

Combining the Self-Organizing Map and K-Means Clustering for On-line Classification of Sensor Data

Kristof Van Laerhoven

Starlab Research Laboratories
Englandstraat 555
B-1180 Brussels, Belgium
kristof@starlab.org

Abstract. Many devices, like mobile phones use contextual profiles like “in the car” or “in a meeting” to quickly switch between behaviors. Achieving automatic context detection, usually by analysis of small hardware sensors, is a fundamental problem in human-computer interaction. However, mapping the sensor data to a context is a difficult problem involving near real-time classification and training of patterns out of noisy sensor signals. This paper proposes an adaptive approach that uses a Kohonen Self-Organizing Map, augmented with on-line k-means clustering for classification of the incoming sensor data. Overwriting of prototypes on the map, especially during the untangling phase of the Self-Organizing Map, is avoided by a refined k-means clustering of labeled input vectors.

1 Introduction

Human-computer interaction research suggests that applications should make obvious decisions autonomously, so that the user does not have to be bothered. One crucial step towards this paradigm is to enable the system to capture its context (location, activity of the user, state of the device itself, etc.) autonomously, which is usually established by adding a module that does pattern matching on incoming data from hardware sensors. Context awareness (see [1] for an overview) is a particularly hot topic in research fields where user interfacing is less obvious, like wearable computing and ubiquitous computing.

Well-known examples where context awareness would improve the application are mobile phones that have different notification behaviors (e.g. ringing or vibrating) for different contexts (e.g. “having a conversation” or “at home”), or handheld computers that start up applications depending on the context (proactive scheduling). Recognizing more (useful) contexts requires more information about the context, and thus more sensors and better recognition algorithms. As hardware sensors and their supporting processing components become cheaper, more reliable, efficient and smaller, they can be effortlessly embedded in clothes, small devices, furniture, etc.

The bottleneck in context awareness is not the hardware, but the algorithm that has to interpret the sensor data. An on-line adaptive algorithm to recognize contexts (as proposed in [2]) would result in a very efficient way to deal with these problems. Not

the designer of the application, but the user decides what contexts the system should distinguish, by labeling the incoming sensor data with a context description. The source of information often contains noise, and processing should be computationally inexpensive, so artificial neural networks are well-suited algorithms to tackle this problem.

2 An Adaptive Algorithm for Context Awareness

As already mentioned in the previous section, sensors should be as effective as possible. It is usually very easy to enhance raw sensor data by preprocessing simple features like standard deviation or mean to replace the raw values. The advantages of this approach are that the algorithm does not have to deal with time series, and throughput of data is considerably slowed down. The drawback is that some information can be lost due to a small interval in which the features are calculated or because of a lack of quality of features themselves. Features solve the throughput problem of input vectors, but also increase their dimension, which leads to a slower learning algorithm due to the *curse of dimensionality* [3].

Once the fast stream of raw sensor data has been filtered by calculating the features of each sensor, context recognition becomes more achievable. The objective is then to map the high-dimensional, possibly noisy, features to the most likely context description. Central to our approach is a Kohonen Self-Organizing Map (KSOM) [4] that builds up a 2D topological map using the input data.

2.1 The Kohonen Self-Organizing Map

The Kohonen Self-Organizing Map is based upon earlier work by von der Malsburg [5] and can be classified as a Winner-Take-All unsupervised learning neural network. It stores prototypes w_{ij} (also known as codebook vectors) of the input vectors at time t , $x(t)$, in the neurons (or cells) of a 2D grid. At each iteration, the neuron that stores the closest prototype to the new input vector (according to the Euclidean metric for instance) is chosen as the winner o_i :

$$o_i = \arg \min_j \left(\sqrt{\sum_{i=1}^n |x_i - w_{ij}|^2} \right) \quad (1)$$

The prototype vectors are usually initialized with small random values.

