Available online at www.sciencedirect.com
ScienceDirect

Pattern Recognition 41 (2008) 1012–1029

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/pr

Spectral clustering with eigenvector selection

Tao Xiang*, Shaogang Gong

Department of Computer Science, Queen Mary, University of London, London E1 4NS, UK

Received 24 October 2006; received in revised form 11 June 2007; accepted 30 July 2007

Abstract

The task of discovering natural groupings of input patterns, or clustering, is an important aspect of machine learning and pattern analysis. In this paper, we study the widely used spectral clustering algorithm which clusters data using eigenvectors of a similarity/affinity matrix derived from a data set. In particular, we aim to solve two critical issues in spectral clustering: (1) how to automatically determine the number of clusters, and (2) how to perform effective clustering given noisy and sparse data. An analysis of the characteristics of eigenspace is carried out which shows that (a) not every eigenvectors of a data affinity matrix is informative and relevant for clustering; (b) eigenvector selection is critical because using uninformative/irrelevant eigenvectors could lead to poor clustering results; and (c) the corresponding eigenvalues cannot be used for relevant eigenvector selection given a realistic data set. Motivated by the analysis, a novel spectral clustering algorithm is proposed which differs from previous approaches in that only informative/relevant eigenvectors are employed for determining the number of clusters and performing clustering. The key element of the proposed algorithm is a simple but effective relevance learning method which measures the relevance of an eigenvector according to how well it can separate the data set into different clusters. Our algorithm was evaluated using synthetic data sets as well as real-world data sets generated from two challenging visual learning problems. The results demonstrated that our algorithm is able to estimate the cluster number correctly and reveal natural grouping of the input data/patterns even given sparse and noisy data.
© 2007 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.

Keywords: Spectral clustering; Feature selection; Unsupervised learning; Image segmentation; Video behaviour pattern clustering

1. Introduction

The task of discovering natural groupings of input patterns, or clustering, is an important aspect of machine learning and pattern analysis. Clustering techniques are more and more frequently adopted by various research communities due to the increasing need of modelling large amount of data. As an unsupervised data analysis tool, clustering is desirable for modelling large date sets because the tedious and often inconsistent manual data labelling process can be avoided. The most popular clustering techniques are perhaps mixture models and K-means which are based on estimating explicit models of data distribution. Typically the distribution of a data set generated by a real-world system is complex and of an unknown shape, especially given the inevitable existence of noise. In this case, mixture models and K-means are expected to yield poor results since an explicit estimation of data distribution is difficult if even possible. Spectral clustering offers an attractive alternative which clusters data using eigenvectors of a similarity/affinity matrix derived from the original data set. In certain cases spectral clustering even becomes the only option. For instance, when different data points are represented using feature vectors of variable lengths, mixture models or K-means cannot be applied, while spectral clustering can still be employed as long as a pair-wise similarity measure can be defined for the data.

In spite of the extensive studies in the past on spectral clustering [1–9], two critical issues remain largely unresolved: (1) how to automatically determine the number of clusters, and (2) how to perform effective clustering given noisy and sparse data. Most previous work assumed that the number of clusters is known or has been manually set [1,2,5]. Recently researchers started to tackle the first issue, i.e. determining the cluster number automatically. Smyth [4] proposed to use a Monte

* Corresponding author. Tel.: +44 0 20 7882 8020; fax: +44 0 20 8980 6533.
E-mail addresses: txiang@dcs.qmul.ac.uk (T. Xiang), sgg@dcs.qmul.ac.uk (S. Gong).

0031-3203/$30.00 © 2007 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2007.07.023

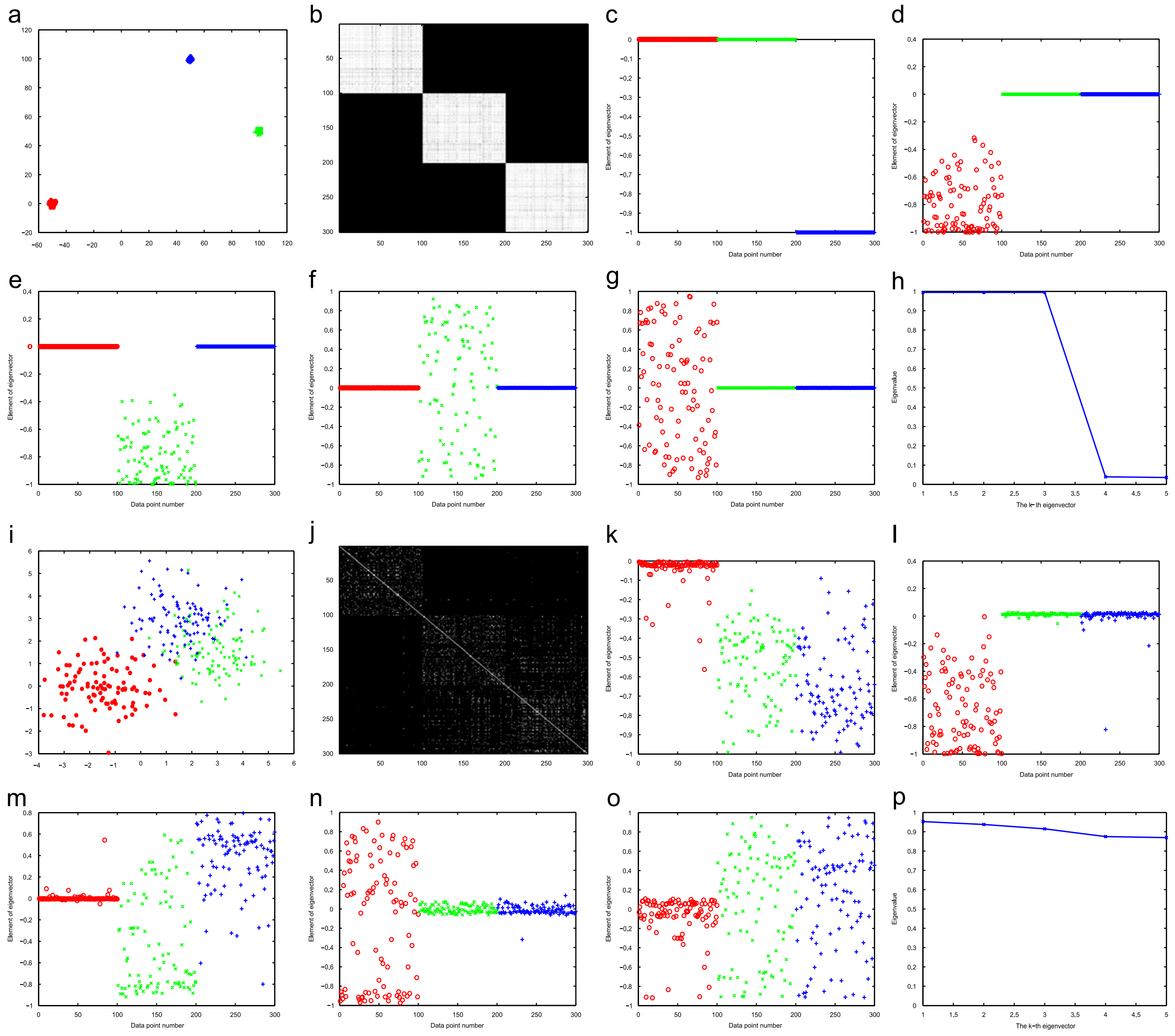

Fig. 1. Examples showing that not all eigenvectors are informative for spectral clustering. (a) shows a well-separated 2-D data set consisting of three clusters. The affinity matrix (b) shows clear block structure. (c)–(e) show that the top 3 eigenvectors contain useful information about the natural grouping of the data. For instance, a simple thresholding of the first eigenvector can separate one cluster from the other two. Comparatively, the fourth and fifth eigenvectors are less informative. (i)–(p) show another example with a fair amount of overlapping between clusters. As expected, in this less 'ideal' case, the distributions of eigenvector elements are less informative in general in that the gaps between elements corresponding to different clusters are more blurred. However, it is still the case that some eigenvectors are more informative than others. Note that for better illustration we have ordered the points in (b)–(g) and (j)–(o) so that points belonging to the same cluster appear consecutively. In all figures, the three clusters are indicated using different symbols in different colours.

Carlo cross validation approach to determine the number of clusters for sequences modelled using hidden Markov models (HMMs). This approach is computationally expensive and thus not suitable for large data sets common to applications such as image segmentation. Porikli and Haga [6] employed a validity score computed using the largest eigenvectors[1] of a data affinity matrix to determine the number of clusters for video-based activity classification. Zelnik-Manor and Perona [7] proposed to determine the optimal cluster number through minimising the cost of aligning the top eigenvectors with a canonical coordinate system. The approaches in Refs. [6] and [7] are similar in that both of them are based on analysing the structures of the largest eigenvectors of a normalised data affinity matrix. In particular, assuming a number K_m that is considered to be safely larger than the true number of clusters K_{true}, the top K_m eigenvectors were exploited in both approaches to infer K_{true}. However, these approaches do not

[1] The largest eigenvectors are eigenvectors whose corresponding eigenvalues are the largest in magnitude.

take into account the inevitable presence of noise in a realistic data set, i.e. they fail to address explicitly the second issue. They are thus error prone especially when the sample size is small.

We argue that the key to solving the two above-mentioned issues is to select the relevant eigenvectors which provide useful information about the natural grouping of data. To justify the need for eigenvector selection, we shall answer a couple of fundamental questions in spectral clustering. First, does every eigenvector provide useful information (therefore is needed) for clustering? It has been shown analytically that in an 'ideal' case in which all points in different clusters are infinitely far apart, the elements of the top K_{true} eigenvectors form clusters with distinctive gaps between them which can be readily used to separate data into different groups [5]. In other words, all top K_{true} eigenvectors are equally informative. However, theoretically it is not guaranteed that other top eigenvectors are equally informative even in the 'ideal' case. Figs. 1(f) and (g) suggest that, in a 'close-to-ideal' case, not all top eigenvectors are equally informative and useful for clustering. Now let us look at a realistic case where there exist noise and a fair amount of similarities between clusters. In this case, the distribution of elements of an eigenvector is far more complex. A general observation is that the gaps between clusters in the elements of the top eigenvectors are blurred and some eigenvectors, including those among the top K_{true}, are uninformative [5,8,9]. This is shown clearly in Fig. 1. Therefore, the answer to the first question is 'no' especially given a realistic data set. Second, is eigenvector selection necessary? It seems intuitive to include those less informative eigenvectors in the clustering process because, in principle, a clustering algorithm is expected to perform better given more information about the data grouping. However, in practice, the inclusion of uninformative eigenvectors can degrade the clustering process as demonstrated extensively later in the paper. This is hardly surprising because in a general context of pattern analysis, the importance of removing those noisy/uninformative features has long been recognised [10,11]. The answer to the second question is thus 'yes'. Given the answers to the above two questions, it becomes natural to consider performing eigenvector selection for spectral clustering. In this paper, we propose a novel relevant eigenvector selection algorithm and demonstrate that it indeed leads to more efficient and accurate estimation of the number of clusters and better clustering results compared to existing approaches. To our knowledge, this paper is the first to use eigenvector selection to improve spectral clustering results.

The rest of the paper is organised as follows. In Section 2, we first define the spectral clustering problem. An efficient and robust eigenvector selection algorithm is then introduced which measures the relevance of each eigenvector according to how well it can separate a data set into different clusters. Based on the eigenvector selection result, only the relevant eigenvectors will be used for a simultaneous cluster number estimation and data clustering based on a Gaussian mixture model (GMM) and the Bayesian information criterion (BIC). The effectiveness and robustness of our approach is demonstrated first in Section 2 using synthetic data sets, then in Sections 3 and 4 on solving two real-world visual pattern analysis problems. Specifically, in Section 3, the problem of image segmentation using spectral clustering is investigated. In Section 4, human behaviour captured on CCTV footage in a secured entrance surveillance scene is analysed for automated discovery of different types of behaviour patterns based on spectral clustering. Both synthetic and real data experiments presented in this paper show that our approach outperforms the approaches proposed in Refs. [6,7]. The paper concludes in Section 5.

2. Spectral clustering with eigenvector relevance learning

Let us first formally define the spectral clustering problem. Given a set of N data points/input patterns represented using feature vectors

$$\mathbf{D} = \{\mathbf{f}_1, \ldots, \mathbf{f}_n, \ldots, \mathbf{f}_N\}, \tag{1}$$

we aim to discover the natural grouping of the input data. The optimal number of groups/clusters K_o is automatically determined to best describe the underlying distribution of the data set. We have $K_o = K_{true}$ if it is estimated correctly. Note that different feature vectors can be of different dimensionalities. An $N \times N$ affinity matrix $\mathbf{A} = \{A_{ij}\}$ can be formed whose element A_{ij} measures the affinity/similarity between the ith and jth feature vectors. Note that $\mathbf{A}$ needs to be symmetric, i.e. $A_{ij} = A_{ji}$. The eigenvectors of $\mathbf{A}$ can be employed directly for clustering. However, it has been shown in Refs. [1,2] that it is more desirable to perform clustering based on the eigenvectors of the normalised affinity matrix $\bar{\mathbf{A}}$, defined as

$$\bar{\mathbf{A}} = \mathbf{L}^{-\frac{1}{2}}\mathbf{A}\mathbf{L}^{-\frac{1}{2}}, \tag{2}$$

where $\mathbf{L}$ is an $N \times N$ diagonal matrix with $L_{ii} = \sum_j A_{ij}$. We assume that the number of clusters is between 1 and K_m, a number considered to be sufficiently larger than K_o. The training data set is then represented in an eigenspace using the K_m largest eigenvectors of $\bar{\mathbf{A}}$, denoted as

$$\mathbf{D_e} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n, \ldots, \mathbf{x}_N\}, \tag{3}$$

with the nth feature vector $\mathbf{f}_n$ being represented as a K_m dimensional vector $\mathbf{x}_n = [e_{1n}, \ldots, e_{kn} \ldots, e_{K_m n}]$, where e_{kn} is the nth element of the kth largest eigenvector $\mathbf{e_k}$. Note that now each feature vector in the new data set is of the same dimensionality K_m. The task of spectral clustering now is to determine the number of clusters and then group the data into different clusters using the new data representation in the eigenspace.

