Self-Organized Mapping for environmental non-determinism simulation at WEKA

Ricardo Diniz Caldas - 18/0040014
Computer Science Dept. - Universidade de Brasilia
Brasilia, Brazil

***Abstract*—Environmental non-determinism demands complex reasoning mechanisms for systems eager to achieve goals on partially-observable and unknown environments. Lately, scientists have been exploring software capable of reorganizing its own internal structure to cope with environmental uncertainties, however it's not trivial to apply the methods and techniques to developed self-adaptive systems as they may present unpredictable behavior if adaptations are not well validated in design phase. In our study group, we developed a simulation of a Body Sensor Network system with vital signal generation through probabilistic models to simulate environmental non-determinism. In the current study, we apply an one-dimensional self-organized mapping neural network for clustering heart rate data values into ranges that will represent the markov chain states.**

***Index Terms*—Self-organized maps, markov chain, self-adaptive software**

I. INTRODUCTION

Volatile environments demands robust systems that cope with internal and external changes. Design-time strategies are unavailable as engineers should be able to elicit all possible run-time situations and implement cause-effect rules that deals with it. Self-adaptive systems (SAS) provides means of modifying itself in run-time due to changing needs, in which atomic components combined may present distinct global behaviors at run-time.

Ongoing studies argues that there is an high capacity of providing adaptable behavior through self-adaptive software systems. However, exploring such solutions demands software-intensive applications built over composable architectures that operates in variable environments, which may be costly and dangerous when dealing with safety-critical domains. We are developing, in the Software Engineering Lab (LES) at UnB, a simulation of an Body Sensor Network (BSN) application to monitor patients vital signals at the Intensive Care Unit. At first, the BSN should simply collect data from a set of sensors (e.g. thermometer, electrocardiograph, pulse oximeter), fuse it and analyze whether the patient is at normal or critical risk state.

Simulating non-deterministic physiological data is not a trivial task since the human vital signals are in constant change and patients in ICU situations are up to sudden state transitions. However with enough ICU data in hand we can model vital signals evolution over time with first-order markov chains and use it to simulate sensor data generation regarding the needed uncertainties. To do so, we would need to: (i) extract the possible states of the data and (ii) unveil the probability distributions attributed to transitions between states.

Weka is a collection of tools widely used to perform machine learning and data mining for data scientists and hobbyists. It contains tools for data preparation, classification, regression, clustering, association rules mining, and visualization. Feature extraction algorithms, such as clustering, can be used to extract the sensor data states for example. In this work, we chose to apply the plug'n'play Weka's self-organizing mapping (SOM) algorithm to gather the states from the data in order to build the aforementioned markov chain.

In the next sections we briefly discuss the underlying theory and applications tools used in the study, in the Background. Then, we detail the data used and simulation performed as experiment. Furthermore, we discuss the results obtained. Last, we conclude our work with final remarks.

II. BACKGROUND

A. Self-Organized Maps

Self-organized maps (SOM), proposed in the early 80's by Kohonen, has been widely used in industrial applications such as pattern recognition, biological modeling, signal processing, and data mining [1]. Among competitive learning neural networks, the SOM algorithm is commonly applied in non-supervised feature extraction solutions and relies on biomimetic relations with cerebral cortex sensory mappings. In short, the algorithm consists of a set of neighborhood interactive neurons that compete amongst each other, in which a winner neuron updates itself and respective neighbors weights according to similarities with inputs patterns.

Structured as m-dimensional lattice, the network maps n-dimensional inputs into clusters representing the feature extracted on a two-phase process. First, the mechanism topologically orders the neurons over a pre-set number of epochs and, furthermore, the feature map goes through an interactive convergence phase of statistical analysis.

The ordering phase operates through three processes: (i) initialization, (ii) competition and (iii) cooperation. The initialization consists of setting all the neurons weight vectors (W_i) with randomized small values. The competition comprises the iteration over all input space instances applying a proximity function (e.g. euclidean squared distance) between the input instance and each neuron weight vector, to define the winning neuron (I), by computing the minimum calculated value. And finally, the winning neuron and its respective neighbors collaboratively have its weights updated. The processes (ii) and (iii) are executed over and over through all ordering epochs.

In the other hand, the convergence phase, receives the trained network and runs over each neuron computing statistics such as average, standard deviation, mean, min and max values for further clustering analysis.

Finally, the SOM technique outputs a mapping according to patterns or features encountered in the input in an ordered fashion. The most usual way to display the clustered structure is through distance matrices, such as U-Matrix.

B. SOM@Weka

Weka is an open-source java-based conglomerate of tools with algorithms ready to perform data science experiments. It has intuitive tools for data generation, pre-processing, filtering, classifying, clustering, estimating and visualizing developed by an huge community of data scientists and hobbyists. It also comes with a built-in SOM algorithm 1 in which the user must provide a set of parameters to configure the network structure and its input space. For instance, the user must provide: (i) learning rate (L), (ii) ordering epochs (O), (iii) convergence epochs (C), (iv) lattice height (H) and (v) lattice width (W). Depending on the experiment, the user is able to configure how the algorithm will use the uploaded data dividing it in training and test data through percentages or even selecting attributes to ignore.

