Unsupervised clustering with growing self-organizing neural network – a comparison with non-neural approach

Martin Hynar, Michal Burda, and Jana Šarmanová

Department of Computer Science, VŠB – Technical University of Ostrava
17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic
{martin.hynar, michal.burda, jana.sarmanova}@vsb.cz

Abstract. Usually used approaches for non-hierarchical clustering of data are well known k-means or k-medoids methods. However, these fundamental methods are poorly applicable in situations where number of clusters is almost unpredictable. Formerly, they were adapted to allow splitting and merging when some defined criterion is met. On the other hand there are also methods based on artificial neural networks concretely on self-organizing maps. One of the interesting ideas in this domain is to allow growing of the net which corresponds to adapted k-means method. In this article we are going to compare both approaches in a view of ability to detect clusters in unknown data.

Key words: data mining, cluster analysis, partitioning algorithms, competitive learning, self-organizing map, growing neural network.

1 Introduction

In common practice of various domains, e.g. pattern recognition (recognition of objects in range of data, letters), information retrieval (grouping documents, looking for common topics), data mining (searching for interesting patterns in arbitrary data sets) there could be used as a valuable tool some clustering technique. The most widely used techniques are mainly *k-means* based methods (see [5], [3], [1]), which are easy to use and obtained results are fairly understandable. Nevertheless, these methods are too static in a particular point of view; at the beginning we have to specify the number of expected clusters and the algorithm is then responsible to find them in the data set. But, what if we could not determine this number? There are two alternatives to solve such problem.

- Firstly, we can do multiple computations with varying number of expected clusters. Such approach is admissible only in situations where input data set is not to extensive; on large data it will be very time consuming. Moreover, the decision what partitioning is appropriate is then based on a subjective estimation.

K. Richta, V. Snášel, J. Pokorný (Eds.): Dateso 2005, pp. 58–68, ISBN 80-01-03204-3.

- Second solution is in adaptation of the algorithm where it will be allowed to split and/or merge clusters according to some predefined condition. Usually, clusters are splitted if the inter-cluster variability increases over some threshold and merged if the typical points of neighbouring clusters are near enough. We can also ignore insufficiently numerous clusters.

The second solution was used for example in algorithms like ISODATA (see [7], [4], [5]) or CLASS (see [7].

On the other hand, we can use a model of artificial neural network based on competitive learning known as the self-organizing map (SOM) or Kohonen map (see [6] or [8]). It is known that fundamental feature of SOM is to preserve data topology. So, neurons of the map are likely to occupy the place in the input space where more dense places are situated. The basic model of SOM consists of neurons whose quantity is specified in advance. That is, with this model we are able to discover only a predefined number of clusters.

Like in previous case also SOM was adapted to allow the topology grow. One of the usable approaches is the *Growing neural gas* (see [2])

In this article we first re-introduce the *k-means* algorithm in section 2 and one of its derivative (CLASS method) allowing adaptation of the number of clusters in section 3. In the section 4 we focus on fundamentals of SOM and on brief introduction of *Growing neural gas* algorithm in section 5. In the 6 there are provided some experiments with the comparison of the results obtained with both types of methods.

2 *k*-means based methods

So called *partitional* clustering methods could be stated as "given N patterns in n-dimensional space, determine the partition of the patterns into k groups where patterns in the same group are more similar than patterns from different groups" The notion of the patterns similarity have to be adopted in advance and different algorithms use different ones.

Moreover, the issue of determining the appropriate k is not decidable in all situations. If there exist some general perspective over clustered data, we can use a fixed value of k. The task of a partitional algorithm is then to locate clusters in the input space. If we are not able to determine k at the beginning, we could use trial-and-error way with modifying clustering parameters. However, there are also methods trying to locate clusters in the input space and to determine the number of such clusters at a time.

An algorithm, generally known as k-means clustering is the one where the number of expected clusters have to be given as the clustering parameter. Such algorithm given the set of patterns then tries to locate k clusters.

Choose typical points:

Place k typical points of clusters according to chosen method into the multidimensional space containing examined patterns.

