ESANN'2006 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2006, d-side publi., ISBN 2-930307-06-4.

Outlier Identification with the Harmonic Topographic Mapping

Marian Peña and Colin Fyfe

Applied Computational Intelligence Research Unit
School of Computing
The University of Paisley
Scotland

Abstract. We review two versions of a topology preserving algorithm one of which we had previously [1] found to be more successful in defining smooth manifolds and tight clusters. In the context of outlier detection, however, the other is shown to be more successful. Nevertheless, we show that, by using local kernels for calculation of responsibilities, the first one can also be used in this manner.
Keywords: Topographic Mapping, kernels,responsibilities, HaToM.

1 Introduction

Topographic mappings have proved to be a very useful tool when dealing with high-dimensional data. The non-linear projection of the data into a two dimensional latent space gives first of all a way to visualise the data, and secondly allows us to find clusters in this low-dimensional space. In this paper, we investigate the effect of outliers on the mapping and show how a mapping which had previously been shown to be successful at identifying smooth manifolds, can also identify outliers by using local kernels to calculate responsibilities.

2 Harmonic K-Means

Harmonic Means or Harmonic Averages are defined for spaces of derivatives. For example, if you travel $\frac{1}{2}$ of a journey at 10 km/hour and the other $\frac{1}{2}$ at 20 km/hour, your total time taken is $\frac{d}{10} + \frac{d}{20}$ and so the average speed is $\frac{2d}{\frac{d}{10}+\frac{d}{20}} = \frac{2}{\frac{1}{10}+\frac{1}{20}}$. In general, the Harmonic Average of K centre points, $a_1, ..., a_K$, is defined as

$$HA(\{a_i, i = 1, \cdots, K\}) = \frac{K}{\sum_{k=1}^{K} \frac{1}{a_k}} \tag{1}$$

Harmonic Means were applied to the K-Means algorithm in [2] to make the K-means a more robust algorithm. The Harmonic average performance function and the recursive formula to update the centres are

$$Perf_{HA} = \sum_{i=1}^{N} \frac{K}{\sum_{k=1}^{K} \frac{1}{d(\mathbf{x}_i, \mathbf{m}_k)^2}} \qquad \mathbf{m}_k = \frac{\sum_{i=1}^{N} \frac{1}{d_{i,k}^4 (\sum_{l=1}^{K} \frac{1}{d_{i,l}^2})^2} \mathbf{x}_i}{\sum_{i=1}^{N} \frac{1}{d_{i,k}^4 (\sum_{l=1}^{K} \frac{1}{d_{i,l}^2})^2}} \tag{2}$$

where $d_{i,k}$ is the Euclidean distance between the i^{th} data point and the k^{th} centre.

[2] have extensive simulations showing that this algorithm converges to a better solution (less prone to finding a local minimum because of poor initialisation) than both standard K-means or a mixture of experts trained using the EM algorithm.

3 The Harmonic Topograpic Map

The Harmonic Topographic Map (HaToM) was developed as a clustering alternative to the Topographic Product of Experts (ToPoE) [3], which is also inspired by the Generative Topographic Map(GTM). The HaToM has the same structure as the GTM, with K latent points that are mapped to a feature space by M Gaussian basis functions, and then into the data space by a matrix of weights W. Each latent point, indexed by k is mapped, through a set of M basis functions, $\Phi_1(t_k), \Phi_2(t_k), \cdots, \Phi_M(t_k)$ to a centre in data space, $\mathbf{m}_k = \Phi(t_k)W$. But the similarity ends there because the objective function is not the GTM one, nor is it optimised with the Expectation-Maximization (EM) algorithm. Instead, the HaToM uses the well proved clustering abilities of the K-means algorithm, improved by using harmonic means to make it insensitive to initialisation ([2]).

The basic batch algorithm often exhibited twists, such as are well-known in the Self-organizing Map (SOM) [4], so we developed a growing method that prevents the mapping from developing these twists. The latent points are arranged in a square grid in a similar manner to the SOM grid.

We developed two versions of the algorithm (see [1]). The main structure for the Data-driven HaToM or D-HaToM is as follows:

1. Initialise K to 2. Initialise the W weights randomly and spread the centres of the M basis functions uniformly in latent space.
2. Initialise the K latent points uniformly in latent space.
3. Calculate the projection of the latent points to data space. This gives the K centres, $\mathbf{m}_k$.
   (a) count=0
   (b) For every data point, $\mathbf{x}_i$, calculate $d_{i,k} = ||\mathbf{x}_i - \mathbf{m}_k||$.
   (c) Recalculate centres, $\mathbf{m}_k$, using (2).
   (d) If count<MAXCOUNT, count= count +1 and return to 2c
4. Recalculate W using $(\Phi^T\Phi + \delta I)^{-1}\Phi^T\mathbf{\Xi}$ where $\mathbf{\Xi}$ is the matrix containing the K centres, I is identity matrix and δ is a small constant, necessary because initially $K < M$ and so the matrix $\Phi^T\Phi$ is singular.
5. If $K < K_{max}$, $K = K + 1$ and return to 2.