(i) -L is the initial training rate for the algorithm.
Constraints: $(0.01 < L \leq 1)$

(ii) -O is the number of epochs in ordering phase.
Constraints: $(O \geq 2000)$

(iii) -C is the number of epochs to train through.
Constraints: $(C \geq 1000)$

(iv) -H is the height of lattice.
Constraints: $(H \geq 1)$

(v) -W is the width of lattice.
Constraints: $(W \geq 1)$

```
Input: Input, Learning Rate, Ordering Epoch,
       Convergence Epoch, Height, Width
Output: Clusters
M = Height * Width;
Initialize neuron centroids $w_i(value_{rand})$, i = 0,1,...,M;
while Epochs are not over do
    for Input from n = 1,...,InputSIZE do
        $x^n$ is the current input;
        SELECT winner neuron m;
        UPDATE neighbor neurons weights;
        Gradually DECREASE L;
    end
end
```

Algorithm 1: Weka's SOM algorithm

The SELECT winner function dictates how the SOM training algorithm compares each input value with the current neuron weights, through proximity functions. The user is able to develop and define its own proximity function in Weka, however the defaut and most used one is the euclidean distance, presented in the equation 1.

$$d_j(x) = \sum_{i=1}^{D}(x_i - w_{ji}) \qquad (1)$$

In which i represents the input value instance to be compared and j the neuron within the comparison; And so, x_i is the numerical value that represents i, and w_{ji} the respective weight between instance i and neuron j. Thus, the computation of equation 1 outputs an array d_j where the neuron associated to the minimum value is the one chosen as winner. So, it will have its weights (w_{ij}) updated.

III. Experiment

Our interest with the study is to build a markov model to simulate environmental uncertainties within the possible values provided by physiological signals, then in this section we present the data set collected that is applied to the SOM clustering, its configuration and execution context. It is important to mention that the experiment ran on a Windows 10 64-bit over an Intel core i7-5500U CPU @ 2.40Ghz with 8.00GB primary memory.

A. Data

The data used for experimentation was obtained from the 2012 Physionet challenge[1] that consisted of predicting mortality of ICU patients. We collected 4,000 sets out of the 12,000 disposed by the challenge, which each record corresponds to a 48h time series of anonymous patient in ICU. Each record were made available in .csv files containing 7 attributes that would define the patient, as shown in table I, and the vital signals collected for each specific patient associated with timestamps that would represent the time passing from the beginning of data collection.

Attribute	**Type**	**Description**
RecordID	integer	a unique integer for each ICU stay
Age	integer	years
Gender	integer	0: female, or 1: male
Height	integer	cm
Weight	float	kg
ICUType	integer	*

*1: Coronary Care Unit, 2: Cardiac Surgery Recovery Unit, 3: Medical ICU, or 4: Surgical ICU

TABLE I: Attributes and respective types defining each patient

Time series vital signals were observed once, more than once or not at all in each record, those comprise 37 variables and some are presented in table II.

Albumin (g/dL)	HR (bpm)	$PaCO_2$ (mmHg)
Bilirubin (mg/dL)	Lactate(mmol/L)	Blood pH
Cholesterol (mg/dL)	Na (mEq/L)	RespRate (bpm)
$FiO_2(0-1)$	NIDiasABP (mmHg)	SysABP (mmHg)
Glucose (mg/dL)	Temp (^oC)	Urine (mL)

TABLE II: Reduced example set of measured vital signals

[1] https://physionet.org/challenge/2012/#data-for-the-challenge

B. Simulation

At first we decided to apply the technique as an experiment only to Heart Rate (HR) data, so a pre-processing step would have to be taken before uploading the files to Weka. We executed a script that printed out all the patient identification attributes followed by the HR time series values with respective timestamps to an unique output.csv file, excluding all the other time series possible variables, which would be uploaded to the tool. We chose HR data because not only it is relevant to vital signals monitoring in ICU patients, but also each patient has an average of 57.13 HR measurements per record, which would give us enough data to achieve high precision clustering. The distribution of the sampled data is presented in figure 1.