Clustering:
Assign each pattern to exactly one typical point – to the nearest one.

Recompute typical points:
Using all patterns assigned to particular cluster recompute its typical point as the mean value.

Check termination condition:
The computation ends if the termination condition is fulfilled. Typical condition is that there are no or minimal changes in cluster memberships.

The problem of initial placing of k typical points could be solved using several techniques. The simplest ones are choosing random points or randomly chooses k input points.

Each pattern of the examined data set is then assigned to some typical point. The appropriate one is determined as the one with the smallest Euclidean distance.

$$d_E(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

When all patterns are processed the new typical points should be computed. New typical point of a cluster is determined as a mean vector of patterns in the group. The end of the algorithm usually becomes when no change in cluster membership occurs in two subsequent iterations.

3 CLASS – where clusters may arise and disappear

A clustering method CLASS is inspired in former method ISODATA which is in comparison with k-means method able to refine the number of clusters during the computation but it has some crucial disadvantages, mainly that many parameters need to be set by user. Such approach usually leads to incorrect results if data are not understood properly. On the other hand, the CLASS method in most cases determines the parameters from the input data.

The CLASS method proceeds in few steps. At the beginning, user has to set three input parameters: maximum number of iterations GAMMA, minimum number of members in a cluster THETAN and the initial splitting threshold S_0. For the testing purposes we used a modified version of CLASS method where it is also possible to set the initial number of typical points K and choose arbitrary method for placing them in the input space.

The very first step in the clustering is assigning each pattern to exactly one of the typical points. This proceeds using the k-means algorithm described above. After this the CLASS method goes through three following steps:

Excluding small clusters:
All clusters with less than THETAN members which were not changed during last two iterations are excluded from the subsequent analysis.

Splitting clusters:

In the m^{th} iteration we have to compute the splitting threshold s_m

$$S_m = S_{m-1} + \frac{1 - S_0}{GAMA}$$

In each cluster we need to compute deviations (d_{ij}) between the typical point and each pattern belonging to this typical point. Then, for the j^{th} attribute we can compute the average deviations D_{j1} of patterns situated on the right and D_{j2} of the patterns situated on the left side.

$$D_{j1} = \frac{1}{k_1} \sum_{i=1}^{k_1} d_{ij}, \qquad D_{j2} = \frac{1}{k_2} \sum_{i=1}^{k_2} d_{ij},$$

where k_1 and k_2 are the numbers of points situated on the right and on the left side of the typical point within given attribute. Now we are ready to determine parameters controlling the partitioning of the clusters. For each cluster we compute parameters a_1 and a_2 for patterns on the right side and left side.

$$a_1 = \max_j \left(\frac{D_{j1}}{max\, x_{ij}} \right), \qquad a_2 = \max_j \left(\frac{D_{j2}}{max\, x_{ij}} \right)$$

where $j = 1, 2, \ldots, p$ and i denotes all points from the same cluster. If in m^{th} iteration holds: number of clusters is less than $2K$, $a_1 > S_m$ or $a_2 > S_m$ and number of processed patterns greater than $2(THETAN + 1)$ then we split the cluster with respect to attribute j where a_1 or a_2 is maximal. The newly created clusters contain patterns from the right and left side of the typical point within the j^{th} attribute.

Revoking clusters:

To decide which cluster has to be revoked we need to compute average minimum distance of clusters in advance.

$$TAU = \frac{1}{h} \sum_{i=1}^{h} D_i$$

where h is current number of clusters and D_i is the minimum distance of i^{th} typical point from the other ones. If $D_i < TAU$ for some cluster i and number of clusters is greater than $\frac{K}{2}$ we revoke i^{th} cluster. Patterns belonging to revoked cluster are dispersed to the nearest typical points.

These steps proceed until clusters remaining unchanged or maximum number of iterations GAMA is reached.