As analysed earlier in the paper, intrinsically only a subset of the K_m largest eigenvectors are relevant for grouping K_o clusters and it is important to first identify and remove those irrelevant/uninformative eigenvectors before performing clustering. How do we measure the relevance of an eigenvector? An intuitive solution would be investigating the associated eigenvalue for each eigenvector. The analysis in Ref. [5] shows that in an 'ideal' case where different clusters are infinitely far apart, the top K_{true} (relevant) eigenvectors have a corresponding eigenvalue of magnitude 1 and others do not. In this case, simply selecting those eigenvectors would solve the problem. In fact, estimation of the number of clusters also becomes trivial by simply looking at the eigenvalues: it is equal to the number of eigenvalues of magnitude 1. Indeed, eigenvalues are useful when the data are clearly separated, i.e. close to the 'ideal' case. This is illustrated in Fig. 1(h) which shows that both eigenvector selection and cluster number estimation can be solved based purely on eigenvalues. However, given a 'not-so-ideal' data set such as the one in Fig. 1(i), the eigenvalues are not useful as all eigenvectors can assume high magnitude and higher eigenvectors do not necessarily mean higher relevance (see Figs. 1 (k)–(p)). Next, we propose a data-driven eigenvector selection approach based on exploiting the structure of each eigenvector with no assumption made about the distribution of the original data set **D**. Specifically, we propose to measure the relevance of an eigenvector according to how well it can separate a data set into different clusters.

We denote the likelihood of the kth largest eigenvector $\mathbf{e_k}$ being relevant as $R_{\mathbf{e_k}}$, with $0 \leqslant R_{\mathbf{e_k}} \leqslant 1$. We assume that the elements of $\mathbf{e_k}$, e_{kn} can follow two different distributions, namely unimodal and multimodal, depending on whether $\mathbf{e_k}$ is relevant. The probability density function (pdf) of e_{kn} is thus formulated as a mixture model of two components:

$$p(e_{kn}|\theta_{e_{kn}}) = (1 - R_{\mathbf{e_k}})p(e_{kn}|\theta^1_{e_{kn}}) + R_{\mathbf{e_k}}p(e_{kn}|\theta^2_{e_{kn}}),$$

where $\theta_{e_{kn}}$ are the parameters describing the distribution, $p(e_{kn}|\theta^1_{e_{kn}})$ is the pdf of e_{kn} when $\mathbf{e_k}$ is irrelevant/redundant and $p(e_{kn}|\theta^2_{e_{kn}})$ otherwise. $R_{\mathbf{e_k}}$ acts as the weight or mixing probability of the second mixture component. In our algorithm, the distribution of e_{kn} is assumed to be a single Gaussian (unimodal) to reflect the fact that $\mathbf{e_k}$ cannot be used for data clustering when it is irrelevant:

$$p(e_{kn}|\theta^1_{e_{kn}}) = \mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1}),$$

where $\mathcal{N}(.|\mu, \sigma)$ denotes a Gaussian of mean μ and covariance σ^2. We assume the second component of $P(\mathbf{e_k}|\boldsymbol{\theta}_{\mathbf{e_k}})$ as a mixture of two Gaussians (multimodal) to reflect the fact that $\mathbf{e_k}$ can separate one cluster of data from the others when it is relevant:

$$p(e_{kn}|\theta^2_{e_{kn}}) = w_k\mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2}) + (1 - w_k)\mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3}),$$

where w_k is the weight of the first Gaussian in $p(e_{kn}|\theta^2_{e_{kn}})$. There are two reasons for using a mixture of two Gaussians even when e_{kn} forms more than two clusters and/or the distribution of each cluster is not Gaussian: (1) in these cases, a mixture of two Gaussians ($p(e_{kn}|\theta^2_{e_{kn}})$) still fits better to the data compared to a single Gaussian ($p(e_{kn}|\theta^1_{e_{kn}})$); (2) its simple form means that only small number of parameters are needed to describe $p(e_{kn}|\theta^2_{e_{kn}})$. This makes model learning possible even given sparse data.

There are 8 parameters required for describing the distribution of e_{kn}:

$$\theta_{e_{kn}} = \{R_{\mathbf{e_k}}, \mu_{k1}, \mu_{k2}, \mu_{k3}, \sigma_{k1}, \sigma_{k2}, \sigma_{k3}, w_k\}. \tag{4}$$

The maximum likelihood (ML) estimate of $\theta_{e_{kn}}$ can be obtained using the following algorithm. First, the parameters of the first mixture component $\theta^1_{e_{kn}}$ are estimated as $\mu_{k1} = (1/N)\sum_{n=1}^N e_{kn}$ and $\sigma_{k1} = (1/N)\sum_{n=1}^N (e_{kn} - \mu_{k1})^2$. The rest 6 parameters are then estimated iteratively using expectation maximisation (EM) [12]. Specifically, in the E-step, the posterior probability that each mixture component is responsible for e_{kn} is estimated as:

$$h^1_{kn} = \frac{(1 - R_{\mathbf{e_k}})\mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1})}{(1 - R_{\mathbf{e_k}})\mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1}) + w_k R_{\mathbf{e_k}}\mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2}) + (1 - w_k)R_{\mathbf{e_k}}\mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3})},$$

$$h^2_{kn} = \frac{w_k R_{\mathbf{e_k}}\mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2})}{(1 - R_{\mathbf{e_k}})\mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1}) + w_k R_{\mathbf{e_k}}\mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2}) + (1 - w_k)R_{\mathbf{e_k}}\mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3})},$$

$$h^3_{kn} = \frac{(1 - w_k)R_{\mathbf{e_k}}\mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3})}{(1 - R_{\mathbf{e_k}})\mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1}) + w_k R_{\mathbf{e_k}}\mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2}) + (1 - w_k)R_{\mathbf{e_k}}\mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3})}.$$

In the M-step, 6 distribution parameters are re-estimated as:

$$R^{new}_{\mathbf{e_k}} = 1 - \frac{1}{N}\sum_{n=1}^N h^1_{kn}, \quad w^{new}_k = \frac{1}{R^{new}_{\mathbf{e_k}} N}\sum_{n=1}^N h^2_{kn},$$

$$\mu^{new}_{k2} = \frac{\sum_{n=1}^N h^2_{kn} e_{kn}}{\sum_{n=1}^N h^2_{kn}}, \quad \mu^{new}_{k3} = \frac{\sum_{n=1}^N h^3_{kn} e_{kn}}{\sum_{n=1}^N h^3_{kn}},$$

$$\sigma^{new}_{k2} = \frac{\sum_{n=1}^N h^2_{kn}(e_{kn} - \mu^{new}_{k2})^2}{\sum_{n=1}^N h^2_{kn}},$$

$$\sigma^{new}_{k3} = \frac{\sum_{n=1}^N h^3_{kn}(e_{kn} - \mu^{new}_{k3})^2}{\sum_{n=1}^N h^3_{kn}}.$$

Since the EM algorithm is essentially a local (greedy) searching method, it could be sensitive to parameter initialisation especially given noisy and sparse data [12]. To overcome this problem, the value of $R_{\mathbf{e_k}}$ is initialised as 0.5 and the values of the other five parameters, namely μ_{k2}, μ_{k3}, σ_{k2}, σ_{k3} and w_k are initialised randomly. The solution that yields the highest $p(e_{kn}|\theta^2_{e_{kn}})$ over multiple random initialisations is chosen.

It is important to point out the following:

1. Although our relevance learning algorithm is based on estimating the distribution of the elements of each eigenvector, we are only interested in learning how likely the distribution is unimodal or multimodal, which is reflected by the value of $R_{\mathbf{e_k}}$. In other words, among the 8 free parameters of the eigenvector distribution (Eq. (4)), $R_{\mathbf{e_k}}$ is the only parameter that we are after. This is why our algorithm works well

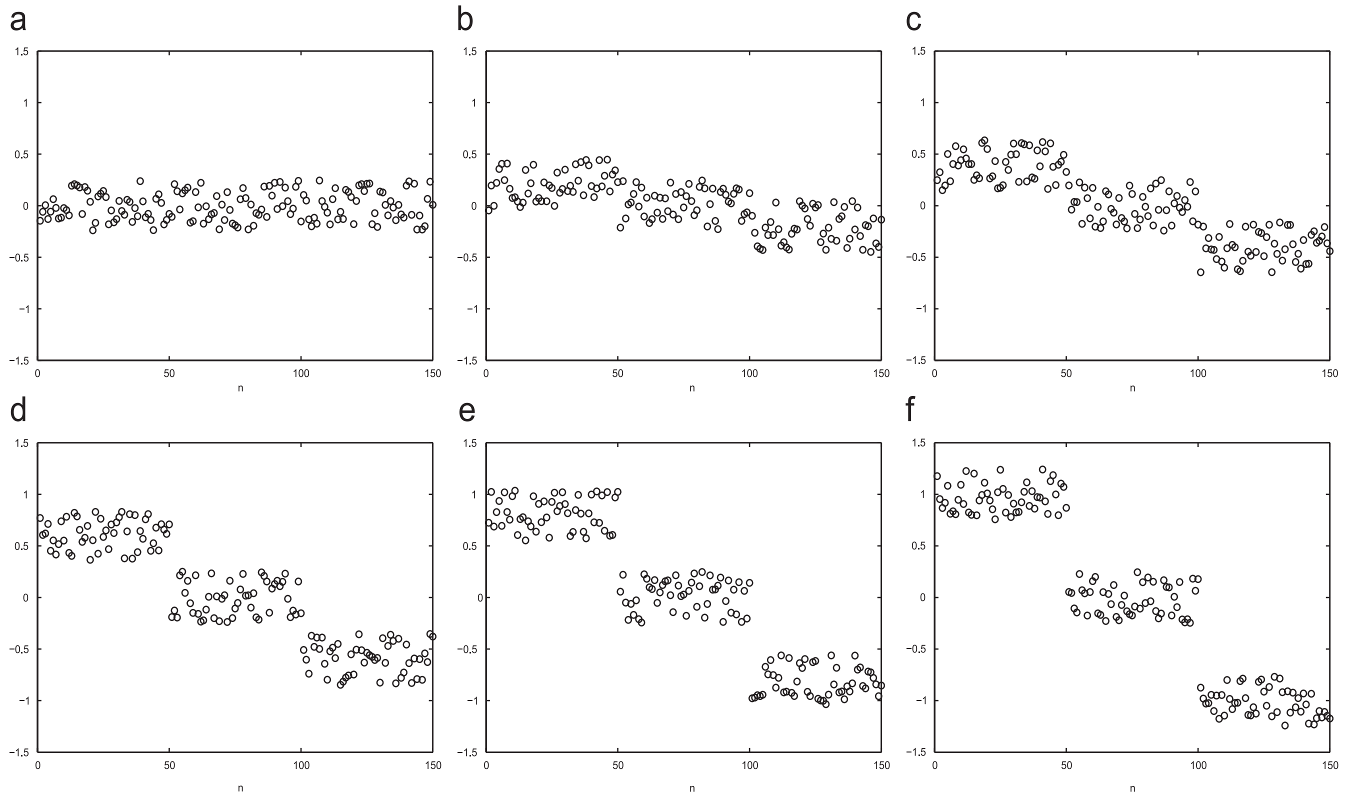

Fig. 2. Synthetic eigenvectors and the estimated relevance measure $R_{\mathbf{e_k}}$. The elements of each eigenvectors are composed of three uniformly distributed clusters. The mean of the three clusters are $m1$, 0, and $-m1$, respectively. Obviously, the bigger the value of $m1$, the more distinctive three clusters are formed in the distribution of the eigenvector and the more relevant the eigenvector is. (a) $m1=0.0$, $R_{\mathbf{e_k}}=0.2992$. (b) $m1=0.2$, $R_{\mathbf{e_k}}=0.4479$. (c) $m1=0.4$, $R_{\mathbf{e_k}}=0.6913$, (d) $m1=0.6$, $R_{\mathbf{e_k}}=0.7409$. (e) $m1=0.8$, $R_{\mathbf{e_k}}=0.8302$. (f) $m1=1.0$, $R_{\mathbf{e_k}}=1.0$.

even when there are more than 2 clusters and/or the distribution of each cluster is not Gaussian. This is demonstrated by a simple example in Fig. 2 and more examples later in the paper. In particular, Fig. 2 shows that when the distributions of eigenvector elements belonging to different clusters are uniform and there are more than two clusters, the value of $R_{\mathbf{e_k}}$ estimated using our algorithm can still accurately reflect how relevant/informative an eigenvector is. Note that in the example shown in Fig. 2 synthetic eigenvectors are examined so that we know exactly what the distribution of the eigenvector elements is.

