

Machine Learning Miniproject 1

Vlad Balan

October 28, 2004

1 Introduction to SOMs

A Self-Organizing Map (SOM or Kohonen Map) is a method of mapping vectors of an n -dimensional space V into elements of a lower-dimensional subspace A having a discretized structure consisting of a lattice of so-called neurons, objects associated with vectors of the space V . We denote the with $r \in A$ elements of A and with $\mathbf{w}_r \in \mathbf{V}$ their associated vectors.

The SOM algorithm begins by assigning random values to the elements of A . During the training phase, the SOM is presented with sequences of training data vectors from V . Its purpose is to create a continuous structure embedded in V (an approximation of a smooth sub-manifold of V). In order to achieve this goal, neighboring neurons have to be associated with vectors of close values. A neighborhood activation function h_{rs} has to be selected, for example the Gaussian

$$h_{rs} = \exp\left(-\frac{|r-s|^2}{\sigma^2}\right)$$

with a variance σ^2 controlling the spread of the neighborhood. When a vector $\mathbf{x} \in \mathbf{V}$ is presented to the SOM, the neuron s for which the dot product $\mathbf{x} \cdot \mathbf{w}_s$ is maximal is selected, and the vectors of the neurons in its neighborhood are updated according to the formula:

$$\mathbf{w}_r^{(\text{new})} = (\mathbf{1} - \epsilon \cdot \mathbf{h}_{rs})\mathbf{w}_r^{(\text{old})} + \epsilon \cdot \mathbf{h}_{rs} \cdot \mathbf{x}$$

with the factor $\epsilon \ll 1$ commonly decreasing over time.

In order for this approach to function, we have to assume that the training data has been previously normalized.

The most common topologies in building the A lattice are the rectangular and the hexagonal ones.

In the best case the SOM algorithm leaves us with a proper sub-manifold. However, as discussed

in [2], depending of the actual dimensional structure of the pattern we are trying to approximate we can obtain pathological models, resembling space-filling curves or having several almost disconnected components.

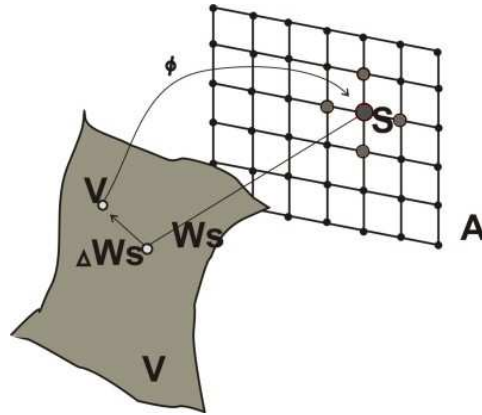


Figure 1: Rectangular Self Organizing Map[4].

Note: The presentation in this section was inspired by [2].

2 Problem description

Consider bitmap images of dimension 15 by 16 pixels containing handwritten representations of decimal digits. We are interested in correctly classifying these objects, i.e. in finding a computable mapping $C : \text{digits} \rightarrow \{0, \dots, 9\}$ providing a satisfactory degree of accuracy.

For training and testing purposes an array of 2000 such samples extracted from Dutch utilities maps will be used [1]. For each of the ten classes, the sample contains 200 objects, out of which half will be used for training our classifier and the other half will be kept aside for testing purposes. This

splitting will allow not only a good appreciation of the classifier’s performance, but also of the effect of the bias-variance dilemma on our choice of parameters.

The dataset is publicly available for algorithm comparison purposes.

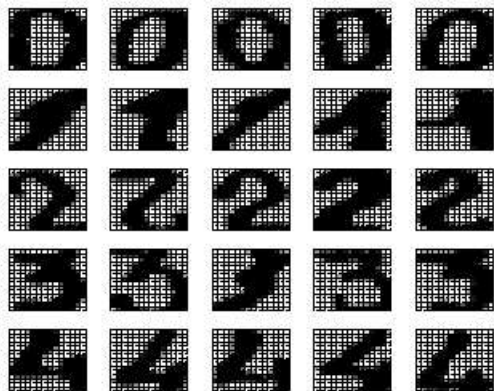


Figure 2: Samples of the dataset used for this classification problem [3].

3 Description of the approach

3.1 SOM classification

The implementation used the SOM supervised training algorithm, described in [6], which extends the usual training procedure by taking into account the class inclusion labels during the training period. The training vectors $x \in V$ receive c additional fields, where c is the number of different classes. If the vector x has been assigned to a certain class, then the corresponding field is set to the value 1, otherwise it is set to the value 0.