4 Self-organizing map expresses the topology

The self-organizing map (SOM) is an artificial neural network based on an issue of competitive learning. The net consists of a set $\mathcal{A}$ with n neurons, represented

with weight vectors $\boldsymbol{w}_i$. Furthermore, neurons are mutually interconnected and these bindings form some topological grid (usually rectangular or triangular). If we present a pattern $\boldsymbol{x}$ into this network then exactly one neuron could be the *winner* and its weights are adapted proportionally to the pattern (the neuron is then closer). Moreover, neurons from the neighbourhood of the winner are adapted too, but not so intensively. Neighbourhood $N(c)$ could be formally defined as set of neurons that are topologically near to the winner.

The winner of the competition is determined as the neuron with the minimum distance to the pattern.

$$c = arg \min_{a \in \mathcal{A}} \{||\boldsymbol{x} - \boldsymbol{w}_a||\} \tag{1}$$

Then, adaptation of the weights proceeds using the equation (2) generally known as *Kohonen's rule*.

$$w_{ji}(t+1) = \begin{cases} w_{ji}(t) + h_{cj}(t)(x_i(t) - w_{ji}(t)) & j \in N(c) \\ w_{ji}(t) & \text{otherwise.} \end{cases} \tag{2}$$

Weight vectors for the next iteration $t+1$ of the winner and neurons in the neighbourhood are adapted in a way that current weights are modified (either added or subtracted) with a variance of current weight and input pattern.

Parameter $h_{cj}(t)$ is usually represented with unimodal Gauss function with center in c, width $\sigma(t)$ and maximal unit movement $h_0(t)$. Values of $\sigma(t)$ and $h_0(t)$ are decreasing in time – this corresponds with rough learning in the beginning and fine learning later.

$$h_{cj}(t) = h_0(t)\, exp\Big(- \frac{||\boldsymbol{w}_c - \boldsymbol{w}_j||^2}{2\sigma^2(t)} \Big)$$

One of the features of SOM is its topology preserving behaviour. This means, that SOM tries to adapt weights of neurons to cover the most dense regions and therefore SOM naturally finds data clusters. The limitation of SOM lies in fact that it is designed to have number of neurons specified as the input parameter and immutable during the learning process.

5 A self-organizing map that grows

The *Growing Neural Gas* (GNG) method [2] is the modification of the SOM where number of neurons is not immutable input parameter but is changed during the competition. Connections between neurons are not permanent as well. The result of competition could be then set of separate neural networks covering some region of the input data.

In the beginning the network itself contains only two neurons a_1 and a_2 representing two randomly chosen input patterns. Denote set of neurons as A and set of connections as $\mathcal{C}$ which is empty in the beginning.

Competition

The pattern $\boldsymbol{x}$ is presented to the network. The winner s_1 of competition and the second nearest s_2 neurons are determined using equation (1). If there was not a connection between neurons s_1 and s_2 then it is created ($\mathcal{C} = \mathcal{C} \cup \{(s_1, s_2)\}$). The *age* of the connection is set or updated to 0 ($age(s_1, s_2) = 0$). The squared distance between the winner and the pattern is added to local error variable.

$$\Delta E_{s_1} = ||\boldsymbol{x} - \boldsymbol{w}_{s_1}||^2$$

Adaptation

The weight vectors of the winner and its direct topological neighbours[1] N_{s_1} are adapted by fractions ϵ_b and ϵ_n of the distance to the input pattern. This is analogous to the *Kohonen's rule* (equation (2)) described above.

$$\begin{aligned} \Delta w_{s_1} &= \epsilon_b(\boldsymbol{x} - \boldsymbol{w}_{s_1}) \\ \Delta w_i &= \epsilon_b(\boldsymbol{x} - \boldsymbol{w}_i) \qquad \forall i \in N_{s_1} \end{aligned}$$

The age of all connections leading from the winner neuron are increased by 1 ($age(s_1, i) = age(s_1, i) + 1$ for all $i \in N_{s_1}$).

Removing

If there exist some connections with age greater than given a_{max} then all are removed. If this step results in neurons with no connections then remove also these standalone neurons.