2. The distribution of e_{kn} is modelled as a mixture of two components with one of the mixture itself being a mixture model. In addition, the two mixtures of the model have clear semantic meanings: the first mixture corresponds to the unimodal mode of the data, the second mixture correspond to the multimodal model. This makes the model clearly different from a mixture of three components. This difference must be reflected by the model learning procedure, i.e. instead of learning all 8 parameters simultaneously using EM as one does for a standard 3-component mixture, the parameters are learned in two steps and only the second step is based on the EM algorithm. Specifically, $\theta_{e_{kn}}$ (Eq. (4)) are *not* estimated iteratively using a standard EM algorithm although part of $\theta_{e_{kn}}$, namely $\theta^2_{e_{kn}}$, are. This is critical because if all the 8 parameters are re-estimated in each iteration, the distribution of e_{kn} is essentially modelled as a mixture of three Gaussians, and the estimated $R_{\mathbf{e_k}}$ would represent the weight of two of the three Gaussians. This is very different from what $R_{\mathbf{e_k}}$ is meant to represent, i.e. the likelihood of $\mathbf{e_k}$ being relevant for data clustering.

The estimated $R_{\mathbf{e_k}}$ provides a continuous-value measurement of the relevance of $\mathbf{e_k}$. Since a 'hard-decision' is needed for dimension reduction, we simply eliminate the kth eigenvector $\mathbf{e_k}$ among the K_m candidate eigenvectors if

$$R_{\mathbf{e_k}} < 0.5. \tag{5}$$

The remaining relevant eigenvectors are then weighted using $R_{\mathbf{e_k}}$. This gives us a new data set denoted as

$$\mathbf{D_r} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n, \ldots, \mathbf{y}_N\}, \tag{6}$$

where $\mathbf{y}_n$ is a feature vector of dimensionality K_r which is the number of selected relevant eigenvectors. We model the distribution of $\mathbf{D_r}$ using a GMM for data clustering. BIC is then employed to select the optimal number of Gaussian components, which corresponds to the optimal number of clusters K_o. Each feature vector in the training data set is then labelled as one of the K_o clusters using the learned GMM with K_o Gaussian components. The complete algorithm is summarised in Fig. 3.

```
input  : A set of $N$ data points/input patterns $\mathbf{D}$ (Eqn. (1))
output: The optimal number of clusters $K_o$ and $\mathbf{D}$ grouped into $K_o$ subsets
1 Form the affinity matrix $\mathbf{A} = \{A_{ij}\}$;
2 Construct the normalised affinity matrix $\bar{\mathbf{A}}$;
3 Find the $K_m$ largest eigenvectors of $\bar{\mathbf{A}}$, $\mathbf{e_1}, \mathbf{e_2}, \dots, \mathbf{e_{K_m}}$ and form $\mathbf{D_e}$ (Eqn. (3));
4 Estimate the relevance of each eigenvector $R_{\mathbf{e_k}}$ using the proposed eigenvector selection algorithm;
5 Eliminate the $i$th eigenvector if $R_{\mathbf{e_i}} < 0.5$;
6 Weight the relevant eigenvectors using $R_{\mathbf{e_k}}$ and form a new data set $\mathbf{D_r}$ (Eqn. (6));
7 Model the distribution of $\mathbf{D_r}$ using a GMM and estimate the number of Gaussian component as $K_o$
  using BIC;
8 Assign the original data point $\mathbf{f}_i$ to cluster $j$ if and only if the $i$th data point in $\mathbf{D_r}$ was assigned to the
  $j$th cluster.
```

Fig. 3. The proposed spectral clustering algorithm based on relevant eigenvector selection.

Let us first evaluate the effectiveness of our approach using a synthetic data set. We consider a one-dimensional data set generated from 4 different 3-state HMMs (i.e. the hidden variable at each time instance can assume 3 states). The parameters of an HMM are denoted as $\{\mathbf{A}, \boldsymbol{\pi}, \mathbf{B}\}$, where $\mathbf{A}$ is the transition matrix representing the probabilities of transition between states, $\boldsymbol{\pi}$ is a vector of the initial state probability, and $\mathbf{B}$ contains the parameters of the emission density (in this case Gaussians with a mean μ_i and variance σ_i for the ith state). The parameters of the 4 HMMs are:

$$\mathbf{A_1} = \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1 & 0 & 0 \\ 1/6 & 1/2 & 1/3 \end{bmatrix}, \quad \mathbf{A_2} = \begin{bmatrix} 1/3 & 0 & 2/3 \\ 1/3 & 1/4 & 5/12 \\ 1/6 & 1/2 & 1/3 \end{bmatrix},$$

$$\mathbf{A_3} = \begin{bmatrix} 0 & 1/6 & 5/6 \\ 1/6 & 1/2 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}, \quad \mathbf{A_4} = \begin{bmatrix} 5/12 & 1/2 & 1/12 \\ 0 & 1/6 & 5/6 \\ 1/3 & 1/3 & 1/3 \end{bmatrix},$$

$$\boldsymbol{\pi_1} = \boldsymbol{\pi_2} = \boldsymbol{\pi_3} = \boldsymbol{\pi_4} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix},$$

$$\mathbf{B_1} = \mathbf{B_2} = \mathbf{B_3} = \mathbf{B_4} = \begin{Bmatrix} \mu_1 = 1, \sigma_1^2 = 0.5 \\ \mu_2 = 3, \sigma_2^2 = 0.5 \\ \mu_3 = 5, \sigma_3^2 = 0.5 \end{Bmatrix}. \tag{7}$$

A training set of 80 sequences was generated which was composed of 20 sequences randomly sampled from each HMM. The lengths of these segments were set randomly ranging from 200 to 600. The data were then perturbed with uniformly distributed random noise with a range of [−0.5 0.5]. Given a pair of sequences $\mathbf{S_i}$ and $\mathbf{S_j}$, the affinity between them is computed as:

$$A_{ij} = \frac{1}{2}\left\{\frac{1}{T_j}\log P(\mathbf{S_j}|\mathbf{H_i}) + \frac{1}{T_i}\log P(\mathbf{S_i}|\mathbf{H_j})\right\}, \tag{8}$$

where $\mathbf{H_i}$ and $\mathbf{H_j}$ are the 3-state HMMs learned using $\mathbf{S_i}$ and $\mathbf{S_j}$, respectively,[2] $P(\mathbf{S_j}|\mathbf{H_i})$ is the likelihood of observing $\mathbf{S_j}$ given $\mathbf{H_i}$, $P(\mathbf{S_i}|\mathbf{H_j})$ is the likelihood of observing $\mathbf{S_i}$ given $\mathbf{H_j}$, and T_i and T_j are the lengths of $\mathbf{S_i}$ and $\mathbf{S_j}$, respectively.[3]

The proposed algorithm is used to determine the number of clusters and discover the natural grouping of the data. The results are shown in Figs. 4 and 5. K_m was set to 16 in the experiment. It can be seen from Fig. 5 that the second, third, and fourth eigenvectors contain strong information about the grouping of data while the largest eigenvector is much less informative. The rest eigenvectors contain virtually no information (see Figs. 5(e) and (f)). Fig. 4(b) shows the eigenvalues of the largest 16 eigenvectors. Clearly, from these eigenvalues we cannot infer that the second, third, and fourth eigenvectors are the most informative ones. It can be seen form Fig. 4(c) that the proposed relevance measure $R_{\mathbf{e_k}}$ accurately reflects the relevance of each eigenvectors. By thresholding the relevance measure (Eq. (5)), only $\mathbf{e}_2$, $\mathbf{e}_3$, and $\mathbf{e}_4$ are kept for clustering. Fig. 4(e) shows that the 4 clusters are clearly separable in the eigenspace spanning the top 3 most relevant eigenvectors. It is thus not surprising that the number of clusters was determined correctly as 4 using BIC on the relevant eigenvectors (see Fig. 4(d)). The clustering result is illustrated using the re-ordered affinity matrix in Fig. 4(f), which shows that all four clusters were discovered accurately. We also estimated the number of clusters using three alternative methods: (a) BIC using all 16 eigenvectors; (b) Porikli and Haga's validity score [6] (maximum score correspond to the optimal number); and (c) Zelnik–Perona cost function [7] (minimum cost correspond to the optimal number). Figs. 4(g)–(i) show that none of these methods was able to yield an accurate estimate of the cluster number.

In the previous synthetic data experiment, the 4 clusters in the data set have the same number of data points. It is interesting to evaluate the performance of the proposed algorithm using unevenly distributed data sets since a real-world data set is more likely to be unevenly distributed across clusters. In the next experiment, a data set was generated by the same 4 different 3-state HMMs but with different clusters having different sizes. In particular, the size of the largest cluster is 12 times bigger than that of the smallest one. Each data point was perturbed

[2] Please refer to Ref. [13] for the details on learning the parameters of an HMM from data.

[3] Note that there are other ways to compute the affinity between two sequences modelled using DBNs [14,15]. However, we found through our experiments that using different affinity measures makes little difference.

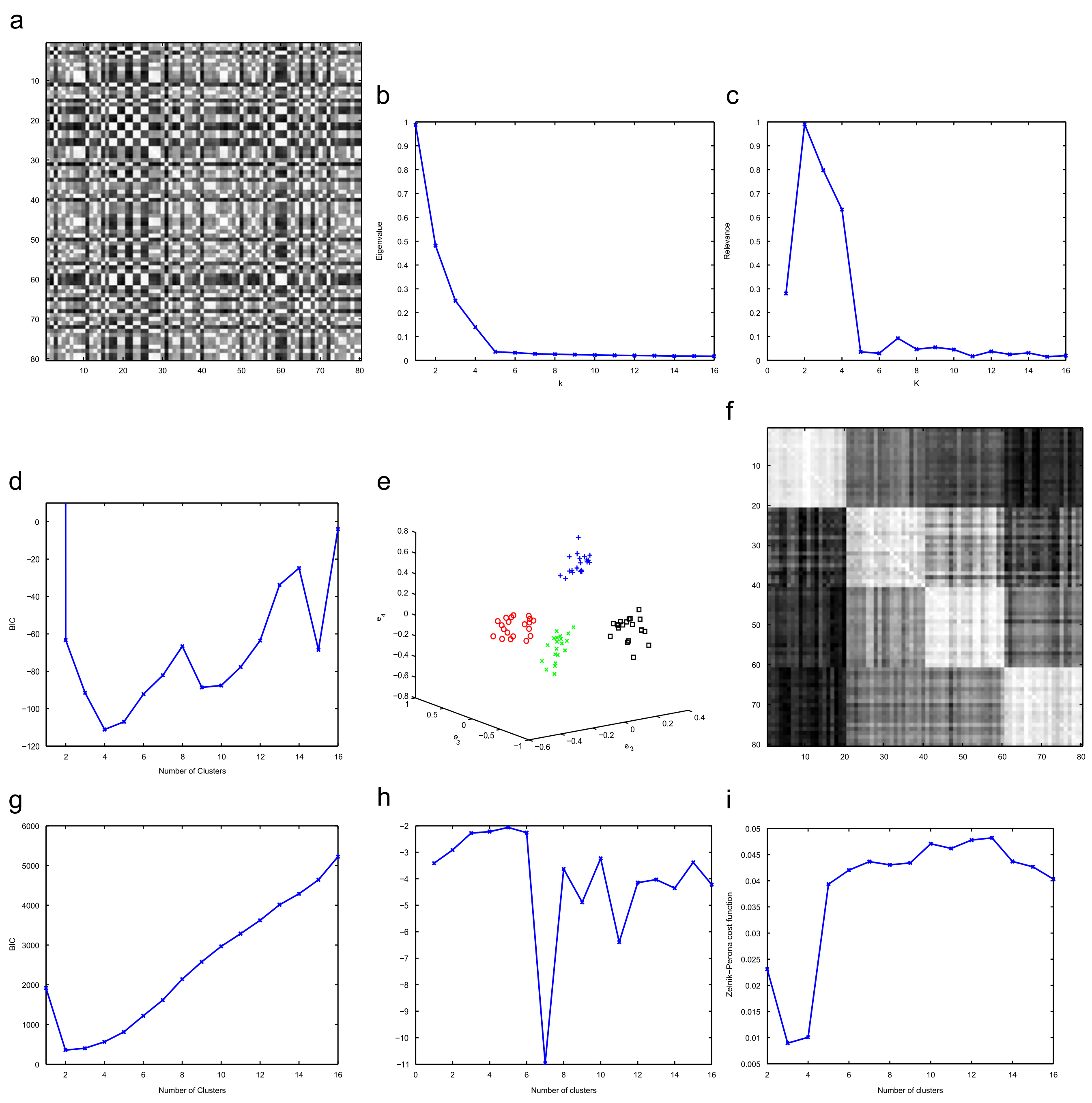

Fig. 4. Clustering a synthetic data set using our spectral clustering algorithm. (a): the normalised affinity matrix constructed by modelling each sequence using an HMM. The eigenvalues of the $K_m = 16$ largest eigenvectors is shown in (b). (c): the learned relevance for the $K_m = 16$ largest eigenvectors. The second, third, and fourth largest eigenvectors were determined as being relevant using Eq. (5). (d) shows the BIC model selection results; the optimal cluster number was determined as 4. (e): the 80 data sample plotted using the three relevant eigenvectors, i.e. $\mathbf{e}_2$, $\mathbf{e}_3$, and $\mathbf{e}_4$. Points corresponding to different classes are colour coded in (e) according to the classification result. (f): the affinity matrix re-ordered according to the result of our clustering algorithm. (g)–(i) show that the cluster number was estimated as 2, 5, and 3, respectively, using three alternative approaches.

by random noise with the identical uniform distribution as the previous experiment. Fig. 6 shows that the cluster number was correctly determined as 4 and all data points were grouped into the right clusters. A data set of a more extreme distribution was also clustered using our algorithm. In this experiment, the size of the largest cluster is 54 times bigger than that of the smallest one. Fig. 7 show that the number clusters was determined as 3. As a result, the smallest cluster was merged with another cluster.