The map is constructed in the usual manner on this extended space V , with the advantage that elements of the same class will most likely be grouped closer by the training procedure since their distance after normalization decreased. Each element of the map is then assigned a class label by taking a maximum over these added components (the class representing fields), and, finally, the supplementary class information is removed.

In optimizing the SOMs performance, the following parameters had to be adjusted:

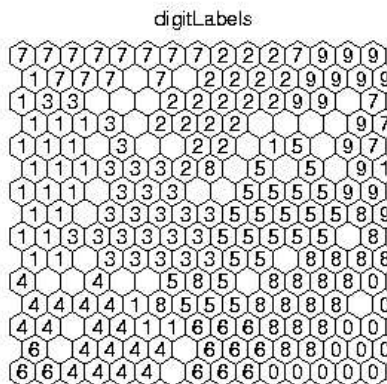
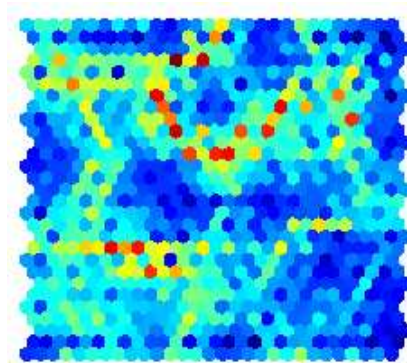


Figure 3: A color chart and class representation of an unoptimized 15 by 15 SOM [3].

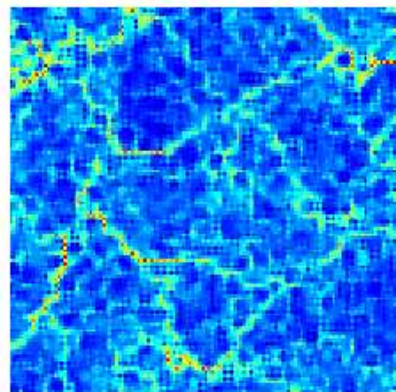


Figure 4: A color representation of an optimized 62 by 62 SOM.

- *the training algorithm:* Due to the nature of the problem the supervised training algorithm was selected. This choice brings an added complexity on the size of the SOM. While with the normal training algorithm the best results seem to be obtained with SOMs of a size situated somewhere between 25 and 30, with a supervised SOM this number most likely lies above 70. Due to limitations in the memory size of the hardware used, and to increasing computing time required, we had to stop experimenting around this value, although further results seemed to be promising.
- *the lattice topology* A rectangular topology was a better choice than a hexagonal topology since it provides fewer connections between elements, and therefore more elastic neighborhoods.
- *the lattice shape* The toroidal shape, as opposed to the sheet and cylindrical one, does not contain borders limiting connectivity but instead uses the full potential of the lattice. As a consequence, it appears that on lattices of this shape, the different clusters of vectors can differentiate better. Arguably, the problem has a certain resemblance to certain problems from electrostatics involving repelling charges, for which such a solution provides a minimal potential.

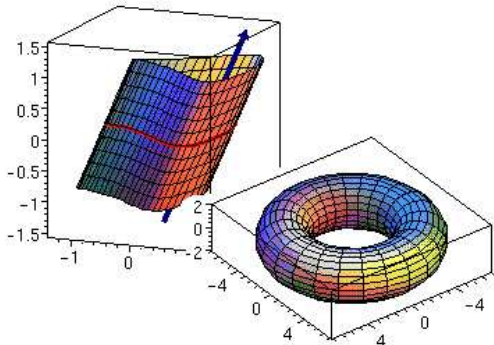


Figure 5: Different possible lattice shapes.[5]

- *the lattice dimension* This parameter was determined experimentally, and, as previously said, perhaps not yet optimal, to be around 70.

3.2 Post-processing

In improving the performance of the recognition algorithm, the main problems are caused by sets of data that do not adhere to the usual patterns. While such cases in the test data are hard to recognize and correct, we can try to compensate for the flaws present in the training data.

The usual classification method for vectors $\mathbf{x} \in \mathbf{V}$ using an SOM is based on selecting the class of the best-matching unit (BMU), i.e. the class label associated with the neuron $r \in A$ for which the product $\mathbf{x} \cdot \mathbf{w}_r$ is maximized.