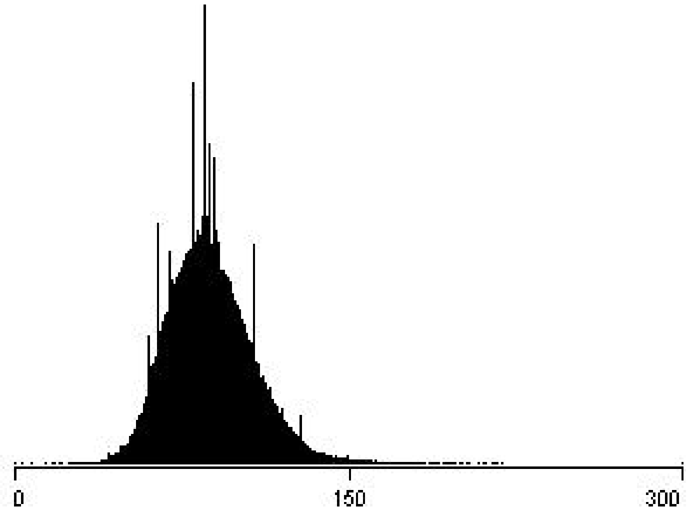


Fig. 1: Heart rate distribution among the ICU patients

Secondly, we set up the neural network using the parameters described in section II, shown in table III. We have also ignored the patients identification attributes, so the algorithm would focus on determining clusters based only on the HR values.

L	O	C	H	W
0.02	2000	1000	5	1

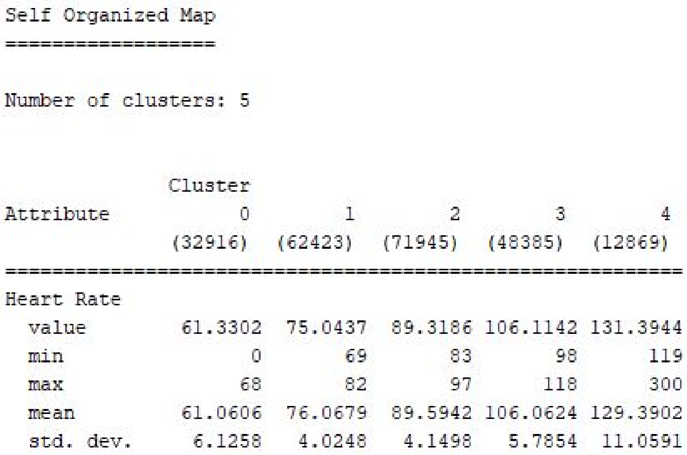
TABLE III: Experimental neural network's configurations

Finally we executed the SOM algorithm and the results will be presented and discussed in the following section.

IV. Results

After 599.85 seconds, the tool prompted ranges obtained by the clustering method, as stated in figure 2. We have obtained 5 clusters (ranges) that might be classified, later on, by domain experts with meaningful classes regarding for example the ICU patient‘s health status. In our case, the ranges shall be used to represent the markov chain states. Figure 3 details the linear graph over the array of values of heart rate, with the colored devised clusters.

No validation indexes were applied to the results to evaluate whether the outcome clusters were optimal, because we would have to vary the algorithm configuration to gather different clusters and statistical analysis and it would be too time consuming for the current project. We, however, will explore it in future works.



```
Self Organized Map
==================


Number of clusters: 5


                 Cluster
Attribute              0         1         2         3         4
                 (32916)   (62423)   (71945)   (48385)   (12869)
================================================================
Heart Rate
  value          61.3302   75.0437   89.3186  106.1142  131.3944
  min                  0        69        83        98       119
  max                 68        82        97       118       300
  mean           61.0606   76.0679   89.5942  106.0624  129.3902
  std. dev.       6.1258    4.0248    4.1498    5.7854   11.0591
```

Fig. 2: Clustering details

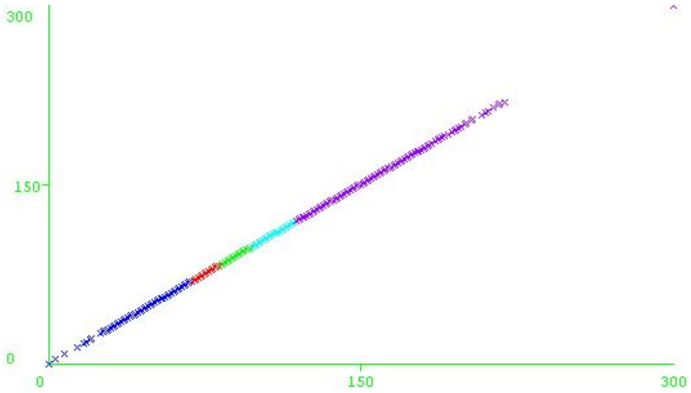


Fig. 3: Linear graph HR x HR

V. Conclusion

In this study, we propose the usage of one-dimensional clustering to extract features that lately will represent the states of a markov chain model to simulate environmental uncertainties of vital signals monitoring. We obtained five clusters, as expected, but as discussed couldn't validate the algorithm performance. Despite, the results doesn't elucidate any news about possibles ranges to classify the heart rate input array, they show that the aligned features may assist domain experts on characterizing the value representation, and so be the states of the markovian model of the environment.

VI. Acknowledgment

This work has been partially funded by Ricardo Caldas' CAPES scholarship.

References

[1] Sitao Wu and Tommy W.S. Chow. Clustering of the self-organizing map using a clustering validity index based on inter-cluster and intra-cluster density. *Pattern Recognition*, 37(2):175 – 188, 2004.