The winner neuron updates its prototype vector, making it more sensitive for latter presentation of that type of input. This allows different cells to be trained for different types of data. To achieve a topological mapping, the neighbors of the winner neuron can adjust their prototype vector towards the input vector as well, but in a lesser degree, depending on how far away they are from the winner. Usually a radial symmetric Gaussian neighborhood function h is used for this purpose:

$$\forall j : w_{ij}(t) = w_{ij}(t-1) + \mathbf{a}(t) \cdot \mathbf{h}(t) \cdot (x_i(t) - w_{ij}(t-1)) \quad (2)$$

The learning rate \mathbf{a} and neighborhood function \mathbf{h} decrease as the value of t' , the time that was spent in the current context, increases. The decrease rate is crucial in on-line clustering since it can lead to overwriting of prototype vectors on the map (known as the *plasticity-stability dilemma* [6]). Different neurons on the map will become more sensitive to different types of input as more input vectors are presented. Neurons that are closer in the grid tend to respond to inputs that are closer in the input space. In short, the KSOM learns a smooth mapping of the input space to neurons in the grid space. The map that gets created would be especially useful as visual feedback to the user of the network state, which is for us the main reason why the KSOM was used. Although more traditional methods yield more accurate results for clustering and multidimensional scaling [7], they were not found to be as suitable for on-line, real-time processing of sensor data as KSOM.

2.2 On-line K-Means Clustering and Classification

The k-means clustering algorithm is mainly used for minimizing the sum of squared distances between the input and the prototype vectors. However, it does not perform topological mapping like KSOM does. Most variants of the k-means clustering algorithm involve an iterative scheme that operates on a fixed number of k clusters to minimize the sum of Euclidean distances between the input and prototype vectors.

The original, on-line version (by MacQueen [8]) first finds the cluster with the closest prototype vector to the input, and then updates it just like KSOM, without further updating of neighboring clusters.

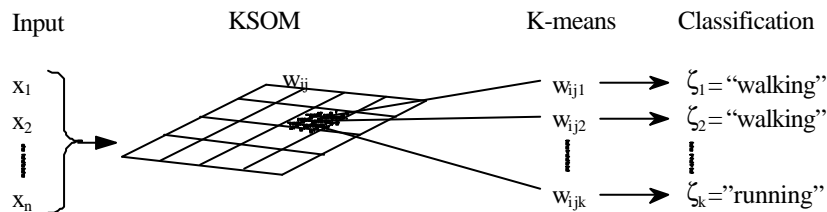


Fig. 1. Diagram of the algorithm. The KSOM clusters the input, while preserving the topology; labeled input vectors are clustered a second time by an on-line k-means algorithm.

Labeling the winner neuron with the label that comes with the input vector is a simple means of classification. In on-line mode, KSOM tends to eagerly overwrite previously learned data, especially in the initial stages (often referred to as the untangling phase). This is a main shortcoming, since the teacher decides when training is done (i.e. when classification labels are supplied with the input vectors). A KSOM neuron that was initially trained for one class, could after a while have its prototype vector overwritten and should belong to another class, or it could be labeled for two or more classes since its prototype vector has not converged yet to one class or another. These errors cause the KSOM to be highly unstable in the initial phase, and make it even possible to stay unstable afterwards.

To avoid overwriting, a second layer with on-line k-means clusters of labeled input vectors (see Figure 2) is introduced for every winning KSOM neuron. The label can

thus be found by first searching the winner and then retrieving the label of the closest k-means sub-cluster for that neuron. Only labeled input vectors are stored in this second layer and a small number is usually chosen for k , so the second layer does not cause a large computational overhead.

If the incoming data is labeled, the input vector x will be clustered to the k-means of the winning KSOM neuron:

$$w_{ijl}(t) = w_{ijl}(t-1) + \mathbf{I} \cdot (x_{ij} - w_{ijl}(t-1)), \text{ with } l \in [1, k] \quad (3)$$

The algorithm initializes the clusters by simply filling them with the first k inputs. Unlabeled data is classified by searching the k closest labeled cells on the KSOM. A vote among the k resulting labels determines the final classification.

3 Experiments and Results

It is hard to evaluate the exact performance of the algorithm, since many requirements have to be fulfilled. One way is to track the percentage of correct classifications as the experiment progresses. This allows verifying the stability of the algorithm and its convergence towards an acceptable recognition.

