

Spectral Clustering with Eigenvector Selection

Tao Xiang* and Shaogang Gong

Department of Computer Science

Queen Mary, University of London, London E1 4NS, UK

{txiang, sgg}@dcs.qmul.ac.uk

Abstract

The task of discovering natural groupings of input patterns, or clustering, is an important aspect 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.

Keywords: Spectral clustering, feature selection, unsupervised learning, image segmentation, video behaviour pattern clustering.

*Corresponding author. Tel: (+44)-(0)20-7882-5201; Fax: (+44)-(0)20-8980-6533

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 data 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 can not 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 [21, 18, 25, 19, 12, 15, 26, 6, 3], 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 [21, 18, 12]. Recently researchers started to tackle the first issue, i.e. determining the cluster number automatically. Smyth [19] proposed to use a Monte-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 [15] employed a validity score computed using the largest eigenvectors¹ of a data affinity matrix to determine the number of clusters for video-based activity classification. Zelnik-Manor and Perona [26] proposed to determine the optimal cluster number through minimising the cost of aligning the top eigenvectors with a canonical coordinate system. The approaches in [15] and [26] 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 take into

¹The largest eigenvectors are eigenvectors that their corresponding eigenvalues are the largest in magnitude.

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

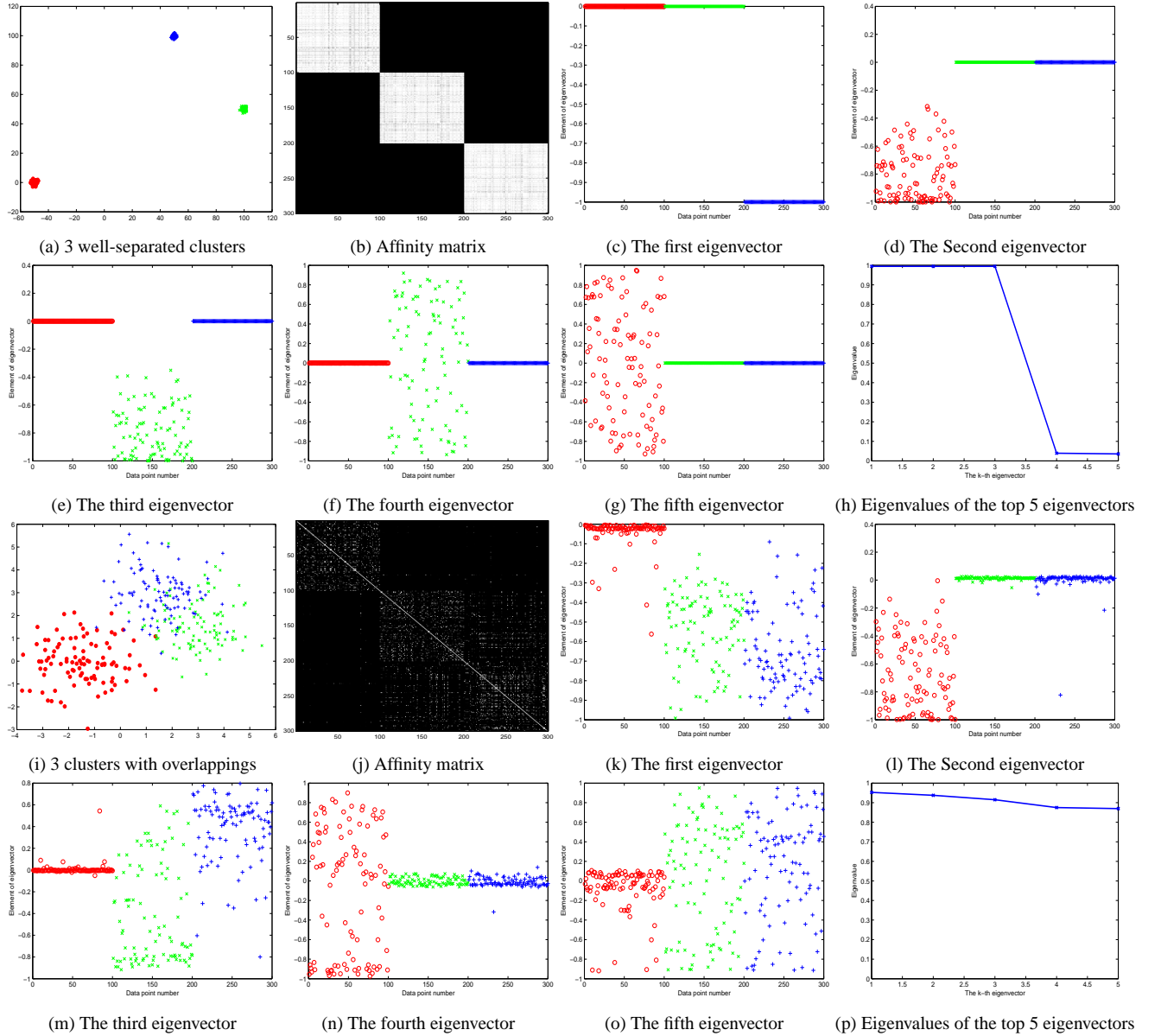


Figure 1: Examples showing that not all eigenvectors are informative for spectral clustering. (a) shows a well-separated 2-D data set consisting of three clusters. The affinity matrix (b) shows clear block structure. (c)-(e) show that the top 3 eigenvectors contain useful information about the natural 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.

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 [12]. 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. Figures 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 [6, 3, 12]. This is shown clearly in Figure 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 [2, 5]. 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 [15] and [26]. 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

$$\mathbf{D} = \{\mathbf{f}_1, \dots, \mathbf{f}_n, \dots, \mathbf{f}_N\}, \quad (1)$$