Note that in the experiments presented above the data synthesised from the true models were perturbed using noise. In a real-world application, there will also be outliers in the data, i.e. the data generated by the unknown model are replaced by noise. In order to examine the effect of outliers on the proposed

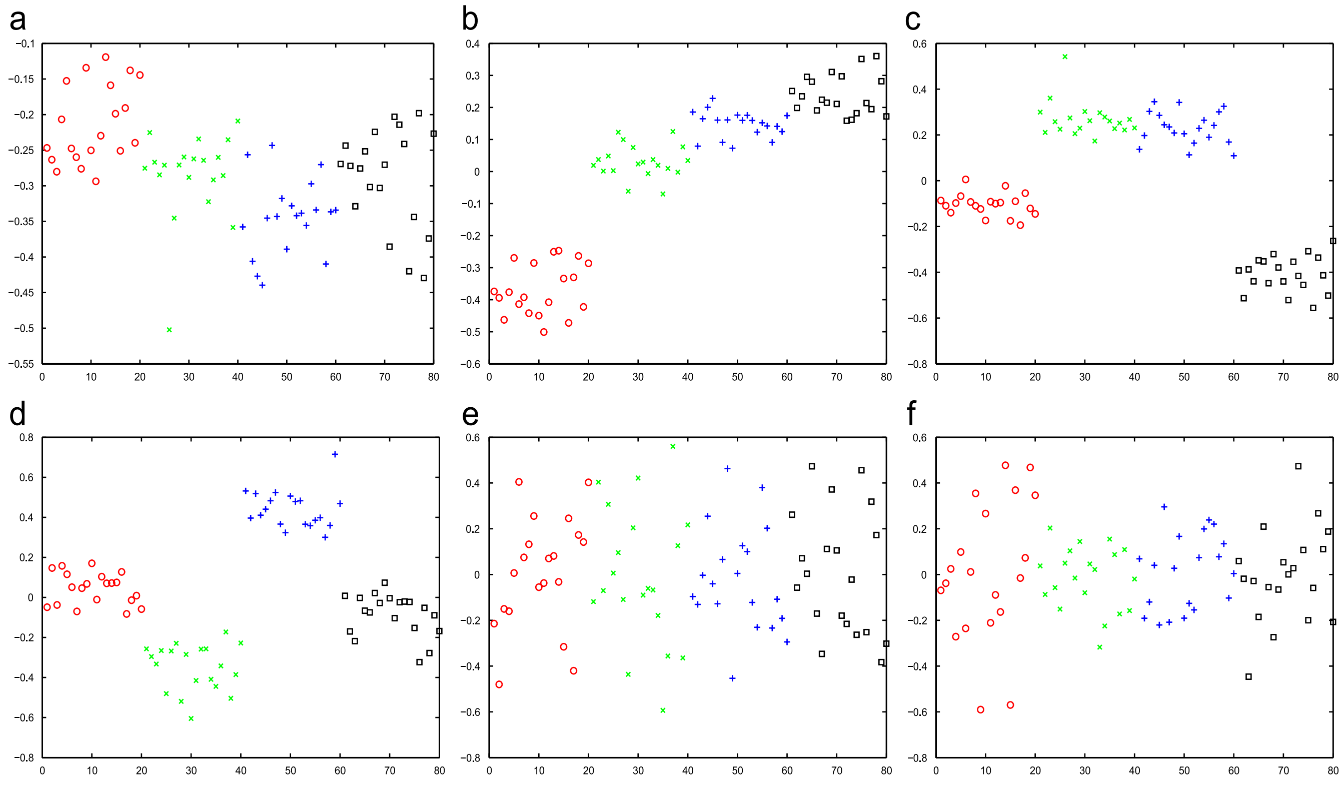

Fig. 5. The distributions of the elements of some eigenvectors of the normalised affinity matrix shown in Fig. 4(a). Elements corresponding to different classes are colour coded according to the classification result. For better illustration we have ordered the points so that points belonging to the same cluster appear consecutively. In all figures, the four clusters are indicated using different colours.

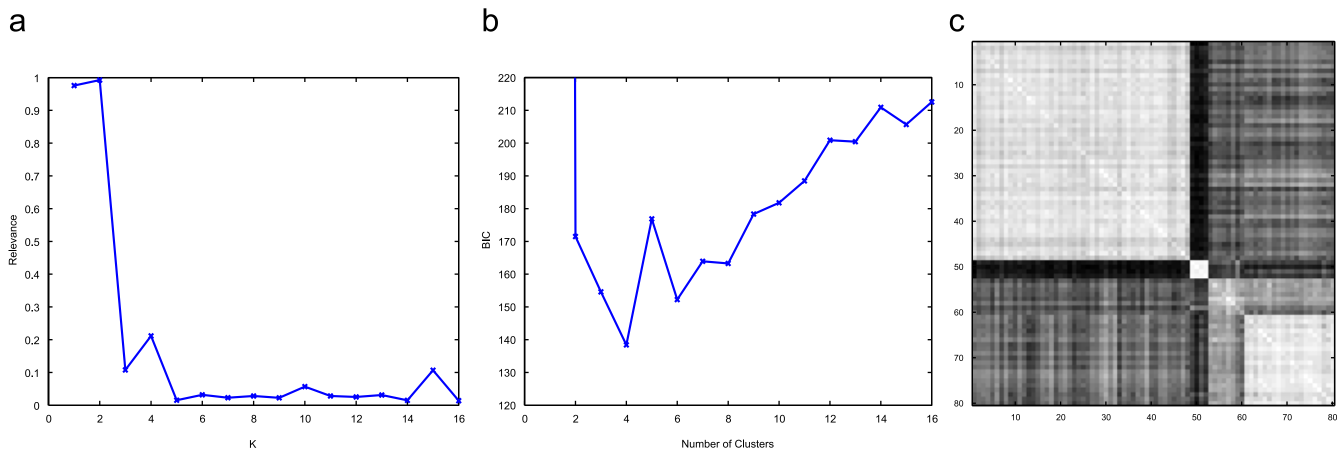

Fig. 6. Clustering an unevenly distributed synthetic data set using our spectral clustering algorithm. The number of data points in each of the four clusters are 4, 8, 20, and 48, respectively. (a): the learned relevance for the $K_m = 16$ largest eigenvectors. The first and second largest eigenvectors were determined as being relevant using Eq. (5). (b) shows the BIC model selection results; the cluster number was determined correctly as 4. (c): the affinity matrix re-ordered according to the result of our clustering algorithm. All four clusters were discovered correctly.

clustering algorithm, two more synthetic data experiments were carried out. In the first experiment, 5% of the data points used in Fig. 4 were replaced with uniformly distributed random noise with a range of [0 6] (e.g. in a sequence of a length 400, 20 data points were randomly chosen and replaced by noise). Fig. 8 indicates that the 5% outliers had little effect on clustering result. In particular, it was automatically determined that there were 4 clusters. After clustering, only 1 data point was grouped into the wrong cluster. In the second experiment, 20% of the data points were substituted using noise. In this experiment, the number of clusters was determined as 3 (see Fig. 8(b)). Fig. 9(c) shows the clustering result. It was found that among the three clusters, one cluster of 19 data points were all generated by one HMM. The other two clusters, sized 32 and

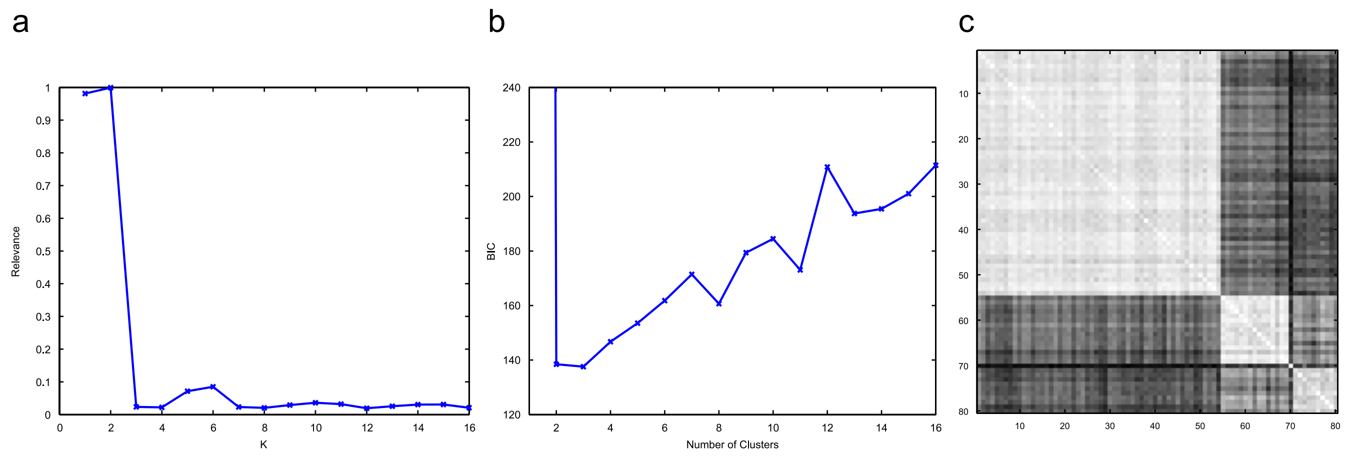

Fig. 7. Clustering a synthetic data set with an extremely uneven distribution using our spectral clustering algorithm. The number of data points in each of the four clusters are 1, 10, 15, and 54, respectively. (a): the learned relevance for the $K_m = 16$ largest eigenvectors. The first and second largest eigenvectors were determined as being relevant using Eq. (5). (b) shows the BIC model selection results; the cluster number was determined as 3. (c): the affinity matrix re-ordered according to the result of our clustering algorithm. The two smallest clusters were merged together.

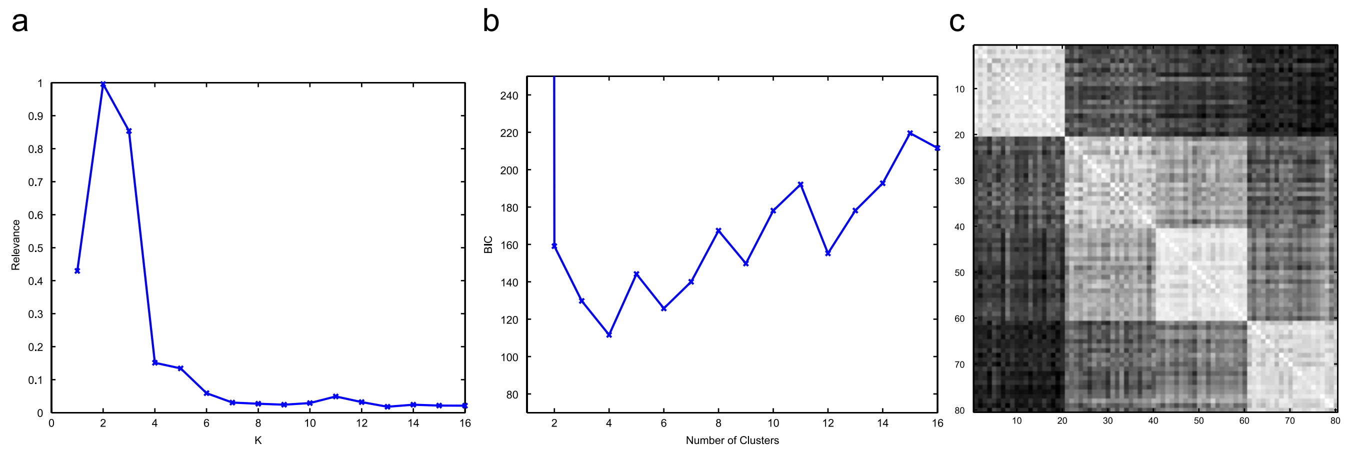

Fig. 8. Clustering a synthetic data set with 5% outliers using our spectral clustering algorithm. (a): the learned relevance for the $K_m = 16$ largest eigenvectors. The second and third largest eigenvectors were determined as being relevant using Eq. (5). (b) shows the BIC model selection results; the cluster number was determined correctly as 4. (c): the affinity matrix re-ordered according to the result of our clustering algorithm.

