

Unsupervised Learning with the Soft-Means Algorithm

Chris Thornton

Cognitive and Computing Sciences
University of Sussex
Brighton
BN1 9QH
UK

Email: Christopher.Thornton@firenet.uk.com

Tel: (44)1273 678856

May 21, 2003

Abstract

This note describes a useful adaptation of the ‘peak seeking’ regime used in unsupervised learning processes such as competitive learning and ‘k-means’. The adaptation enables the learning to capture low-order probability effects and thus to more fully capture the probabilistic structure of the training data.

Relevant areas: automated discovery, neural networks

1 Introduction

Unsupervised learning involves discovering the underlying structure of a dataset without knowing how the individual data items are classified. This has been recognized as a hard problem in machine learning [1]. Many unsupervised learning methods operate by trying to find the ‘prototypes’ of the dataset. The usual approach here is to search for the density peaks in the distribution of training data. This can be done by explicit clustering [2], say, or by some iterative method such as competitive learning [3] or k-means clustering [4, 5].

Methods which seek out density peaks effectively sample the probabilistic structure of the data. However, they are only sensitive to, and can therefore only exploit, the n th-order structure of the data (where n is the number of inputs variables), i.e., the probabilities associated with *complete* data items. However, there are other aspects of the probabilistic structure which may be captured. In particular, there are the various m th-order probabilities (where $m < n$, $m > 0$), all of which may appear in both conditional and unconditional forms.

Consider the following dataset:

x1	x2	x3	x4	x5
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	1	0	0
1	0	1	0	1
1	0	1	1	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	0	1
1	1	1	1	0
1	1	1	1	1

This is based on five binary variables, x_1 , x_2 , x_3 , x_4 and x_5 . The dataset shows very little n th-order structure, which is to say there are no obvious density peaks. This is confirmed by the dendrogram produced from a conventional, hierarchical clustering process (see Figure 1) applied to the training data. The leaf nodes in the dendrogram are the numbers (subscripts) of individual data items and the internal nodes are the maximum distances for items in the relevant cluster. Note the general homogeneity of the structure and, in particular, how all the internal distances [6] for the initial clusters are the same.

Of course, the apparent absence of density peaks in a dataset does not mean that it has no probabilistic structure. In the present case, variables x_2 and x_5 are, in fact, conditionally related, as are variables x_1 and x_3 , since

$$\begin{aligned}P(x_2=1 | x_5=1) &= 1 \\P(x_1=1 | x_3=1) &= 1\end{aligned}$$

and

$$\begin{aligned}P(x_5=1 | x_2=1) &= 1 \\p(x_3=1 | x_1=1) &= 1\end{aligned}$$

In addition to these conditional effects there are various unconditional effects to be taken into account. For example, $P(x_1=1)$ substantially exceeds the chance value for a binary variable. These conditional and unconditional m th-order probabilities are not necessarily reflected in the n th-order statistics and therefore are not exploited by methods that looks solely for density peaks. However, they form an important aspect of the probabilistic structure of the data and may be vital for purposes such as prediction, data compression or classification.

function so that the response of a centre to an input is

$$- \sum W_i(|X_i - V_i|)$$

and arrange for the weight vectors to be updated at the end of each training epoch so that each W_i moves towards the normalized accuracy of the corresponding V_i . The unnormalized accuracy value for the i th component of a particular centre is just

$$\sum |X_i - V_i|$$

With this adaptation in place, the confidence weights for each centre are adapted as the centres converge towards the density peaks. Due to the effective decay of confidence weights on low-quality peak components (i.e., components providing relatively inaccurate estimates of input values) the process is able to discover density peaks in subspaces of the input space. This enables it to capture m th-order probability effects in an n -dimensional input space ($m < n$).

A graphical illustration of the adapted learning process is provided in Figure 2. The top box in this figure represents a 3-dimensional input space. The small circles are data points (training inputs) and the Xs are the centres. Within a conventional peak-seeking regime, the relatively compact clusters might be expected to successfully attract centres to their peaks. However, the cluster situated in the front-left of the space, which is distributed uniformly in the vertical dimension, would pose a problem since it has no obvious density peak.

The adapted competitive learning regime copes gracefully with this scenario. The vertical distribution of the problematic cluster would produce low accuracy values for the ‘vertical’ component of any nearby centre. This would lead to the weight on that component decaying which would, in turn, lead to the cluster being ‘compressed’ vertically, giving it a more distinct peak. The general form of this process is illustrated in the middle and bottom boxes in Figure 2.