We do not randomise W each time we augment K, but we use the value from the previous iteration to update the centres $\mathbf{m}_k$ with the increased number of latent points.

If we wish to use the mapping for visualisation, we must map data points into latent space. To do this, we define the responsibility that the k^{th} latent point has for the i^{th} data point and the new data point is placed at y_i

$$r_{ik} = \frac{\exp(-\gamma d_{i,k})}{\sum_{l=1}^{K} \exp(-\gamma d_{i,l})} \qquad y_i = \sum_{k=1}^{K} r_{i,k} t_k \tag{3}$$

where t_k is the position of the k^{th} latent point in latent space. γ is known as the width of the responsibilities.

In the Model-driven HaToM or M-HaToM, we give greater credence to the model by recalculating W and hence the centres, $\mathbf{m}_k$, within the central loop each time. Thus we are explicitly forcing the structure of the M-HaToM model on the data:

1. Initialise K and the W weights.
2. Initialise the K latent points uniformly in latent space.
   (a) count=0
   (b) Calculate the K centres, $\mathbf{m}_k$.
   (c) For every data point, $\mathbf{x}_i$, calculate $d_{i,k} = ||\mathbf{x}_i - \mathbf{m}_k||$.
   (d) Recalculate centres, $\mathbf{m}_k$, using (2).
   (e) Recalculate W as in D-HaToM.
   (f) If count<MAXCOUNT, count= count +1 and return to 2c
3. If $K < K_{max}$, $K = K + 1$ and return to 2.

The visualisation of the $\mathbf{y}_i$ values in latent space is the same as above. In [1], we showed that this version had several advantages over the D-HaToM: in particular, the M-HaToM creates tighter clusters of data points and finds an underlying data manifold smoothly no matter how many latent points are used in creating the manifold. The D-HaToM, on the other hand, is too responsive to the data (too influenced by the noise), but as we shall see, this quality makes it more suitable for outlier detection.

3.1 The Fundamental Clustering Problems Suite (FCPS)

In this paper we deal with the influence of outliers in the application of both versions of HaToM. To illustrate outlier detection with HaToM, we used one of the datasets (Target) that appears in The Fundamental Clustering Problems Suite (FCPS)[5]; these datasets are all low-dimensional, but some algorithms like K-Means have difficulty in clustering them because of the existence of outliers, so they are suitable for illustrating the different visualisation capabilities of the

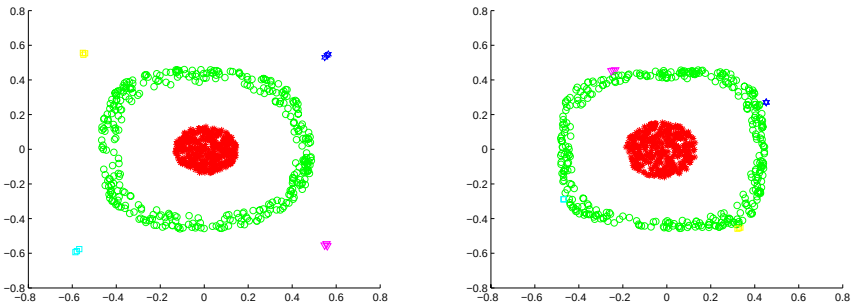

Fig. 1: D-HaTom projection of the target data into latent space with Gaussian kernels(left); the M-HaToM projections in latent space(right).

different kernels. Target comprises a two dimensional data set, consisting of a tight cluster of points at the origin, with a surrounding but disjoint circular manifold and 4 pairs of outliers, each pair close together but separate from all other data points.

We see in Figure 1 that the Gaussian kernel-based responsibilities for the M-HaToM does not separate properly the outliers from the rest of the data, though it can do so for the D-HaToM. However, since we have previously identified M-HaToM as having advantages in projection and clustering, we would like to continue to use M-HaToM but find a way to make its projections of outliers more explicitly different from the main body of the data.

To understand the reason for this difficulty, we show in Figure 2 the position of the projections of the latent points for the two mappings. We see that the noise-resistant properties of M-HaToM have resulted in no centres being drawn to the outliers while in D-HaToM, several centres have been allocated to each pair of outliers. Thus it is easy for the latter to identify outliers. To enable the former to do so, we will use kernels which are more local when calculating responsibilities.