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 i th and j th 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 [21, 18] that it is more desirable to perform clustering based on the eigenvectors of the normalised affinity matrix $\bar{\mathbf{A}}$, defined as

$$\bar{\mathbf{A}} = \mathbf{L}^{-\frac{1}{2}} \mathbf{A} \mathbf{L}^{-\frac{1}{2}} \quad (2)$$

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

$$\mathbf{D}_e = \{\mathbf{x}_1, \dots, \mathbf{x}_n, \dots, \mathbf{x}_N\}, \quad (3)$$

with the n th feature vector \mathbf{f}_n being represented as a K_m dimensional vector $\mathbf{x}_n = [e_{1n}, \dots, e_{kn}, \dots, e_{K_m n}]$, where e_{kn} is the n th element of the k th 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 [12] 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 Figure 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 Figure 1(i), the eigenvalues are not useful as all eigenvectors can assume high magnitude and higher eigenvectors do not necessarily mean higher relevance (see Figures 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 \mathbf{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 k th largest eigenvector \mathbf{e}_k being relevant as $R_{\mathbf{e}_k}$, with $0 \leq R_{\mathbf{e}_k} \leq 1$. 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:

$$p(e_{kn}|\theta_{e_{kn}}) = (1 - R_{\mathbf{e}_k})p(e_{kn}|\theta_{e_{kn}}^1) + R_{\mathbf{e}_k}p(e_{kn}|\theta_{e_{kn}}^2)$$

where $\theta_{e_{kn}}$ are the parameters describing the distribution, $p(e_{kn}|\theta_{e_{kn}}^1)$ is the pdf of e_{kn} when \mathbf{e}_k is irrelevant/redundant and $p(e_{kn}|\theta_{e_{kn}}^2)$ 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:

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

where $\mathcal{N}(\cdot|\mu, \sigma)$ denotes a Gaussian of mean μ and covariance σ^2 . We assume the second component of $P(\mathbf{e}_k|\theta_{\mathbf{e}_k})$ 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:

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

where w_k is the weight of the first Gaussian in $p(e_{kn}|\theta_{e_{kn}}^2)$. 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 ($p(e_{kn}|\theta_{e_{kn}}^2)$) still fits better to the data compared to a single Gaussian ($p(e_{kn}|\theta_{e_{kn}}^1)$); (2) its simple form means that only small number of parameters are needed to describe $p(e_{kn}|\theta_{e_{kn}}^2)$. This makes model learning possible even given sparse data.