Inserting new neurons

If the number of processed patterns reached an integer multiple of given parameter λ then new neuron is inserted using following steps:

1. First of all, the neuron p with the largest accumulated local error is determined using following equation.

$$p = arg \max_{a \in \mathcal{A}} \{E_a\}$$

   Among the neighbours of neuron p determine neuron r with largest accumulated local error.
2. Insert new neuron q to the network ($\mathcal{A} = \mathcal{A} \cup \{q\}$) and set its weight to the mean value of p and r weight vectors.

$$\boldsymbol{w}_q = \frac{1}{2}(\boldsymbol{w}_p + \boldsymbol{w}_r)$$

3. Insert new connection between new neuron q and neurons p and r ($\mathcal{C} = \mathcal{C} \cup \{(p, q), (r, q)\}$). Remove the old connection between neurons p and r ($\mathcal{C} = \mathcal{C} - \{(p, r)\}$).

[1] Note, neuron i is in direct topological neighbourhood of the winner c if there exists connection (i, c) between these two neurons.

4. Local accumulated error variables of neurons p and r are decreased by given fraction α.

$$\Delta E_p = -\alpha E_p \qquad \Delta E_r = -\alpha E_r$$

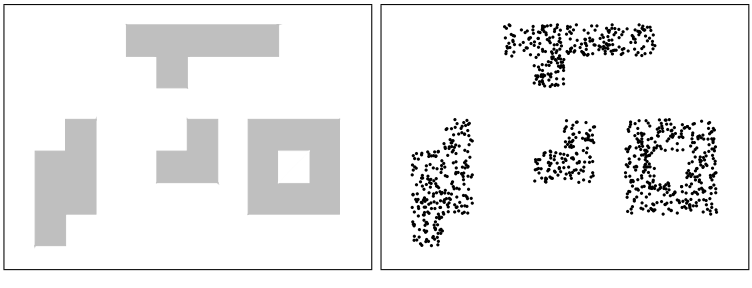
   The accumulated error variable of the new neuron q is set to the mean value of neurons p and r accumulated error variables.
5. Local accumulated error variables of all neurons in the network are decreased by given fraction β.

$$\Delta E_a = -\beta E_a \qquad \forall a \in \mathcal{A}$$

These several steps proceed until pre-defined termination condition is met. This could be some performace measure or usually net size.

6 Examples and comparison

As an example data for the subsequent clustering we will use data determined with distribution depicted in figure (1(a)). The highlighted regions are those where data are located and white regions contain no data. For the purposes of clustering using k-means method and CLASS method we have to generate the points complying given distribution in advance. A set of 1000 points from given distribution is depicted in (1(b)). For the usage with the SOM and GNG methods we may use continuous pseudo-random generator of patterns from the given distribution[2].

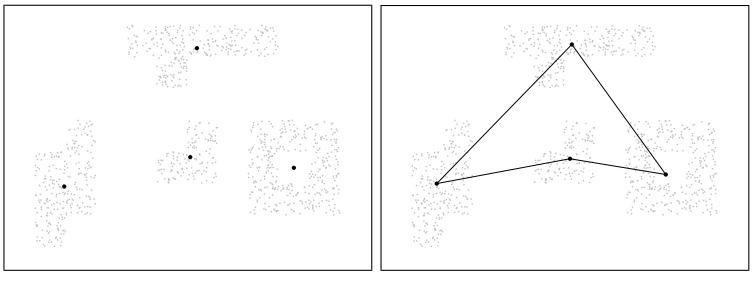


(a) Distribution (b) Objects from distribution

Fig. 1. Sample data for clustering.

[2] The same generator with the same seed we have also used to obtain set of discrete patterns for "classical" clustering. Without injury on generality we may say that explored sets of patterns were practically same.

First of all we have tested if k-means algorithm will produce similar partitioning as SOM neural network. If the number of typical points and number of neurons are set to actual number of clusters in data then both methods will place its representatives to the centers of clusters. Results of clustering using both methods with $K = 4$ are depicted in figure (2).