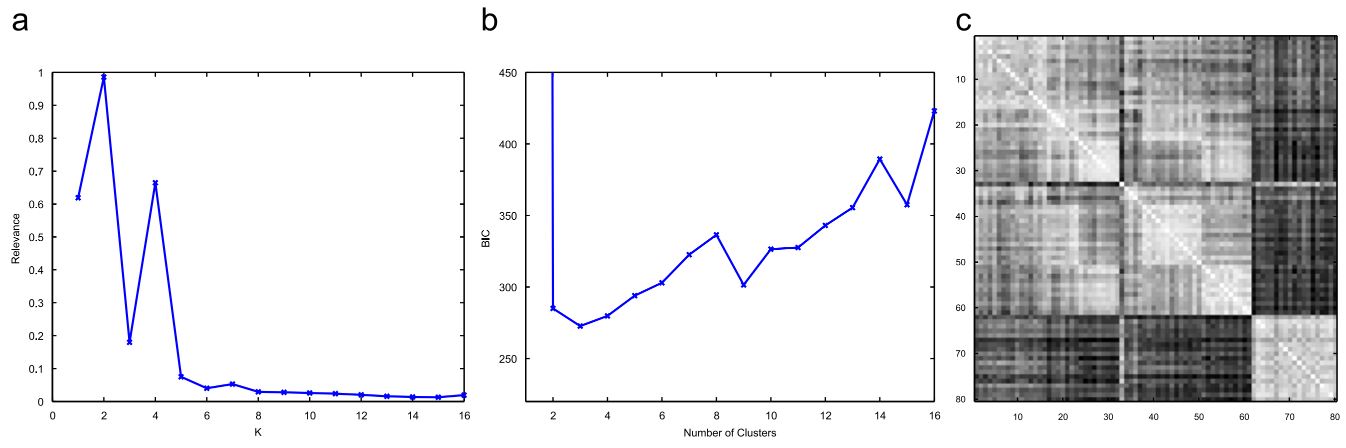

Fig. 9. Clustering a synthetic data set with 20% outliers using our spectral clustering algorithm. (a): the learned relevance for the $K_m = 16$ largest eigenvectors. The first, second, and fourth largest eigenvectors were determined as being relevant using Eq. (5). (b) shows the BIC model selection results; the cluster number was determined as 3. (c): the affinity matrix re-ordered according to the result of our clustering algorithm.

29, respectively, accounted for the other three HMMs in the true model.

In summary, the experiments demonstrate that our spectral clustering algorithm is able to deal with unevenly distributed data sets as long as the size difference between clusters is not too extreme. The algorithm is also shown to be robust to both perturbed noise and outliers.

3. Image segmentation

Our eigenvector selection based spectral clustering algorithm has been applied to image segmentation. A pixel-pixel pairwise affinity matrix **A** is constructed for an image based on the intervening contours method introduced in Ref. [16]. First, for the ith pixel on the image the magnitude of the orientation energy along the dominant orientation is computed as $OE(i)$ using oriented filter pairs. The local support area for the computation of $OE(i)$ has a radius of 30. The value of $OE(i)$ ranges between 0 and infinity. A probability-like variable p_{con} is then computed as

$$p_{con} = 1 - \exp(-OE(i)/\sigma).$$

The value of σ is related to the noise level of the image. It is set as 0.02 in this paper. The value of p_{con} is close to 1 when the orientation energy is much greater than the noise level, indicating the presence of a strong edge. Second, given any pair of pixels in the image, the pixel affinity is computed as

$$A_{ij} = 1 - \max_{x \in M_{ij}} p_{con}(x),$$

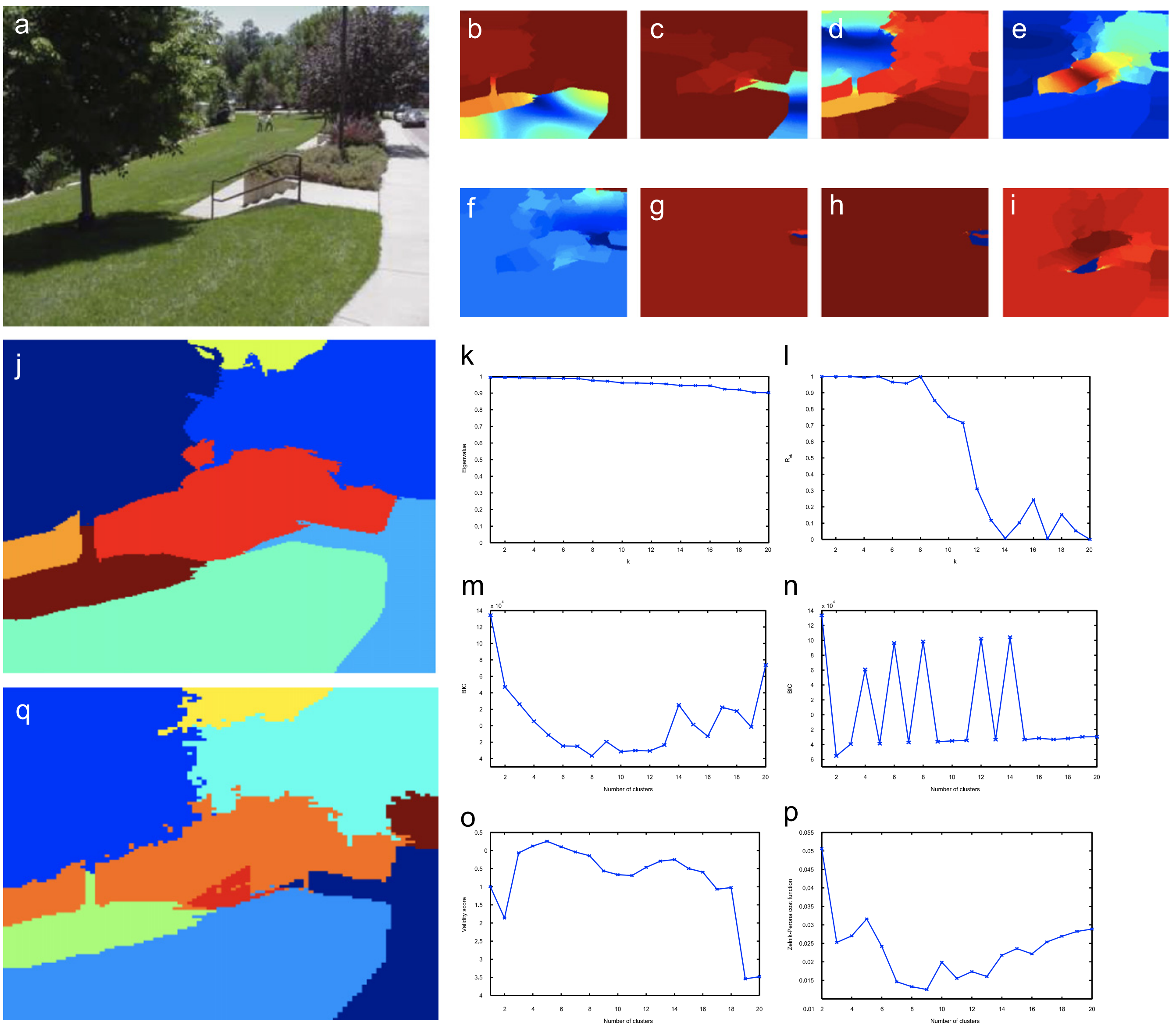

Fig. 10. An example image shown in (a) is segmented as shown in (j). The corresponding eigenvalues of the top 20 eigenvectors are shown in (k). The learned relevance for the 20 largest eigenvectors is shown in (l). (b)–(e) and (f)–(i) show the top 4 most relevant and irrelevant eigenvectors among the 20 largest eigenvectors respectively. (m) and (n) show that K_o was estimated as 8 and 2 with and without relevant eigenvector selection, respectively, using BIC. (o) and (p) show that K_o was estimated as 5 and 9 using the Porikli–Haga validity score and Zelnik–Perona cost function, respectively.

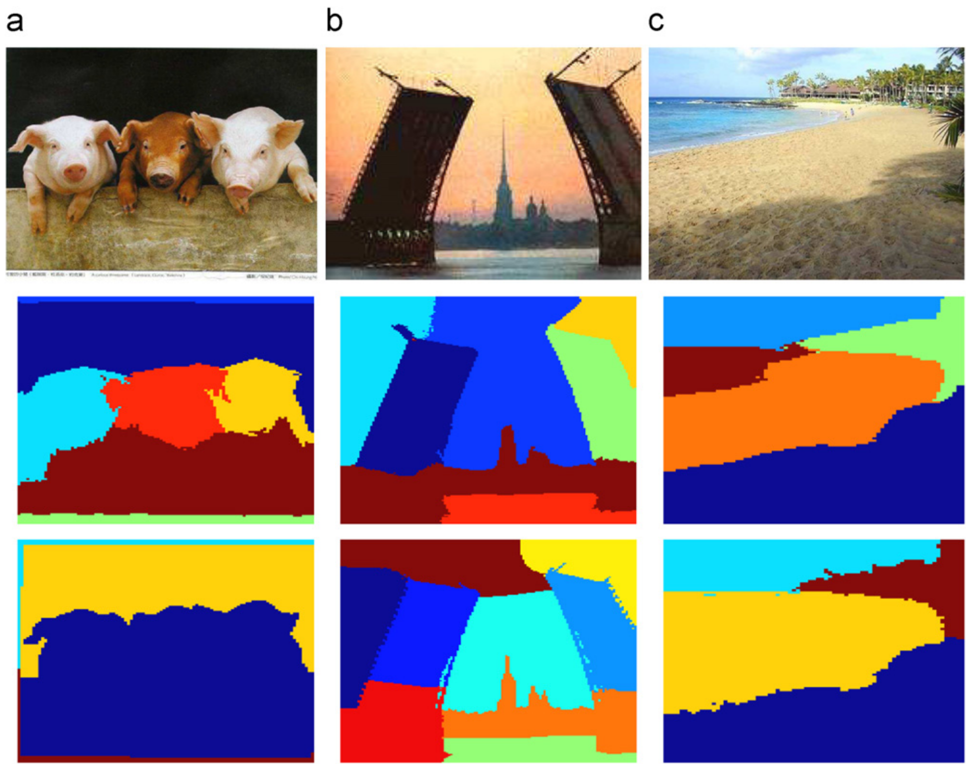

Fig. 11. Further examples of image segmentation. The segmentation results using the proposed algorithm and Zelnik-Manor and Perona's self-tuning spectral clustering algorithm are shown in the middle and bottom row, respectively. From left to right, the optimal number of segments K_o were determined as 7, 7, 5 using our algorithm. They were estimated as 4, 9, 4 using the self-tuning approach.

where M_{ij} are those local maxima along the line connecting pixels i and j. The dissimilarity between pixels i and j is high (A_{ij} is low) if the orientation energy along the line between the two pixels is strong (i.e. the two pixels are on the different sides of a strong edge). The cues of contour and texture differences are exploited simultaneously in forming the affinity matrix. The spectral clustering algorithm using such an affinity matrix aims to partition an image into regions of coherent brightness and texture. Note that colour information is not used this formulation.

Fig. 10 illustrates in detail how our algorithm works for image segmentation. Given the original image in Fig. 10(a), the maximum number of segments was set to 20. The associated eigenvalues are shown in Fig. 10(k). Note that all the top 20 eigenvectors have an eigenvalue of magnitude close to 1. Figs. 10(b)–(i) show the distributions of elements for a number of the top 20 eigenvectors. It can be seen that some eigenvectors contains strong information on the partition of coherent image regions (e.g. $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_5$) while others are rather uninformative (e.g. $\mathbf{e}_{13}, \mathbf{e}_{14}, \mathbf{e}_{17}, \mathbf{e}_{20}$). Fig. 10(k) shows the learned relevance for each of the largest 20 eigenvectors. After eigenvector selection, 12 eigenvectors are kept for clustering. The number of clusters/image segments was determined as 8 by using only the relevant eigenvectors (see Fig. 10(m)). The segmentation result in Fig. 10(j) indicates that the image is segmented into meaningful coherent regions using our algorithm. In comparison, both Porikli and Haga's validity score and BIC without eigenvector selection led to severe under-estimation of the number of image segments. Zelnik-Manor and Perona's self-tuning spectral clustering approach [7][4] yielded comparable results to ours on this particular image (see Figs. 10(p) and (q)).