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

$$\theta_{e_{kn}} = \{R_{\mathbf{e}_k}, \mu_{k1}, \mu_{k2}, \mu_{k3}, \sigma_{k1}, \sigma_{k2}, \sigma_{k3}, w_k\}. \quad (4)$$

The maximum likelihood (ML) estimate of $\theta_{e_{kn}}$ can be obtained using the following algorithm. First, the parameters of the first mixture component $\theta_{e_{kn}}^1$ are estimated as $\mu_{k1} = \frac{1}{N} \sum_{n=1}^N e_{kn}$ and $\sigma_{k1} = \frac{1}{N} \sum_{n=1}^N (e_{kn} - \mu_{k1})^2$. The rest 6 parameters are then estimated iteratively using Expectation Maximisation (EM) [4]. Specifically, in the E-step, the posterior probability that each mixture component is responsible for e_{kn} is estimated as:

$$\begin{aligned} h_{kn}^1 &= \frac{(1 - R_{\mathbf{e}_k}) \mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1})}{(1 - R_{\mathbf{e}_k}) \mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1}) + w_k R_{\mathbf{e}_k} \mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2}) + (1 - w_k) R_{\mathbf{e}_k} \mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3})} \\ h_{kn}^2 &= \frac{w_k R_{\mathbf{e}_k} \mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2})}{(1 - R_{\mathbf{e}_k}) \mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1}) + w_k R_{\mathbf{e}_k} \mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2}) + (1 - w_k) R_{\mathbf{e}_k} \mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3})} \\ h_{kn}^3 &= \frac{(1 - w_k) R_{\mathbf{e}_k} \mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3})}{(1 - R_{\mathbf{e}_k}) \mathcal{N}(e_{kn}|\mu_{k1}, \sigma_{k1}) + w_k R_{\mathbf{e}_k} \mathcal{N}(e_{kn}|\mu_{k2}, \sigma_{k2}) + (1 - w_k) R_{\mathbf{e}_k} \mathcal{N}(e_{kn}|\mu_{k3}, \sigma_{k3})} \end{aligned}$$

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

$$\begin{aligned} R_{\mathbf{e}_k}^{new} &= 1 - \frac{1}{N} \sum_{n=1}^N h_{kn}^1, & w_k^{new} &= \frac{1}{R_{\mathbf{e}_k}^{new} N} \sum_{n=1}^N h_{kn}^2, \\ \mu_{k2}^{new} &= \frac{\sum_{n=1}^N h_{kn}^2 e_{kn}}{\sum_{n=1}^N h_{kn}^2}, & \mu_{k3}^{new} &= \frac{\sum_{n=1}^N h_{kn}^3 e_{kn}}{\sum_{n=1}^N h_{kn}^3}, \\ \sigma_{k2}^{new} &= \frac{\sum_{n=1}^N h_{kn}^2 (e_{kn} - \mu_{k2}^{new})^2}{\sum_{n=1}^N h_{kn}^2}, & \sigma_{k3}^{new} &= \frac{\sum_{n=1}^N h_{kn}^3 (e_{kn} - \mu_{k3}^{new})^2}{\sum_{n=1}^N h_{kn}^3} \end{aligned}$$

Since the EM algorithm is essentially a local (greedy) searching method, it could be sensitive to parameter initialisation especially given noisy and sparse data [4]. 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 $p(e_{kn}|\theta_{e_{kn}}^2)$ over multiple random initialisations is chosen.