(a) k-means (b) SOM

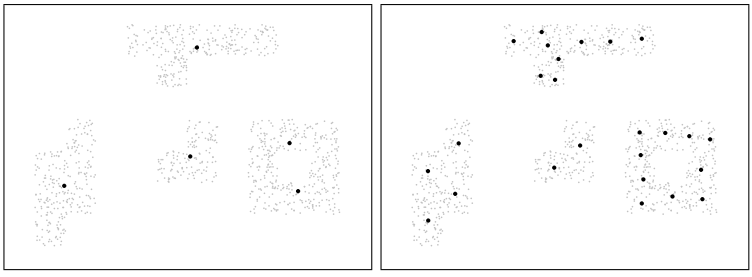
Fig. 2. Clustering using k-means algorithm and SOM neural net with number of representatives set to 4.

If the number of representatives is very slightly different from the actual number of clusters then obtained results are hardly interpretable. If the number is lesser then some representatives are trying to cover more clusters. If the number is greater then extra representatives will join another representative and they will cover one cluster with some more dense areas. The latter case may be incorrectly explained as presence of more clusters. In fact, representatives started to explain topology of clusters and besides finds more fine grained clusters. Figure (3) depicts situation where K was set to value 5 and 25.

It is clear that SOM is able to discover clusters in the input space as well as the k-means method.

Now we can focus on comparing the methods where splitting and merging of clusters is allowed. In case of neural net approach we may talk about net growth. One of the basic disadvantages of SOM is the inability to adapt its size according to proportions in the data (like k-means). On the other side, GNG can either add neurons where the local error is to large or remove neurons that have no relation to the rest (no existing connection).

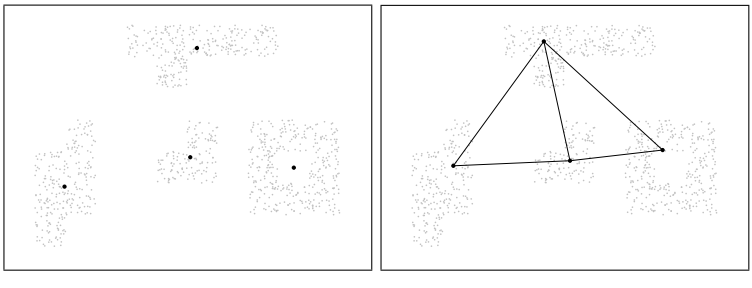
The issue we are interested at this time is how the CLASS method and the GNG method will proceed in clustering given data. Both methods are starting with two representatives. The CLASS method allows to add more representatives in one iteration in contrast to GNG which adds neurons after the pre-defined chunk of steps passes. So, we compared obtained partitionings at moment where both methods had identical number of representatives.



(a) k-means with $K = 5$ (b) k-means with $K = 25$

Fig. 3. Clustering using k-means algorithm with number of representatives set to 5 and 25.

The very first comparison was taken when both methods reached 4 representatives (the situation is depicted in figure (4)). As it is predictable, representatives are placed nearby the centers of the clusters to cover them as a whole.

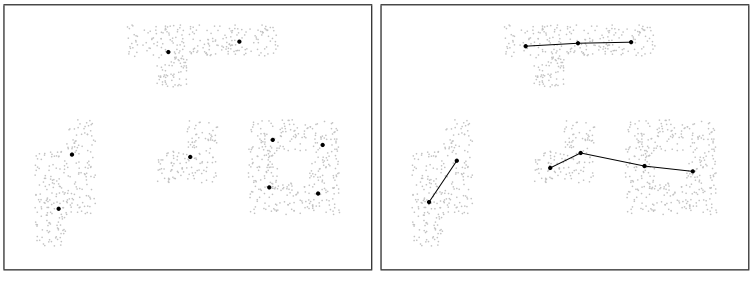


(a) CLASS (b) GNG

Fig. 4. Clustering using CLASS and GNG algorithms - results with 4 representatives.