Our algorithm has been tested on a variety of natural images. Figs. 11 and 12 show some segmentation results. K_m was set to 20 in all our experiments. Our results show that (1) regions corresponding to objects or object parts are clearly separated from each other, and (2) the optimal numbers of image segments estimated by our algorithm reflect the complexity of the images accurately. We also estimated the number of images segments without eigenvector selection based on BIC. The estimated cluster numbers were either 2 or 3 for the images presented in Figs. 11 and 12. This supports our argument that selecting the relevant eigenvectors is critical for spectral clustering. The proposed algorithm was also compared with the se lf-tuning spectral clustering approach introduced in Ref. [7]. It can been seen from Figs. 11 and 12 that in comparison, our approach led to more accurate estimation of the number of image segments and better segmentation. In particular, in Figs. 11(a) and (c) and Fig. 12(b), the self-tuning approach underestimated the number of image segments. This resulted in regions from different objects being grouped into a single segment. In the examples shown in Fig. 11(b) and Figs. 12(a) and (b), although the two approaches obtained similar numbers of clusters, the

[4] Courtesy of L. Zelnik-Manor for providing the code.

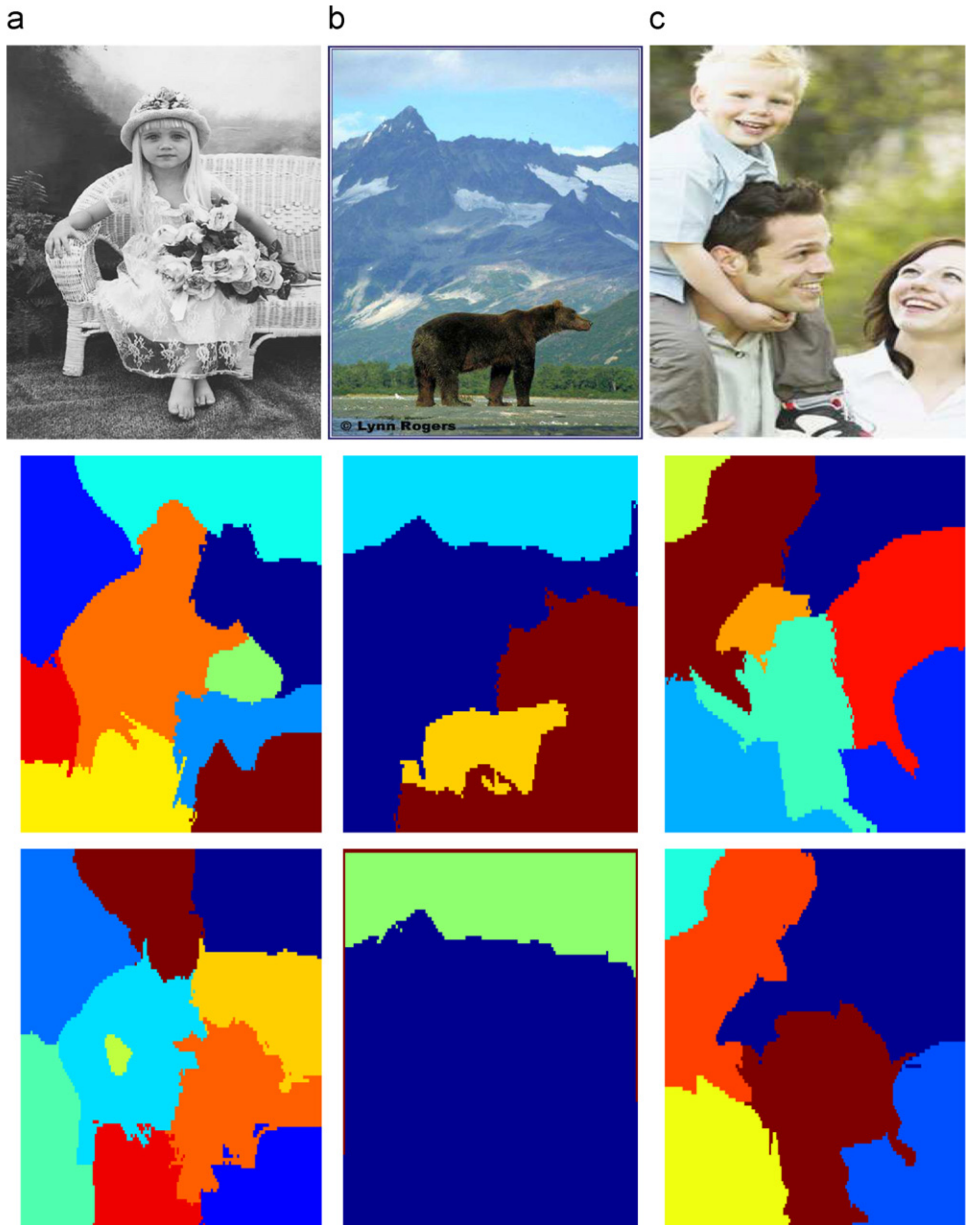

Fig. 12. Another set of examples of image segmentation. The segmentation results using the proposed algorithm and Zelnik-Manor and Perona's self-tuning spectral clustering algorithm are shown in the middle and bottom row, respectively. From left to right, the optimal number of segments K_o were determined as 9, 4, 8 using our algorithm. They were estimated as 9, 2, 7 using the self-tuning approach.

segmentation results obtained using our algorithm are still superior.

4. Video behaviour pattern clustering

Our spectral clustering algorithm has also been applied to solve the video based behaviour profiling problem in automated CCTV surveillance. Given 24/7 continuously recorded video or online CCTV input, the goal of automatic behaviour profiling is to learn a model that is capable of detecting unseen abnormal behaviour patterns whilst recognising novel instances of expected normal behaviour patterns. To achieve the goal, the natural grouping of behaviour patterns captured in a training data set is first discovered using the proposed spectral clustering algorithm. These groupings form behaviour classes. A behaviour model is then constructed based on the clustering result. This model can be employed to detect abnormal behaviours and recognise normal behaviours.

4.1. Behaviour pattern representation

A continuous surveillance video $\mathbf{V}$ is first segmented into N segments $\mathbf{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n, \ldots, \mathbf{v}_N\}$ so that each segment contains approximately a single behaviour pattern. The nth video segment $\mathbf{v}_n$ consisting of T_n image frames is represented as $\mathbf{v}_n = \{\mathbf{I}_{n1}, \ldots, \mathbf{I}_{nt}, \ldots, \mathbf{I}_{nT_n}\}$, where $\mathbf{I}_{nt}$ is the tth image frame.

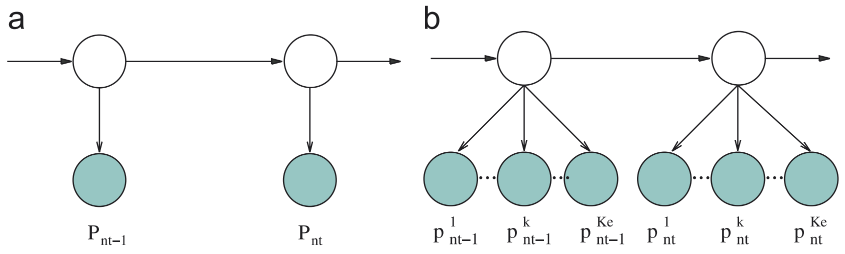

Fig. 13. Modelling a behaviour pattern $\mathbf{P}_n = \{\mathbf{p}_{n1}, \ldots, \mathbf{p}_{nt}, \ldots, \mathbf{p}_{nT_n}\}$ where $\mathbf{p}_{nt} = \{p^1_{nt}, \ldots, p^k_{nt}, \ldots, p^{K_e}_{nt}\}$ using an HMM and a MOHMM. Observation nodes are shown as shaded circles and hidden nodes as clear circles.

Depending on the nature of the video sequence to be processed, various segmentation approaches can be adopted. Since we are focusing on surveillance video, the most commonly used shot change detection based segmentation approach is not appropriate. In a not-too-busy scenario, there are often non-activity gaps between two consecutive behaviour patterns which can be utilised for activity segmentation. In the case where obvious non-activity gaps are not available, an on-line segmentation algorithm proposed in Ref. [17] can be adopted. Alternatively, the video can be simply sliced into overlapping segments with a fixed time duration [18].

A discrete event based approach is then adopted for behaviour representation [19,20]. Firstly, an adaptive Gaussian mixture background model [21] is adopted to detect foreground pixels which are modelled using pixel change history (PCH) [22]. Secondly, the foreground pixels in a vicinity are grouped into a blob using the connected component method. Each blob with its average PCH value greater than a threshold is then defined as a scene-event. A detected scene-event is represented as a 7-dimensional feature vector

$$\mathbf{f} = [\bar{x}, \bar{y}, w, h, R_f, M_px, M_py], \tag{9}$$

where $(\bar{x}, \bar{y})$ is the centroid of the blob, (w, h) is the blob dimension, R_f is the filling ratio of foreground pixels within the bounding box associated with the blob, and (M_px, M_py) are a pair of first order moments of the blob represented by PCH. Among these features, $(\bar{x}, \bar{y})$ are location features, (w, h) and R_f are principally shape features but also contain some indirect motion information, and (M_px, M_py) are motion features capturing the direction of object motion.

Thirdly, classification is performed in the 7D scene-event feature space using a GMM. The number of scene-event classes K_e captured in the videos is determined by automatic model order selection based on BIC [23]. The learned GMM is used to classify each detected event into one of the K_e event classes. Finally, the behaviour pattern captured in the nth video segment $\mathbf{v}_n$ is represented as a feature vector $\mathbf{P}_n$, given as

$$\mathbf{P}_n = [\mathbf{p}_{n1}, \ldots, \mathbf{p}_{nt}, \ldots, \mathbf{p}_{nT_n}], \tag{10}$$

where T_n is the length of the nth video segment and the tth element of $\mathbf{P}_n$ is a K_e dimensional variable:

$$\mathbf{p}_{nt} = [p^1_{nt}, \ldots, p^k_{nt}, \ldots, p^{K_e}_{nt}]. \tag{11}$$

$\mathbf{p}_{nt}$ corresponds to the tth image frame of $\mathbf{v}_n$ where p^k_{nt} is the posterior probability that an event of the kth event class has occurred in the frame given the learned GMM. If an event of the kth class is detected in the tth image frame of $\mathbf{v}_n$, we have $0 < p^k_{nt} \leqslant 1$; otherwise, we have $p^k_{nt} = 0$. Note that multiple events from different event classes can be detected simultaneously in a single frame.

4.2. *Forming a behaviour pattern affinity matrix*

Consider a training data set $\mathbf{D} = \{\mathbf{P}_1, \ldots, \mathbf{P}_n, \ldots, \mathbf{P}_N\}$ consisting of N behaviour patterns, where $\mathbf{P}_n$ is the nth behaviour pattern feature vector as defined above. To cluster the data using the proposed spectral clustering algorithm, a similarity measure between a pair of behaviour patterns needs to be defined. Note that the feature vectors $\mathbf{P}_n$ can be of different lengths; therefore dynamic warping is required before they can be compared with. A definition of a distance/affinity metric among these variable length feature vectors is not simply Euclidean therefore requires a nontrivial string similarity measure.

We utilise dynamic Bayesian networks (DBNs) to provide a dynamic representation of each behaviour pattern feature vector in order to both address the need for dynamic warping and provide a string similarity metric. More specifically, each behaviour pattern in the training set is modelled using a DBN. To measure the affinity between two behaviour patterns represented as $\mathbf{P}_i$ and $\mathbf{P}_j$, two DBNs denoted as $\mathbf{B}_i$ and $\mathbf{B}_j$ are trained on $\mathbf{P}_i$ and $\mathbf{P}_j$, respectively, using the EM algorithm [12,24]. Similar to the synthetic data case (see Section 2), the affinity between $\mathbf{P}_i$ and $\mathbf{P}_j$ is then computed as:

$$A_{ij} = \frac{1}{2}\left\{\frac{1}{T_j}\log P(\mathbf{P}_j|\mathbf{B}_i) + \frac{1}{T_i}\log P(\mathbf{P}_i|\mathbf{B}_j)\right\}, \tag{12}$$

where $P(\mathbf{P}_j|\mathbf{B}_i)$ is the likelihood of observing $\mathbf{P}_j$ given $\mathbf{B}_i$, and T_i and T_j are the lengths of $\mathbf{P}_i$ and $\mathbf{P}_j$, respectively.