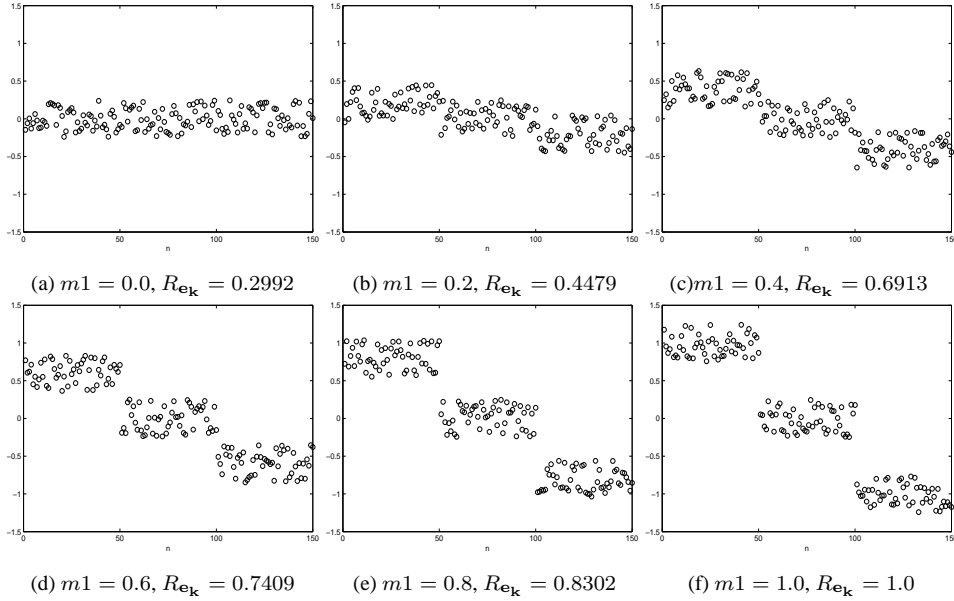


Figure 2: Synthetic eigenvectors and the estimated relevance measure R_{e_k} . The elements of each eigenvectors are composed of three uniformly distributed clusters. The mean of the three clusters are $m1$, 0, and $-m1$ respectively. Obviously, the bigger the value of $m1$, the more distinctive three clusters are formed in the distribution of the eigenvector and the more relevant the eigenvector is.

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_{e_k} . In other words, among the 8 free parameters of the eigenvector distribution (Eqn. (4)), R_{e_k} is the only parameter that we are after. This is why our algorithm works well 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 Figure 2 and more examples later in the paper. In particular, Figure 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_{e_k} estimated using our algorithm can still accurately reflect how relevant/informative an eigenvector is. Note that in the example shown in Figure 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, $\theta_{e_{kn}}$ (Eqn. (4)) are *not* estimated iteratively using a standard EM algorithm although part of $\theta_{e_{kn}}$, namely $\theta_{e_{kn}}^2$, 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 k th eigenvector \mathbf{e}_k among the K_m candidate eigenvectors if

$$R_{\mathbf{e}_k} < 0.5. \quad (5)$$

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

$$\mathbf{D}_r = \{\mathbf{y}_1, \dots, \mathbf{y}_n, \dots, \mathbf{y}_N\}, \quad (6)$$

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 Gaussian Mixture Model (GMM) for data clustering. Bayesian Information Criterion (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 Figure 3.

Let us first evaluate the effectiveness of our approach using a synthetic data set. We consider an 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 a HMM are denoted as $\{\mathbf{A}, \pi, \mathbf{B}\}$, where \mathbf{A} is the transition matrix representing the probabilities of transition between states, π 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

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.

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

the i th state). The parameters of the 4 HMMs are:

$$\begin{aligned}
 \mathbf{A}_1 &= \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1 & 0 & 0 \\ 1/6 & 1/2 & 1/3 \end{bmatrix}, \mathbf{A}_2 = \begin{bmatrix} 1/3 & 0 & 2/3 \\ 1/3 & 1/4 & 5/12 \\ 1/6 & 1/2 & 1/3 \end{bmatrix}, \mathbf{A}_3 = \begin{bmatrix} 0 & 1/6 & 5/6 \\ 1/6 & 1/2 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix} \\
 \mathbf{A}_4 &= \begin{bmatrix} 5/12 & 1/2 & 1/12 \\ 0 & 1/6 & 5/6 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}, \pi_1 = \pi_2 = \pi_3 = \pi_4 = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}, \\
 \mathbf{B}_1 = \mathbf{B}_2 = \mathbf{B}_3 = \mathbf{B}_4 &= \begin{cases} \mu_1 = 1, \sigma_1^2 = 0.5 \\ \mu_2 = 3, \sigma_2^2 = 0.5 \\ \mu_3 = 5, \sigma_3^2 = 0.5 \end{cases}, \tag{7}
 \end{aligned}$$

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 respectively², $P(\mathbf{S}_j | \mathbf{H}_i)$ 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 respectively³.

