregation tests. At any time, if $\tau > \epsilon_k$ the system applies the tests, assuming the leaf has been fed with enough examples, hence it should consider the highest distance to be the real diameter.

3.5 Memory Usage and Time Complexity The ODAC procedure presents the required features of adaptive learning systems. A system which aims at efficiently clustering data streams must comply with constant memory occupation with constant execution time in respect to the number of examples [1]. In ODAC, system space complexity is constant on the number of examples, even considering the infinite amount of examples usually present in data streams. An important feature of this algorithm is that every time a split is performed on a leaf with n variables, the global number of dissimilarities needed to be computed at the next iteration diminishes at least $n - 1$ (worst case scenario) and at most $(n/2)^2$ (best case scenario). As imperious, the time complexity of each iteration of the system is constant given the number of examples, even decreasing with every split occurrence, being therefore capable of addressing data streams.

4 Experimental Evaluation

The first evaluation of the proposed system is made using synthetic data sets created with specific definitions, described in section 4.1. Later on, the system is tested with real data, from the PDMC Sensor Data Set and the Electrical Demand Data Set.
For each data set (with n variables), 10 runs of *k-means*

are executed, each one with all possible numbers of clusters, k. Quality measures are calculated in these runs and an average is considered to find the best number of clusters for the dataset. Afterwards, ODAC is applied in the same data set to enable a comparison between the final structure and the one provided by *k-means*. Comparison with a batch DIANA system, using the same correlation-based distance measure, is also performed. The cluster validity indexes used to evaluate the results are the *MHΓ - Modified Hubert's Γ Statistic* and the *DVI - Dunn's Validity Index* [3]. The ODAC system allows the analysis of another quality measure, the *CPCC - Cophenetic Correlation Coefficient* [4], which measures quality in hierarchical structures. High values of any of these three indexes indicate the presence of a good clustering structure. On all experiments, the Hoeffding bound δ parameter was set to 0.05; the τ parameter of the ODAC system was set to 0.02.

4.1 Evaluation on Artificial Data The data sets used in this section were created using a time series generator that produces n time series belonging to a predefined number k of clusters with a noise constant β. Each cluster c_k has a pivot time series p, and the remaining time series are created as $p + \lambda$ where

$$\lambda \sim U(-\beta p, \beta p) \tag{4.7}$$

We created five data sets with ten variables each, observed along 100K examples, with $\beta = 0.3$ and especially prepared to test different hypothesis: *closed* (one cluster), *sparse* (no clustering), *two clusters* and $4C$ (four clusters). The structure of $4C$ was created as

$$\{\{a1, a2\}, \{a3, a4, a5\}, \{a6\}, \{a7, a8, a9, a10\}\}.$$

For all these stationary data sets, ODAC finds the correct structure, indicating good clustering capabilities.

4.1.1 Concept Drift Data Set For a more complex evaluation of system's dynamics (splitting and aggregation), we create a drifting data set. The data set used is equivalent to $4C$ in the first 50K examples, and exactly at that time, the configuration changes to

$$\{\{a1, a2, a3, a4, a10\}, \{a5, a6, a7\}, \{a8, a9\}\}$$

The system models the first concept with ease, achieving a stable structure in 18448 examples. The system starts to try to grow a sub-tree after 3220 examples from the drift, searching for a better structure. After 12448 examples from the concept drift, the system collapses all the structure, assuming the concept is completely different from the previous one. The second concept is found 21672 examples after the real concept drift. The global results suggest good performance on splitting, concept drift detection and aggregation.

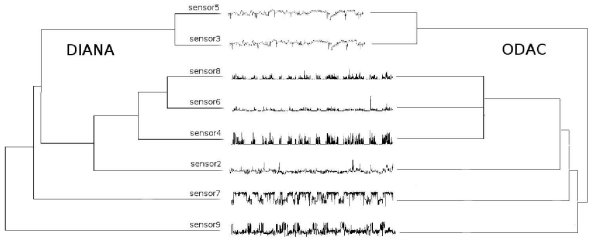

Figure 1: Comparison between ODAC and DIANA (user01 data set). Although higher-level structure is different, at cluster level both structures are quite close.

4.2 PDMC Sensor Data Sets The training data set for the PDMC competition consists of approximately 10,000 hours of sensor data, containing several variables: userID, sessionID, sessionTime, characteristic[1..2], annotation, gender and sensor[1..9]. We concentrate on sensors 2 to 9, since we are interested in finding the relations between different sensor data, and extract all data by userID, ending with several data sets of eight continuous variables. In figure 1, a comparison between ODAC and a batch DIANA system using the same dissimilarity measure over the data of one user (user01 with 93344 examples) is shown, presenting the closeness between the two results. For other userIDs, results are very similar. We also try to find the right number of clusters for each user with more than 50K examples, possibly the same for all. Tables 1 and 2 present the evaluation for *user06* and *user25* data sets. Apparently, from a conservative point of view, the best number of clusters is 3. Figure 2 sketches the ODAC system's final structure, for the same data sets. The same structure was found, agreeing with the expected number of clusters given by *k-means* experience.