3 Applications

I use the term ‘soft-means algorithm’ to label the modified peak-seeking regime described above. The following two examples illustrate the behaviour of this algorithm. In the first example the algorithm is applied to the training data from above:

x1	x2	x3	x4	x5
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	1	0	0
1	0	1	0	1
1	0	1	1	0
1	0	1	1	1

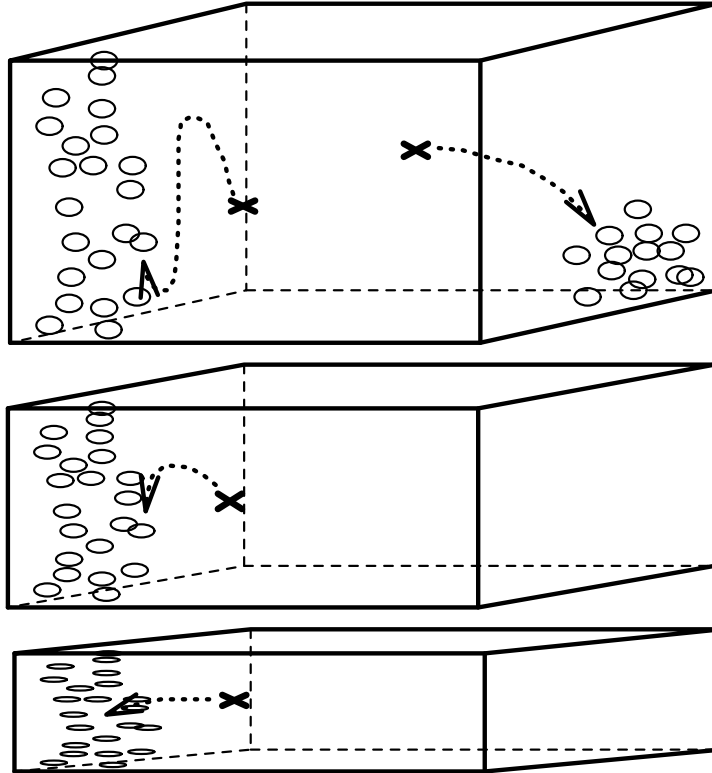


Figure 2:

```

1 1 0 0 1
1 1 0 1 1
1 1 1 0 0
1 1 1 0 1
1 1 1 1 0
1 1 1 1 1

```

The behaviour of the algorithm is illustrated by the listing below. This shows the behaviour of the centres through the first ten epochs of processing.¹ Each segment of the listing is headed with the relevant epoch number and contains just two lines. The first line shows the state of centre 1 (in the relevant epoch). The second line shows the state of centre 2. The bracketed part of each line shows the components and weights of the relevant centre. Each colon separates one component/weight pair. The component value appears before the colon; the weight comes after. Both values are real numbers in the range 0-1 but only the first two decimal places are shown. (1.00 always appears as '99').

¹An epoch here is simply a complete sweep through all the examples in the training set.

Epoch 1
1 {99:50 99:50 00:50 99:50 99:50}
2 {00:50 99:50 99:50 00:50 99:50}

Epoch 2
1 {90:56 99:67 20:64 87:53 99:67}
2 {79:65 70:52 97:73 43:56 54:56}

Epoch 3
1 {65:41 99:81 19:67 72:47 99:81}
2 {94:76 72:35 99:89 56:39 57:30}

Epoch 4
1 {53:33 99:90 28:54 60:32 99:90}
2 {99:89 71:20 99:95 60:23 56:18}

Epoch 5
1 {50:24 99:95 30:38 57:22 99:95}
2 {99:95 71:12 99:98 61:14 56:11}

Epoch 6
1 {49:16 99:97 31:25 56:15 99:97}
2 {99:98 71:08 99:99 61:09 56:07}

Epoch 7
1 {49:11 99:98 31:17 56:11 99:98}
2 {99:99 71:05 99:99 61:06 56:05}

Epoch 8
1 {49:07 99:99 31:11 56:08 99:99}
2 {99:99 71:03 99:99 61:04 56:03}

Epoch 9
1 {49:05 99:99 31:08 56:06 99:99}
2 {99:99 71:02 99:99 61:02 56:02}

Epoch 10
1 {49:03 99:99 31:05 56:04 99:99}
2 {99:99 71:01 99:99 61:02 56:01}

Note how the two centres rapidly ‘capture’ the two main probabilistic effects in the training data, namely

$$\begin{aligned} P(x_2=1 | x_5=1) / P(x_5=1 | x_2=1) &= 1 \\ p(x_1=1 | x_3=1) / P(x_3=1 | x_1=1) &= 1 \end{aligned}$$