²Please refer to [10] for the details on learning the parameters of a HMM from data.

³Note that there are other ways to compute the affinity between two sequences modelled using DBNs [13, 14]. However, we found through our experiments that using different affinity measures makes little difference.

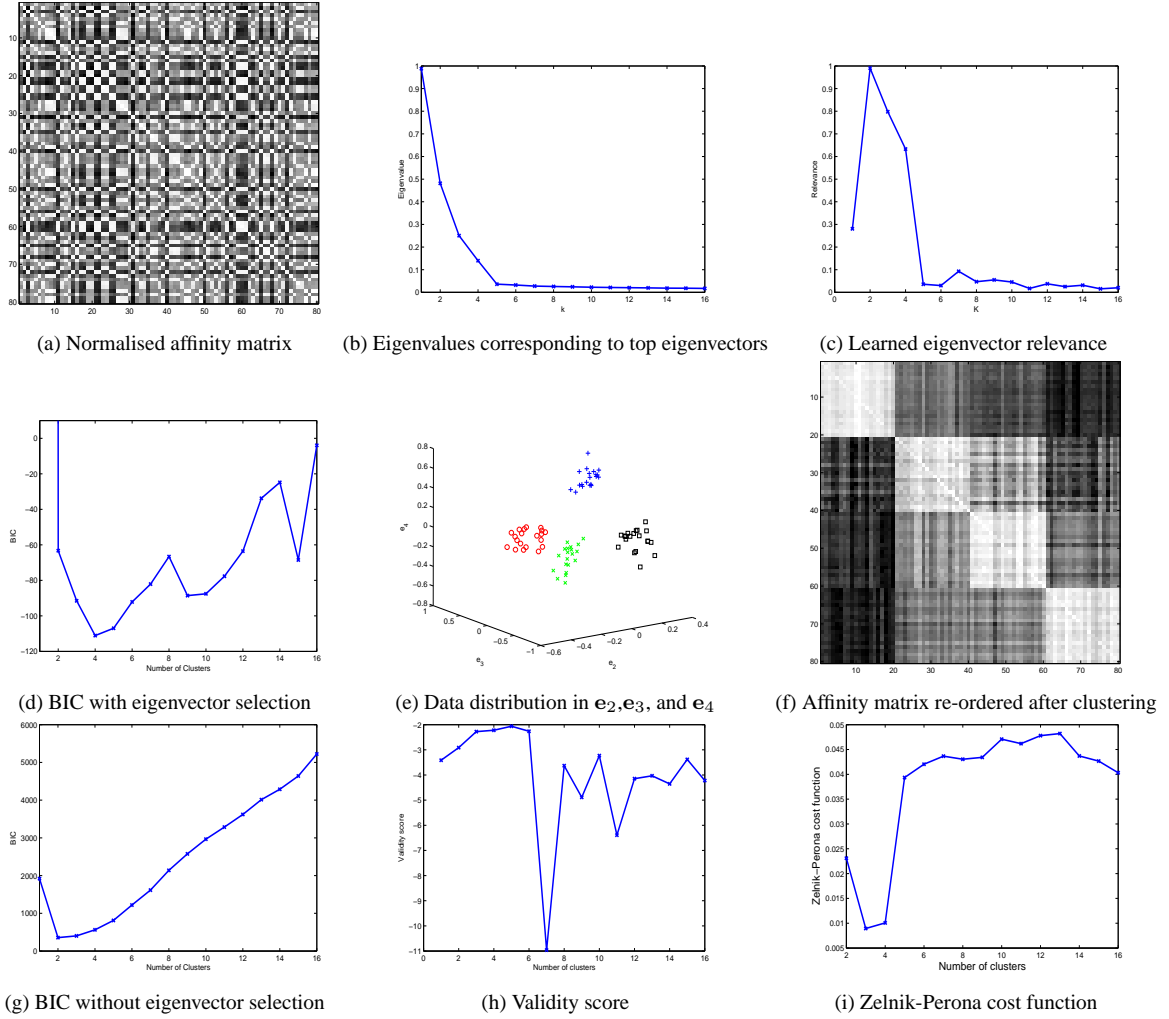


Figure 4: Clustering a synthetic data set using our spectral clustering algorithm. (a): The normalised affinity matrix constructed by modelling each sequence using a HMM. The eigenvalues of the $K_m = 16$ largest eigenvectors is shown in (b). (c): the learned relevance for the $K_m = 16$ largest eigenvectors. The second, third, and fourth largest eigenvectors were determined as being relevant using Eqn. (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. e_2, e_3 , and 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.

The proposed algorithm is used to determine the number of clusters and discover the natural grouping of the data. The results are shown in Figures 4 and 5. K_m was set to 16 in the experiment. It can be seen from Figure 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 Figures 5(e)&(f)). Figure 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 