System	*nc*	MHΓ	DVI	CPCC
K-Means	2	0.141	0.300	–
	3	**0.308**	**0.300**	–
	4	0.311	0.198	–
ODAC	**3**	0.377	0.891	0.251

Table 1: K-Means and ODAC Quality Results (user06)

System	*nc*	MHΓ	DVI	CPCC
K-Means	2	0.099	0.272	–
	3	**0.360**	**0.325**	–
	4	0.362	0.242	–
ODAC	**3**	0.191	1.026	0.479

Table 2: K-Means and ODAC Quality Results (user25)

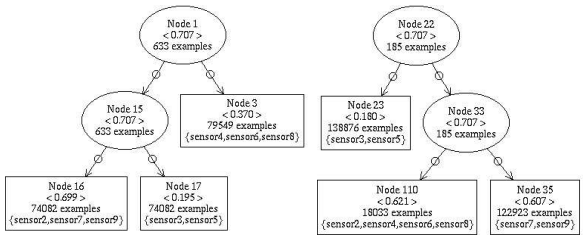

Figure 2: ODAC Final Structure (user06 and user25)

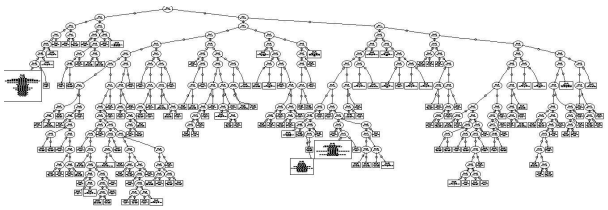

Figure 3: ODAC Final Structure (active power)

4.3 Electrical Demand Data Set Time series of electrical data are one of the most widely studied sets of data, mainly for forecast purposes, usually by means of neural networks. From the raw data received at each sub-station, gathered during four months, we extract only the variables related to load intensity (I), active (P) and reactive (Q) power (according to the, possibly erroneous, variable ID). Each example represents the average value for the past five minutes of each variable, resulting in data sets with 34734 examples, from which we model only active power (887 variables). The final structure can be analyzed on figure 3. The image is rather large and is presented only as an example of the structure obtained as a whole, revealing a tree-like structure, away from the list-like worst-case scenario. We present a sketch of the variables, for one of the clusters, in figure 4. This cluster expresses good intra cluster correlation ($\mu = 0.629$, $\sigma = 0.227$), considering that this is real data from five variables along 7385 observations.

5 Conclusions

This paper introduces a time series whole clustering system that incrementally constructs a hierarchy of clusters from a divisive point of view. The main setting of the system is the monitoring of existing clusters' diameters. The examples are processed as they arrive, using a single scan over the data. The system incrementally computes the dissimilarities between time series, maintaining and updating the sufficient statistics at each new example arrival. One important feature of the system is that the dissimilarity matrix is updated only for the cluster currently being tested, diminishing the global number of dissimilarities needed to be computed at each step. The system uses a correlation-based dissimilarity measure and supports the splitting and aggregating decisions on a significance level given by the Hoeffding bound. Experimental results show competitive performance when compared with batch clustering analysis, evolving and adapting in the presence of concept drift. Future work will focus on the application of the system to more real-world data and on a fuzzy approach to the assignment of variables to new clusters.

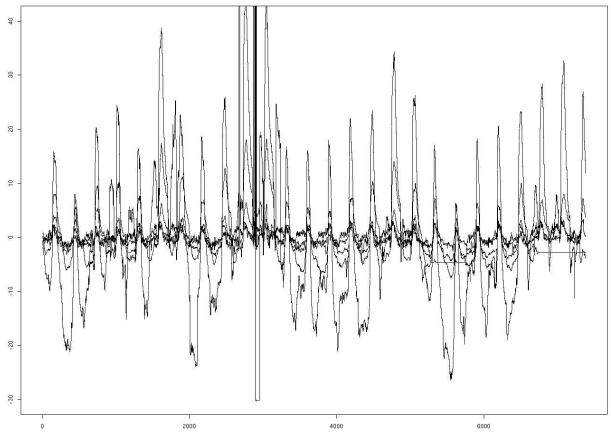

Figure 4: Cluster Plot (active power). In this cluster, five variables are kept together, along 7385 examples.

References

[1] D. Barbará. Requirements for clustering data streams. *SIGKDD Explorations*, 3(2):23–27, January 2002.

[2] P. Domingos and G. Hulten. Mining high-speed data streams. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 71–80. ACM Press, 2000.

[3] M. Halkidi, Y. Batistakis, and M. Varzirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3):107–145, 2001.

[4] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.

[5] E. J. Keogh, J. Lin, and W. Truppel. Clustering of the time series subsequences is meaningless: Implications for previous and future research. In *Proceedings of the IEEE International Conference on Data Mining*, pages 115–122. IEEE Computer Society Press, 2003.

[6] M. Wang and X. S. Wang. Efficient evaluation of composite correlations for streaming time series. In *Advances in Web-Age Information Management - WAIM 2003*, pages 369–380. Springer Verlag, 2003.

[7] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(2):69–101, 1996.