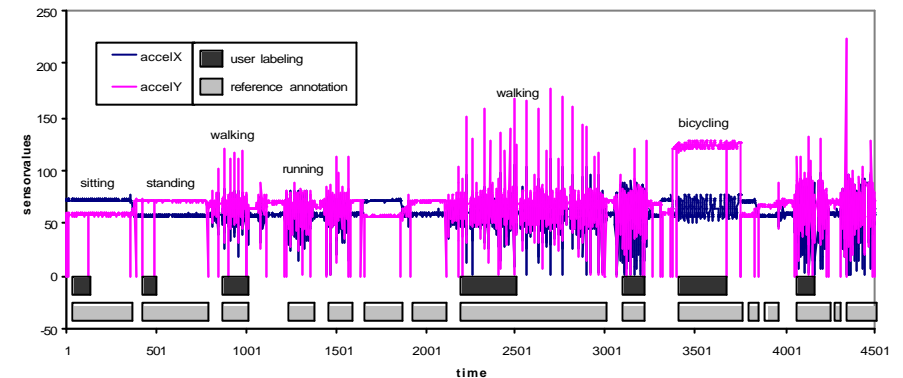


Fig. 2. Part of the raw sensor data stream, generated by two accelerometers. The black and gray boxes indicate the labeling by the user and by the experimenter after the recording.

The data from only two sensors was used in the experiments since the stability of the algorithm in the initial phase is the focus of this paper. Two accelerometers (sensors that measure motion and position) placed just above the knee of a person were used to recognize simple, everyday activities: sitting, standing, walking, running and bicycling. The data was labeled sporadically by the user during capturing. Afterwards most of the recorded data was annotated to enable an evaluation of the algorithm (see Figure 2). The minimum, the maximum, the mean and the standard deviation were recalculated over a sliding window of 3 seconds (or 100 data points) at one tenth of the speed the raw data is streaming in.

To illustrate the detrimental effect of overwriting in the KSOM, a 10×10 KSOM was used. Initial learning rate and neighborhood radius were set to $\mathbf{a} = 0.05$ and $\mathbf{h} = 2$,

decreasing linearly in function of the time that was spent in a particular context. Performance is quite good immediately after the KSOM is trained for a context (as can be seen in Figure 3), but it becomes overwritten as other contexts gets trained. Convergence occurs only after the winning KSOM neurons for a certain context are far enough apart on the map, and is slightly less qualitative (<5%) on the long run.

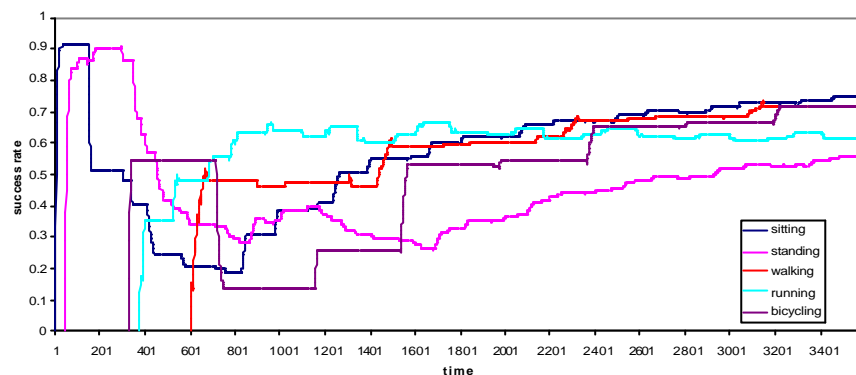


Fig. 3. Performance of a 10×10 Kohonen Self-Organizing Map (KSOM) with initial learning rate $\alpha=0.05$ and neighborhood radius $h=2$, decaying linearly per class (or context). Due to overwriting, initially learned concepts deteriorate which leads to slow convergence.