from Figure 4(c) that the proposed relevance measure R_{e_k} accurately reflects the relevance of each eigenvectors. By thresholding the relevance measure (Eqn. (5)), only e_2, e_3 , and e_4 are kept for

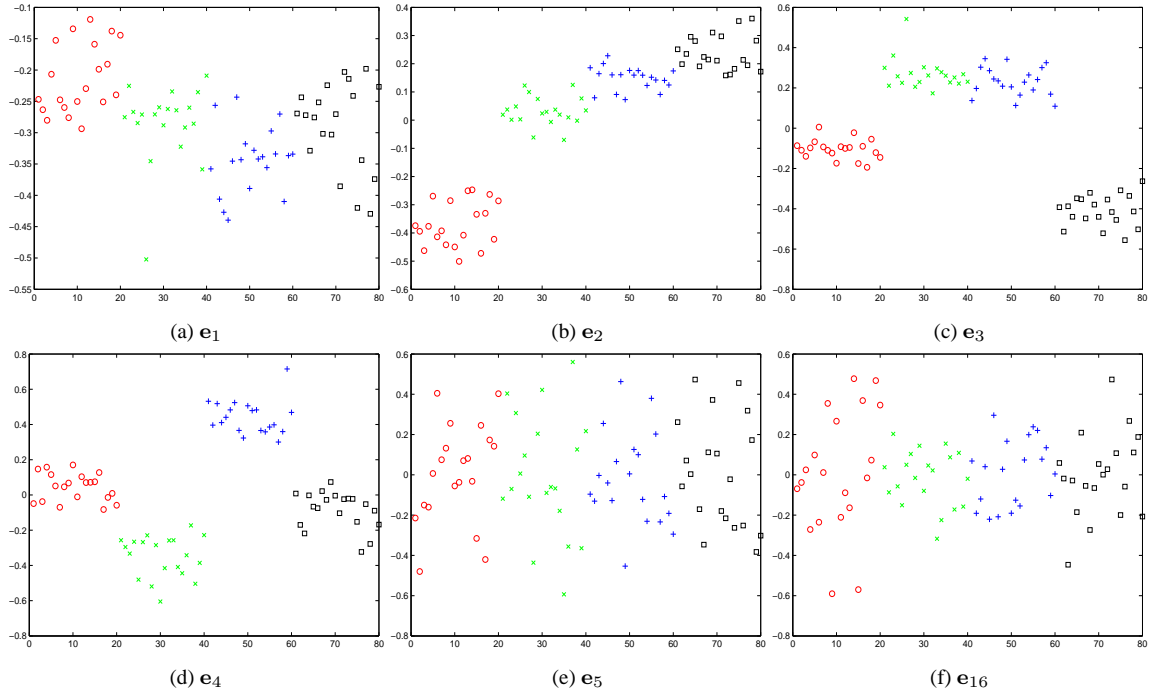


Figure 5: The distributions of the elements of some eigenvectors of the normalised affinity matrix shown in Figure 4(a). Elements corresponding to different classes are colour coded according to the classification result. For better illustration we have ordered the points so that points belonging to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

clustering. Figure 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 Figure 4(d)). The clustering result is illustrated using the re-ordered affinity matrix in Figure 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 [15] (maximum score correspond to the optimal number); and (c) Zelnik-Perona cost function [26] (minimum cost correspond to the optimal number). Figures 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. 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 by random noise with the identical uniform distribution as the previous experiment. Figure 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