The correction strategy presented in this case relies on the fact that for a vector \mathbf{w}_r of the correct class, the four elements in the immediate neighborhood most probably have the same class and are themselves the next best-matching units. Therefore, instead of selecting just one best-matching unit, the best five are selected and they are attributed the weights $5 \cdots 1$. Finally the class for which the sum of the weights of the elements belonging to it is maximal is selected.

This mechanism provides a simple solution for out-voting most likely wrongly matching elements. The decision on the linearly decreasing weights was taken after considering several possible scenarios of a mismatch occurring, and inferring in which way an out-voting was possible. This approach was not tried on a hexagonal topology.

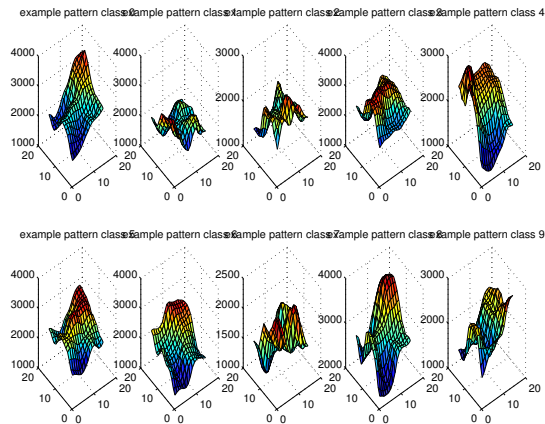


Figure 6: Response charts of SOM elements to class members [3].

4 Intermediate and final results

The unoptimized SOM presented in the example code yields on the test data a recognition rate of **91.1%**.

The optimized supervised SOM presented in the example code yields, without post-processing, on the test data a recognition rate of **97.4%**.

The post-processing algorithm on the example code improves its performance on the test data to **94.1%**.

The optimized supervised SOM presented in the example code yields, with post-processing, on the test data a recognition rate of **97.6%**. This might make it seem that preprocessing does not work as well in this case, but the fact that the error margin to be reduced is much smaller has to be taken into account.

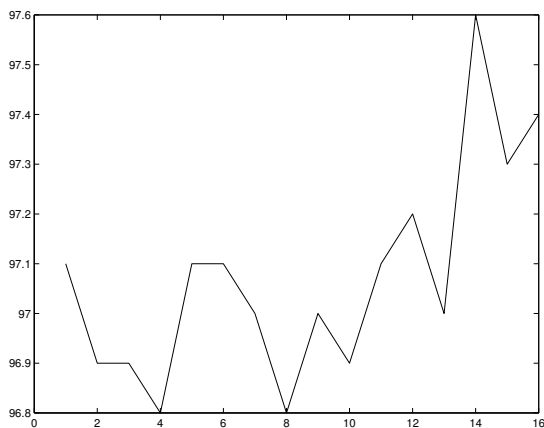


Figure 7: Recognition percentages for sizes 55 to 70 of the SOM.

5 Problems encountered

A constant care was taken in the algorithm adjustment phase to the effects of the bias-variance dilemma. For this purpose, the results of test runs on both training data and test data had to be considered as a bi-dimensional performance measure. Especially in evaluating post-processing methods, this combined measure helped in avoiding a number of pitfalls.

The fact that no pre-processing of the data for feature extraction was allowed made it impossible to use any previous knowledge of the data structure in building the classifier. An interesting feature would have been for example a matching function between the colors of different parts of the image (with an example rule of the type: activate if the upper half is light and the lower half is dark). We know that such feature extraction techniques have been successfully used in the construction of state of the art recognition algorithms targeted on the same problem.

The essential problem encountered was the limitation in computing capacity (mainly memory), which declined us the possibility of investigating the full potential of this methods. By using the CLAMV machines (which provided a double quantity of physical memory) we could push this limit somehow further, but it seems that from this points on this technique becomes computationally too expensive.

References

- [1] Robert P.W. Duin, David M.J. - Experiments with Classifier Combining Rules, 2000
- [2] Helge Ritter - Self-Organizing Feature Maps: Hohonen Maps, 1998
- [3] Miniproject 1 Handout and Guidelines
- [4] <http://www.zalf.de/lasad/persweb/wieland/welt/Pictures/KOHONEN1a.png>
- [5] http://orange.math.buffalo.edu/241/cylinder_sphere_and_torus.html
- [6] The SOM Toolbox documentation.