4 Different Kernels for the Responsibilities

One of the main attractions of the HaToM compared with e.g. ToPoE is that the algorithm does not require responsibilities. These are only used when we are using HaToM to visualise data, i.e. when we are working in latent space, L. Thus we do not have to give any consideration to probabilistic constraints, and the computational effort is reduced. If $\mathbf{y}_n$ is the point in latent space corresponding to $\mathbf{x}_n$, we have

$$\mathbf{y}_n = \sum_{k=1}^{K} \mathbf{t}_k r_{kn} \in L \qquad r_{kn} = \frac{\exp(-(\mathbf{x}_n - \mathbf{m}_k)^2)}{\sum_{j=1}^{K} \exp(-(\mathbf{x}_n - \mathbf{m}_j)^2)} \tag{4}$$

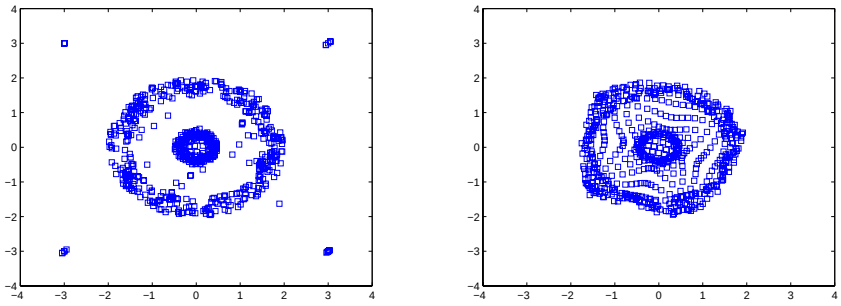

Fig. 2: The latent points' projections ($\mathbf{m}_k$) into data space using D-HaTom(left) , and those from the M-HaToM (right) with delta=0.001.

where $\mathbf{t}_k$ is the coordinate of the k^{th} latent point in L and $r_{kn}$is determined in data space. (4) recalls the Nadaraya-Watson kernel estimator

$$\widehat{f}(x) = \frac{\sum_{i=1}^{N} K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^{N} K_\lambda(x_0, x_i)} \tag{5}$$

This might suggest that we could use other kernels for the responsibilities such as the Epanechnikov quadratic kernel

$$C_\lambda(k,n) = \quad D\left(\frac{|\mathbf{x}_n - \mathbf{m}_k|}{\lambda}\right) \quad \text{where } D(t) = \quad \begin{cases} \frac{3}{4}(1-t^2) & \text{if } |t| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

or the tri-cube function with

$$D(t) = \begin{cases} (1-t^3)^3 & \text{if } |t| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

both of which are more local and have compact support with

$$r_{kn} = \frac{C_\lambda(k,n)}{\sum_{j=1}^{K} C_\lambda(j,n)} \qquad \mathbf{y}_n = \frac{\sum_{k=1}^{K} \mathbf{t}_k C_\lambda(k,n)}{\sum_{k=1}^{K} C_\lambda(k,n)} \tag{8}$$

In Figure 3, we show the effect of using the Epanechnikov kernel with both the D-HaToM and M-HaToM: in both the outliers can be easily detected. Similar results are achieved with the tri-cube kernel.

5 Conclusions

In this paper we reviewed both versions of the HaToM algorithm. We noted that the previous success [1] of the M-HaToM in finding smooth manifolds and tight clusters was based on features which meant that this algorithm found it difficult

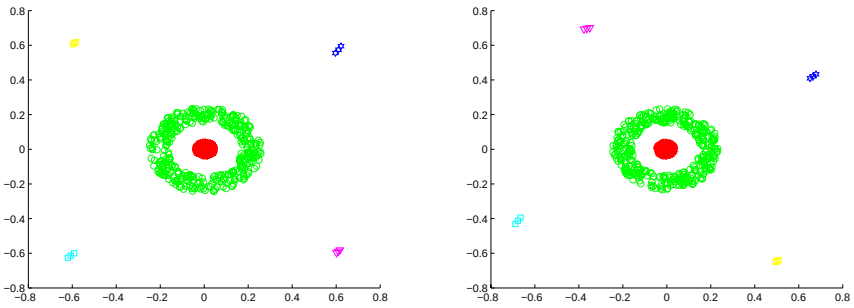

Fig. 3: The projection of the target data into latent space with Epanechnikov kernels using D-HaTom (left), and M-HaToM(right).

to separate outliers from the main data manifolds. The D-HaToM did not share this difficulty. However, by using local kernels when calculating responsibilities, we were able to detect the outliers again easily in the projections. We illustrated this on a simple two-dimensional data set. Future work will investigate the effect on higher dimensional data.

References

[1] M. Peña and C. Fyfe. Model- and data-driven harmonic topographic maps. *WSEAS TRANSACTIONS ON COMPUTERS*, 4(9):1033–1044, September 2005.

[2] B. Zhang, M. Hsu, and U. Dayal. K-harmonic means - a data clustering algorithm. Technical report, HP Laboratories, Palo Alto, October 1999.

[3] C. Fyfe. Topographic product of experts. In *International Conference on Artificial Neural Networks, ICANN2005*, 2005.

[4] T. Kohonen. *Self-Organising Maps*. Springer, 1995.

[5] A. Ultsch. Clustering with som: U*c. In *Proc. Workshop on Self-Organizing Maps*, pages 75–82, 2005.