DBNs of different topologies can be used. However, it is worth pointing out that since a DBN needs to be learned for every single behaviour pattern in the training data set which could be short in duration, a DBN with less number of parameters is desirable. In this work, we employ a multi-observation hidden Markov model (MOHMM) [19] shown in Fig. 13(b). Compared to a standard HMM (see Fig. 13(a)), the observational space is factorised by assuming that each observed feature (p^k_{nt}) is independent of each other. Consequently, the

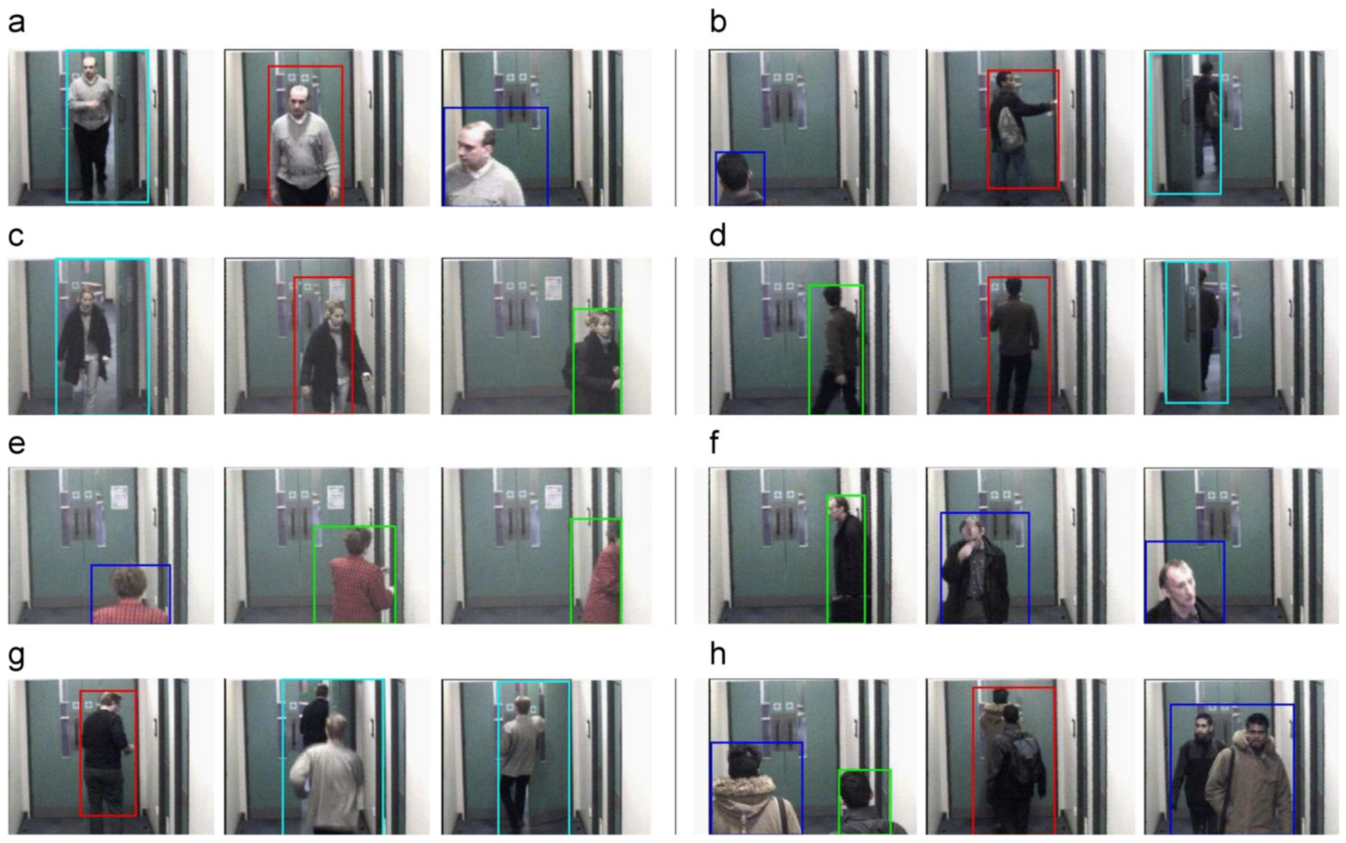

Fig. 14. Examples of behaviour patterns captured in a corridor entrance scene. (a)–(f) show image frames of commonly occurred behaviour patterns belonging to the 6 behaviour classes listed in Table 1. (g) and (h) show examples of rare behaviour patterns captured in the scene. (g): one person entered the office following another person without using an entry card. (h): two people left the corridor after a failed attempt to enter the door. The four classes of events detected automatically, 'entering/leaving the near end of the corridor', 'entering/leaving the entry-door', 'entering/leaving the side-doors', and 'in corridor with the entry door closed', are highlighted in the image frames using bounding boxes in blue, cyan, green and red, respectively.

number of parameters for describing an MOHMM is much lower than that for an HMM ($2K_eN_s+N_s^2-1$ for an MOHMM and $(K_e^2+3K_e)N_s/2+N_s^2-1$ for an HMM). Note that in this paper, the number of hidden states for the MOHMM is set to K_e, i.e. the number of event classes. This is reasonable because the value of N_s should reflect the complexity of a behaviour pattern, so should the value of K_e.

4.3. Constructing a behaviour model

Using our relevant eigenvector selection based spectral clustering algorithm described in Section 2, the N behaviour patterns in the training set are clustered into K_o behaviour pattern classes. To build a model for the observed/expected behaviour, we first model the kth behaviour class using an MOHMM $\mathbf{B}_k$. The parameters of $\mathbf{B}_k$, $\boldsymbol{\theta}_{\mathbf{B}_k}$ are estimated using all the patterns in the training set that belong to the kth class. A behaviour model $\mathbf{M}$ is then formulated as an mixture of the K_o MOHMMs. Given an unseen behaviour pattern, represented as a behaviour pattern feature vector $\mathbf{P}$, the likelihood of observing $\mathbf{P}$ given $\mathbf{M}$ is

$$P(\mathbf{P}|\mathbf{M})=\sum_{k=1}^{K}\frac{N_k}{N}P(\mathbf{P}|\mathbf{B}_k), \tag{13}$$

Table 1
Six classes of commonly occurred behaviour patterns in the entrance scene

C1	From the office area to the near end of the corridor
C2	From the near end of the corridor to the office area
C3	From the office area to the side-doors
C4	From the side-doors to the office area
C5	From the near end of the corridor to the side-doors
C6	From the side-doors to the near end of the corridor

where N is the total number of training behaviour patterns and N_k is the number of patterns that belong to the kth behaviour class.

Once the behaviour model is constructed, an unseen behaviour pattern is detected as abnormal if

$$P(\mathbf{P}|\mathbf{M})<Th_A, \tag{14}$$

where Th_A is a threshold. When an unseen behaviour pattern is detected as normal, the normal behaviour model $\mathbf{M}$ can also be used for recognising it as one of the K behaviour pattern classes learned from the training set. More specifically, an unseen behaviour pattern is assigned to the $\hat{k}$th behaviour class when

$$\hat{k}=\arg\max_k\{P(\mathbf{P}|\mathbf{B}_k)\}. \tag{15}$$

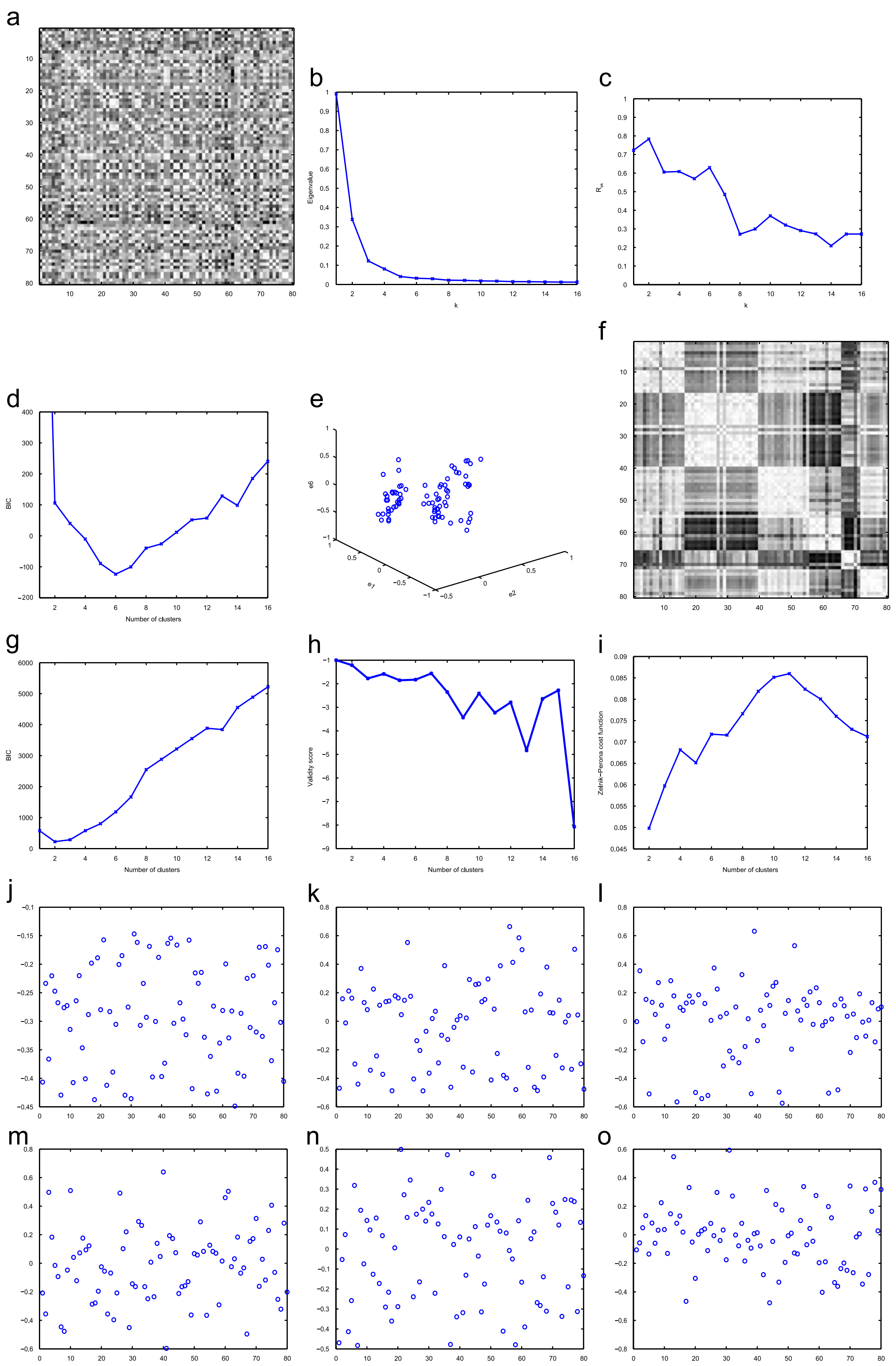

Fig. 15. An example of behaviour pattern clustering. (c) shows that the top 6 largest eigenvectors were determined as relevant features for clustering. (d) and (g) show the number of behaviour classes was determined as 6 and 2 using BIC with and without relevant eigenvector selection, respectively. (h) and (i) show that using Porikli and Haga's validity score and Zelnik-Manor and Perona's cost function, the class number was estimated as 1 and 2, respectively.

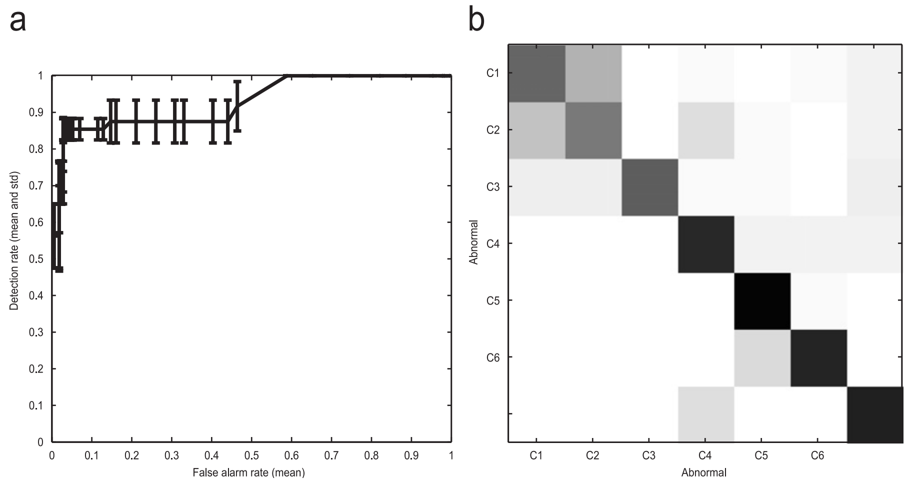

Fig. 16. The performance of abnormality detection and behaviour recognition for the corridor scene. (a): the mean and ±1 standard deviation of the ROC curves for abnormality detection obtained over 20 trials. (b): confusion matrix for behaviour recognition. Each row represents the probabilities of that class being confused with all the other classes averaged over 20 trials. The main diagonal of the matrix shows the fraction of patterns correctly recognised and is as follows: [.68 .63 .72 .84 .92 .85 .85].

4.4. Experiments

Experiments were conducted on an entrance surveillance scenario. A CCTV camera was mounted on the ceiling of an office entry corridor, monitoring people entering and leaving the office area (see Fig. 14). The office area is secured by an entrance-door which can only be opened by scanning an entry card on the wall next to the door (see middle frame in Fig. 14(b)). Two side-doors were also located at the right hand side of the corridor. People from both inside and outside the office area have access to those two side-doors. Typical behaviours occurring in the scene would be people entering or leaving either the office area or the side-doors, and walking towards the camera. Each behaviour pattern would normally last a few seconds. For this experiment, a data set was collected over 5 different days consisting of 6 h of video totalling 432 000 frames captured at 20 Hz with 320×240 pixels per frame. This data set was then segmented into sections separated by any motionless intervals lasting for more than 30 frames. This resulted in 142 video segments of actual behaviour pattern instances. Each segment has on average 121 frames with shortest 42 and longest 394.

Model training: A training set consisting of 80 video segments was randomly selected from the overall 142 segments without any behaviour class labelling of the video segments. The remaining 62 segments were used for testing the trained model later. This model training exercise was repeated 20 times and in each trial a different model was trained using a different random training set. This is in order to avoid any bias in the abnormality detection and normal behaviour recognition results.