The same data was also used with the k-means layer added to the KSOM, as illustrated in Figure 1. The parameter k was set at a low 5, theoretically limiting the maximum number of clusters to be 500. This worst-case number was not met at all, however (up to 210 clusters were formed during the experiments). Figure 4 confirms that the performance is substantially better than the KSOM's, showing no traces of overwriting. The main reason for this is the hierarchical structure of the algorithm: the k-means sub-clusters refine the early KSOM clustering and secure data preservation, when the KSOM's topological mapping interferes with the clustering.

5 Conclusions

The typically large dimension of the input space, noisy signals, and the irregular training make context awareness a challenging problem to learning algorithms. This paper proposed an algorithm to cope with real-time processing of sensor data and on-line training, while the user can decide when the training should occur.

The Kohonen Self-Organizing Map can be very unstable, especially in the initial untangling stage when the parameters that control the flexibility are high. To avoid overwriting, KSOM was extended with a second layer of labeled on-line k-means sub-clusters. The k-means clustering assures rapid vector quantization and a better safekeeping of memorized prototypes, while the KSOM gradually creates its topological map. Experiments show that unlearning is indeed kept to a minimum, resulting in a stable topological mapping that is executed in real-time on sensor data.

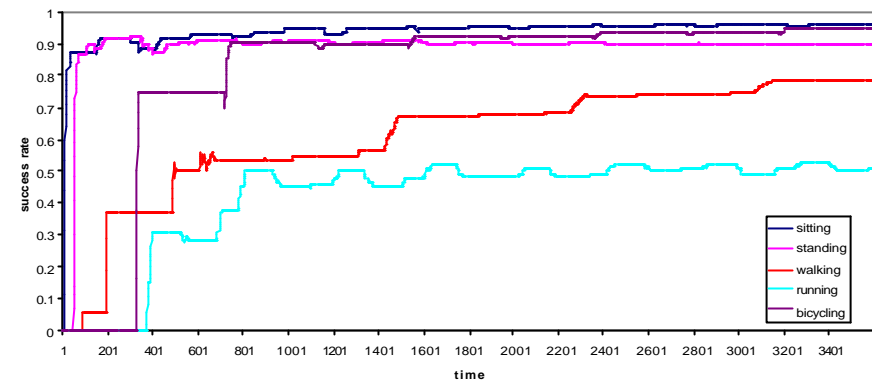


Fig. 4. Average performance over three experiments of a 10×10 KSOM, with identical parameters as in Figure 3, and a second k-means clustering layer with $k = 5$ and $\alpha = 0.2$.

Acknowledgements

Thanks to all people from Starlab, especially Erol Sahin and Ronald Schrooten for reviewing the drafts of this paper, and Steven Lowette for prototyping the software that was used during the experiments. The research described in this paper is partly funded by Starlab's i-Wear project, thanks go out to all partners for their support.

References

1. Abowd, G.D., Dey, A.K., Brotherton, J. Context Awareness in Wearable and Ubiquitous Computing. Proceedings of the First International Symposium on Wearable Computing (ISWC), Boston, MA: IEEE Press. (1997) 179-180
2. Van Laerhoven, K. & Cakmakci, O. What shall we teach our pants? Proceedings of the Fourth International Symposium on Wearable Computing (ISWC), Atlanta, USA: IEEE Press (2000) 77-84
3. Bellman, R. Adaptive Control Processes. Princeton University Press (1961)
4. Kohonen, T. Self-Organizing Maps. Springer-Verlag Berlin-Heidelberg (1994)
5. von der Malsburg. Self-organization of orientation sensitive cells in the striate cortex. Kybernetik. Vol. 14, Springer-Verlag (1973) 85-100
6. Grossberg, S. Adaptive pattern classification and universal recoding, I: Parallel development and coding of neural feature detectors. Biological Cybernetics 23. (1976) 121-134
7. Flexer, A. Limitations of Self-organizing Maps for Vector Quantization and Multidimensional Scaling, in Mozer M.C., et al.(eds.), Advances in Neural Information Processing Systems 9, MIT Press/Bradford Books, (1997) 445-451
8. MacQueen, J. On convergence of k-means and partitions with minimum average variance, Ann. Math. Statist. 36. (1965) 1084