More interesting results were reached when both methods started co cover clusters at the finer level. With 9 representatives we can distinguish little different behaviour of both methods. The dislocation of representatives obtained with CLASS method can be seen as dislocation of centers of circles with similar perimeter in an effort to cover all patterns – spherical clusters. On the other side the GNG proceeds with an extended goal. The GNG algorithm is covering all patterns with the neurons and moreover it covers the cluster's topology with the connections. We may see that in the figure (5(b)) there are three groups of interconnected neurons. We may interpret this situation for example as "there are three clusters with topology given by connections", but it is not so defini-

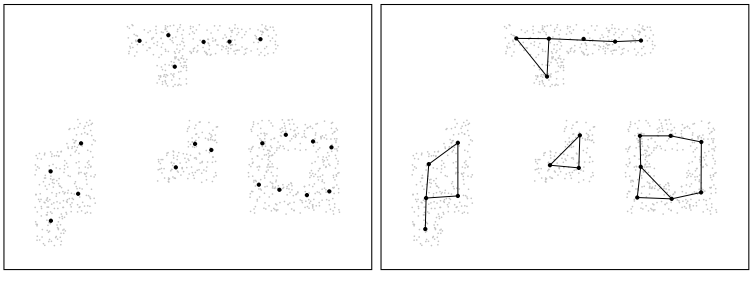
tive. In fact, there are four clusters but the GNG method does not discover them appropriately. Nevertheless, 9 representatives is still too little.



(a) CLASS (b) GNG

Fig. 5. Clustering using CLASS and GNG algorithms - results with 9 representatives.

Much more interesting results we can see when the number of representatives reaches value 21. The dislocation of representatives obtained by both methods is very similar and representatives cover all clusters effectively enough – they express the topology. But, there is remaining one more question – what is the resulting partitioning? In case of the CLASS method we have a set of non-related representatives. In 2-dimensional space we can do some visualisations to decide but in multi-dimensional space is the resulting partitioning uncertain. The problem lies in fact that the number of clusters is not the only thing we want to know. We need to know something more, it is how the clusters look like and how to formally express the category the cluster represents. Another big issue is the correctness of results obtained with these methods on multi-dimensional data.



(a) CLASS (b) GNG

Fig. 6. Clustering using CLASS and GNG algorithms - results with 21 representatives.

The connections between neurons of GNG are very helpful in this point of view. They inform about which neuron belongs to which cluster. From the figure (6(b)) we may decide on existence of 4 clusters and using the connections we can also state some conclusions on their shape or topology. But note, in the network could remain some edges that connect two neighbouring not so distant clusters. Thus, the results obtained with GNG method have to be evaluated carefully.

7 Conclusions

From the comparisons between k-means method and SOM neural network and between the CLASS method and GNG neural network we see that utilizing neural networks (with competitive learning) is good idea in the clustering domain. The most important feature of such neural networks is their natural ability to find dense areas in the input space. The extension of the basic SOM algorithm to dynamically reflect relations in the data (possibility of the net growth) makes neural networks even much more interesting.

The results obtained using both types of methods show that neural networks are able to give at least the same results. Moreover, the fact that SOM-like neural networks are likely to preserve the topology can mean that the results could be even better. Nevertheless, as using any other method for arbitrary knowledge discovery, the results have to be interpreted very carefully to be correct. This holds also in this case.

References

1. Everitt, B. S., Landau, S., and Leese, M. *Cluster analysis.* Oxford University Press, 2001.
2. Fritzke, B. A growing neural gas network learns topologies. *Advances in neural information processing systems 7* (1995).
3. Han, J., and Kamber, M. *Data mining: Concepts and techniques.* Morgan Kaufmann Publishers, 2000.
4. Jain, A. K., and Dubes, R. C. *Algorithms for clustering data.* Advanced reference series. Prentice hall, 1988.
5. Jain, A. K., Murty, M. N., and Flynn, P. J. Data clustering: A review. *ACM Computing Surveys 31*, 3 (September 1999), 264 – 323.
6. Kohonen, T. *Self-organizing maps.* Springer Verlag, 1984.
7. Lukasová, A., and Šarmanová, J. *Metody Shlukové Analýzy.* SNTL, 1985.
8. Šíma, J., and Neruda, R. *Teoretické otázky neuronových sítí.* Matfyzpress, 1996.