Discrete events were detected and classified using automatic model order selection in clustering, resulting in four classes of events corresponding to the common constituents of all behaviours in this scene: 'entering/leaving the near end of the corridor', 'entering/leaving the entry-door', 'entering/leaving the side-doors', and 'in corridor with the entry door closed'. Examples of detected events are shown in Fig. 14 using colour-coded bounding boxes. It is noted that due to the narrow view nature of the scene, differences between the four common events are rather subtle and can be mis-identified based on local information (space and time) alone, resulting in errors in event detection. The fact that these events are also common constituents to different behaviour patterns means that local events treated in isolation hold little discriminative information for behaviour profiling.

The upper limit of the behaviour class number K_m was set to 16 in the experiments. Over the 20 trials, on average 6 eigenvectors were automatically determined as being relevant for clustering with smallest 4 and largest 9. The number of clusters for each training set was determined automatically as 6 in every trial. It is observed that each discovered data cluster mainly contained samples corresponding to one of the 6 behaviour classes listed in Table 1 (on average, 85% of the data samples in each cluster belong to one of the 6 behaviour classes). In comparison, all three alternative approaches, including BIC without eigenvector selection, Porikli and Haga's validity score, and Zelnik-Manor and Perona's cost function, tended to severely underestimate the class number. Fig. 15 shows an example of discovering behaviour classes using spectral clustering. Compared to the synthetic data and image segmentation data, the behaviour pattern data are much more noisy and difficult to group. This is reflected by the fact that the elements of the eigenvectors show less information about the data grouping (see Figs. 15(j)–(o)). However, using only the relevant/informative eigenvectors, our algorithm can still discover the behaviour classes correctly. Based on the clustering result, a normal behaviour model was constructed as a mixture of MOHMMs as described in Section 4.3.

Abnormality detection: To measure the performance of the learned models on abnormality detection, each behaviour pattern in the testing sets was manually labelled as normal if there were similar patterns in the corresponding training sets and abnormal otherwise. On average, there were 7 abnormal behaviour patterns in each testing set which consists of 62 behaviour patterns. The detection rate and false alarm rate of abnormality detection are shown in the form of an ROC curve. Fig. 16(a) shows that high detection rate and low false alarm rate have been achieved. Th_A (see Eq. (14)) was set to -0.2 in the rest results unless otherwise specified, which gave an abnormality detection rate of $85.4 \pm 2.9\%$ and false alarm rate of $6.1 \pm 3.1\%$.

Recognition of normal behaviours: To measure the performance of behaviour recognition results, the normal behaviour patterns in the testing sets were manually labelled into different behaviour classes. A normal behaviour pattern was recognised correctly if it was detected as normal and classified into the right behaviour class. The behaviour recognition results are illustrated as a confusion matrix shown in Fig. 16(b). Overall, the recognition rates had a mean of 77.9% and standard deviation of 4.8% for the 6 behaviour classes over 20 trials.

Our experiments show that given a challenging dynamic visual data clustering problem, the proposed clustering algorithm is able to determine the correct number of clusters and groups the data into behaviour classes accurately. In comparison, alternative approaches tend to severely under-estimate the number of clusters (see Fig. 15). Our experiments also demonstrate that our behaviour model constructed based on the clustering result can be used for successfully detecting abnormal behaviour patterns and recognising normal ones.

5. Discussion and conclusion

In this paper, we analysed and demonstrated that: (1) not every eigenvector of a data affinity matrix is informative and relevant for clustering; (2) eigenvector selection is critical because using uninformative/irrelevant eigenvectors could lead to poor clustering results; and (3) the corresponding eigenvalues cannot be used for relevant eigenvector selection given a realistic data set. Motivated by the analysis, a novel spectral clustering algorithm was proposed which differs from previous approaches in that only informative/relevant eigenvectors are employed for determining the number of clusters and performing clustering. The key element of the proposed algorithm is a simple but effective relevance learning method which measures the relevance of an eigenvector according to how well it can separate the data set into different clusters. Our algorithm was evaluated using synthetic data sets as well as real-world data sets generated from two challenging visual learning problems. The results demonstrated that our algorithm is able to estimate the cluster number correctly and reveal natural grouping of the input data/patterns even given sparse and noisy data.

It is interesting to note that eigen-decomposition of a similarity matrix is similar to principal component analysis (PCA) in the sense that both aim to reduce the dimensionality of the feature space for data representation. Each row of an affinity matrix can be used for representing one data point. In doing so N data points are represented in an N-dimensional feature space. Although no information will be lost in this representation, the clustering process will suffer from the 'curse of dimensionality problem'. After eigen-decomposition, if all N eigenvectors are used for data representation, the same problem remains. Therefore, all spectral clustering algorithms must perform eigenvector selection. However, the selection criteria used by previous approaches are simple: either the largest K_{true} is selected if K_{true} is known or the largest K_m are selected where K_m is considered to be safely larger than the unknown K_{true}. In this paper we have demonstrated that the previous criteria would not work given realistic data. To solve the problem, we have proposed a completely different criterion. Specifically eigenvector selection is performed based on measuring how informative/relevant each eigenvector is.

We chose different K_m in different experiments presented in the paper. As mentioned earlier, K_m is a number considered to be safely larger than the true model order K_{true}. Then a problem will arise: how to chose K_m when you have no idea at all on what the true model order is. Fortunately, there is a solution. As a rule of thumb, given N data points generated from a model of C_k parameters, if $N < 5C_k$ then there is no hope that the data can be modelled or clustered properly. In our approach, K_{true} clusters will be modelled using at least $3K_{true} - 1$ parameters (when only one eigenvector is relevant and modelled by a Gaussian mixture model). Therefore $K_m = N/5$ would be a reasonable choice for a number that is safely larger than K_{true}. Otherwise, the data set will be too sparse to cluster. In our video behaviour pattern clustering experiment, we know nothing about how many different classes of behaviour patterns there could be, so we used $K_m = N/5$. In the case of image segmentation, N is in the order of 100 000; so using that equation is inappropriate. However we often know roughly how many regions there will be in an image in a normal case. We therefore chose $K_m = 20$ in the image segmentation experiments.

It is noted in our experiments that following an identical procedure using BIC but without eigenvector selection will lead to an underestimation of the number of clusters. It is not surprising as BIC is known to have the tendency of underestimating the model complexity given sparse and/or noisy data [25,26]. In particular, in a high-dimensional eigenspace spanned by the top K_m eigenvectors, the GMM used for modelling data distribution in the eigenspace would suffers from the 'curse of dimensionality' problem, therefore can only be learned poorly. This contributes to the underestimation of cluster numbers. This problem will remain even if the data set is free of noise and the clusters are well-separated. The approaches proposed in Refs. [6] and [7] are less likely to suffer from the same problem because no explicit model fitting is involved. However, they are still sensitive to the presence of noise since all noise-corrupted eigenvectors are used without discrimination. This is why their performance on clustering real-world data is inferior to that of our algorithm.

It is worth pointing out that the distribution of the elements of an eigenvector depends on the data distribution in the original feature space. The latter will also affect the way noise is propagated in the eigenspace. Since the data distribution in the original feature space is unknown and often difficult to be expressed in an analytical form, an analysis of eigenspace distribution and error propagation is nontrivial. Our eigenvector selection algorithm is essentially a data-driven approach which is independent of the data distribution in the original feature space. This is one of desirable characteristic of the algorithm.

In this paper, the BIC was used with GMM to estimate the number of clusters. Numerous alternative model selection criteria exist for a GMM although BIC is arguably the most commonly used one [26,27]. The ongoing work includes investigating the effect of employing different model selection criteria on the performance of our algorithm.

References

[1] Y. Weiss, Segmentation using eigenvectors: a unifying view, in: IEEE International Conference on Computer Vision, 1999, pp. 975–982.
[2] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[3] S. Yu, J. Shi, Multiclass spectral clustering, in: IEEE International Conference on Computer Vision, 2003, pp. 313–319.
[4] P. Smyth, Clustering sequence with hidden markov models, Adv. Neural Inf. Process. Syst. 9 (1997) 648–654.
[5] A. Ng, M. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, Adv. Neural Inf. Process. Syst. 14 (2001) 849–856.
[6] F. Porikli, T. Haga, Event detection by eigenvector decomposition using object and frame features, in: IEEE Conference on Computer Vision and Pattern Recognition Workshop, 2004, pp. 114–121.
[7] L. Zelnik-Manor, P. Perona, Self-tuning spectral clustering, Adv. Neural Inf. Process. Syst. 2004.
[8] M. Fiedler, Algebraic connectivity of graphs, Czech. Math. J. 23 (1973) 298–305.
[9] F. Chung, Number 92 in CBMS Regional Conference Series in Mathematics, American Mathematical Society, Providence, RI, 1997.
[10] A. Blum, P. Langley, Selection of relevant features and examples in machine learning, Artif. Intell. 97 (1997) 245–271.
[11] J. Dy, C. Brodley, A. Kak, L. Broderick, A. Aisen, Unsupervised feature selection applied to content-based retrival of lung images, IEEE Trans. Pattern Anal. Mach. Intell. 25 (2003) 373–378.
[12] A. Dempster, N. Laird, D. Rubin, Maximum-likelihood from incomplete data via the EM algorithm, J. Roy. Statist. Soc. B 39 (1977) 1–38.
[13] L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE 77 (2) (1989) 257–286.
[14] A. Panuccio, M. Bicego, V. Murino, A hidden markov model-based approach to sequential data clustering, in: Proceedings of the Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition, London, UK, 2002, Springer, Berlin, pp. 734–742.
[15] F. Porikli, Trajectory distance metric using hidden markov model based representation, in: The Sixth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, 2002.
[16] J. Malik, S. Belongie, T. Leung, J. Shi, Contour and texture analysis for image segmentation, Int. J. Comput. Vision 43 (1) (2001) 7–27.
[17] T. Xiang, S. Gong, Activity based video content trajectory representation and segmentation, in: British Machine Vision Conference, 2004.
[18] H. Zhong, J. Shi, M. Visontai, Detecting unusual activity in video, in: IEEE Conference on Computer Vision and Pattern Recognition, 2004, pp. 819–826.
[19] S. Gong, T. Xiang, Recognition of group activities using dynamic probabilistic networks, in: IEEE International Conference on Computer Vision, 2003.
[20] T. Xiang, S. Gong, Beyond tracking: modelling activity and understanding behaviour, Int. J. Comput. Vision 67 (2006) 21–51.
[21] C. Stauffer, W. Grimson, Learning patterns of activity using real-time tracking, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 747–758.
[22] T. Xiang, S. Gong, D. Parkinson, Autonomous visual events detection and classification without explicit object-centred segmentation and tracking, in: British Machine Vision Conference, 2002, pp. 233–242.
[23] G. Schwarz, Estimating the dimension of a model, Ann. Statist. 6 (1978) 461–464.
[24] Z. Ghahramani, Learning dynamic bayesian networks, in: Adaptive Processing of Sequences and Data Structures, Lecture Notes in Artificial Intelligence, 1998, pp. 168–197.
[25] S. Roberts, D. Husmeier, I. Rezek, W. Penny, Bayesian approaches to Gaussian mixture modelling, IEEE Trans. Pattern Anal. Mach. Intell. 20 (11) (1998) 1133–1142.
[26] M. Figueiredo, A.K. Jain, Unsupervised learning of finite mixture models, IEEE Trans. Pattern Anal. Mach. Intell. 24 (3) (2002) 381–396.
[27] C. Biernacki, G. Celeux, G. Govaert, Assessing a mixture model for clustering with the integrated completed likelihood, IEEE Trans. Pattern Anal. Mach. Intell. 22 (7) (2000) 719–725.

About the Author—TAO XIANG is a Lecturer at the Department of Computer Science, Queen Mary, University of London. Dr Xiang was awarded his Ph.D. in Electrical and Computer Engineering from the National University of Singapore in 2002, which involved research into 3-D Computer Vision and Visual Perception. He also received his B.Sc. degree in Electrical Engineering from Xi'an Jiaotong University in 1995, and his M.Sc. degree in Electronic Engineering from the Communication University of China (CUC) in 1998. His research interests include computer vision, image processing, statistical learning, pattern recognition, machine learning, and data mining. He has been working recently on topics such as spectral clustering, video based behaviour analysis and recognition, and model order selection for Dynamic Bayesian Networks.

About the Author—SHAOGANG GONG is Professor of Visual Computation at the Department of Computer Science, Queen Mary, University of London and a Member of the UK Computing Research Committee. He heads the Queen Mary Computer Vision Group and has worked in computer vision and pattern recognition for over 20 years, published over 170 papers and a monograph. He twice won the Best Science Prize (1999, 2001) of British Machine Vision Conferences, the Best Paper Award (2001) of IEEE International Workshops on Recognition and Tracking of Faces and Gestures, and the Best Paper Award (2005) of IEE International Symposium on Imaging for Crime Detection and Prevention. He was a recipient of a Queen's Research Scientist Award (1987), a Royal Society Research Fellow (1987, 1988), a GEC-Oxford Fellow (1989), a visiting scientist at Microsoft Research (2001) and Samsung (2003).