4 Supervised learning with soft-means

The soft-means algorithm's sensitivity to low-order probabilistic effects often enables it to reproduce the functionality associated with *supervised* learning processes. To elicit supervised learning from soft-means, we form a set of vectors by appending each input from the relevant training set to its corresponding output. We then feed these to the algorithm as pure input vectors. To test the algorithm's 'output' on a test input, we present the algorithm with the test input appended to a null output (which is ignored by the algorithm) and then return the output component from the most strongly responsive unit.

In many cases, this approach enables the algorithm to produce performance on supervised learning problems comparable to fully supervised algorithms. For example, consider the algorithm's performance on the benchmark problem known as the 'third MONKS problem' [7]. The underlying rule for this problem is as follows: (attribute_5 = 3 and attribute_4 = 1) or (attribute_5 != 4 and attribute_2 != 3), with != denoting inequality.

A small sample of the training set (which contains 5% misclassifications) is as follows.

```
1 1 1 1 1 2 --> 1
1 1 1 1 2 1 --> 1
1 1 1 1 2 2 --> 1
1 1 1 1 3 1 --> 0
1 1 1 1 4 1 --> 0
1 1 1 2 1 1 --> 1
1 1 1 2 2 2 --> 1
1 1 1 2 4 2 --> 0
1 1 2 1 2 2 --> 1
1 1 2 1 4 2 --> 0
1 1 2 2 2 2 --> 1
1 1 2 2 4 1 --> 0
```

The initial two centres generated by the soft-means algorithm applied to these training data are as follows.

```
Epoch 2
1 {98:83 93:72 83:38 51:35 62:27 69:12 00:99}
2 {98:93 28:62 50:26 76:50 56:59 60:23 99:98}
```

Using the output-generation procedure described above, these produce a classification accuracy on the testing data of 87%. This is comparable to the accuracy produced by several of the supervised algorithms in the original MONKS study. [7]

5 Summary

The paper has described an enhancement of the ‘peak-seeking’ regime for unsupervised learning. As far as the author has been able to ascertain, this enhancement has not yet been investigated by the community. Its main advantage is that it enables the regime to capture probabilistic structure involving low-order probabilistic effects (i.e., effects which impact sub-spaces of the input space). A secondary advantage is that it enables the reproduction of supervised-learning functionality without necessitating the explicit classification of training inputs. The algorithm would appear to have possible applications in any task requiring unsupervised learning and to suggest a possible form for a novel neural-network learning method.

References

- [1] Duda, R. and Hart, P. (1973). *Pattern Classification and Scene Analysis*. New York: Wiley.
- [2] Van Ryzin, J. (1977). *Classification and Clustering*. London: Academic Press.
- [3] Rumelhart, D. and Zipser, D. (1986). Feature discovery by competitive learning. In D. Rumelhart, J. McClelland and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructures of Cognition. Vol I* (pp. 151-193). Cambridge, Mass.: MIT Press.
- [4] Darken, C. and Moody, J. (1990). Fast adaptive k-means clustering: some empirical results. *Proceedings of IJCNN, San Diego*.
- [5] Selim, S. and Ismail, M. (1984). K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6, No. 1 (pp. 81-87).
- [6] Everitt, B. (1974). *Cluster Analysis*. London: Heinemann.
- [7] Thrun, S., Bala, J., Bloedorn, E., Bratko, I., Cestnik, B., Cheng, J., De Jong, K., Dzeroski, S., Fisher, D., Fahlman, S., Hamann, R., Kaufman, K., Keller, S., Kononenko, I., Kreuziger, J., Michalski, R., Mitchell, T., Pachowicz, P., Reich, Y., Vafaie, H., Van de Welde, W., Wenzel, W., Wnek, J. and Zhang, J. (1991). The MONK’s problems - a performance comparison of different learning algorithms. CMU-CS-91-197, School of Computer Science, Carnegie-Mellon University.