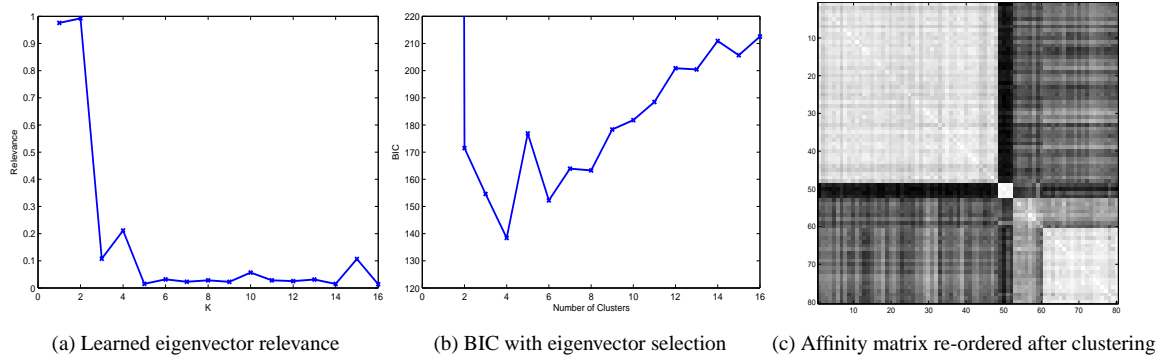


Figure 6: Clustering an unevenly distributed synthetic data set using our spectral clustering algorithm. The number of data points in each of the four clusters are 4, 8, 20, and 48 respectively. (a): the learned relevance for the $K_m = 16$ largest eigenvectors. The first and second largest eigenvectors were determined as being relevant using Eqn. (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.

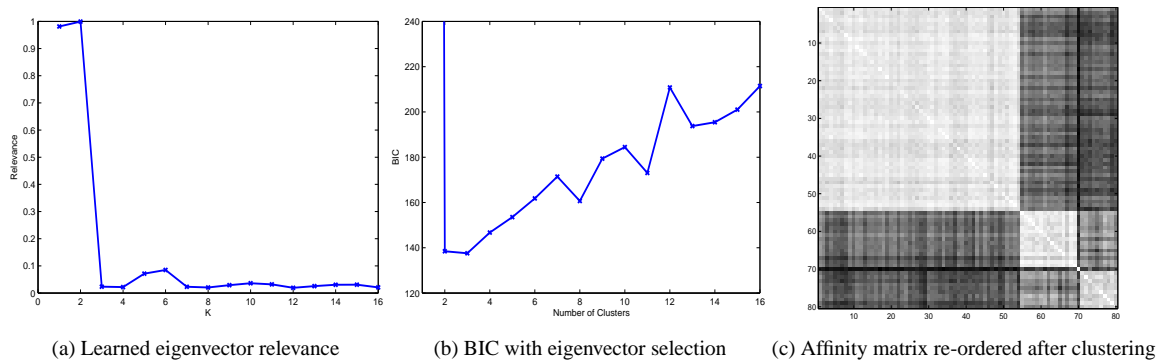


Figure 7: Clustering a synthetic data set with an extremely uneven distribution using our spectral clustering algorithm. The number of data points in each of the four clusters are 1, 10, 15, and 54 respectively. (a): the learned relevance for the $K_m = 16$ largest eigenvectors. The first and second largest eigenvectors were determined as being relevant using Eqn. (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.

experiment, the size of the largest cluster is 54 times bigger than that of the smallest one. Figure 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 clustering algorithm, two more synthetic data experiments were carried out. In the first experiment, 5 percent of the data points used in Figure 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). Figure 8 indicates that the 5 percent 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.

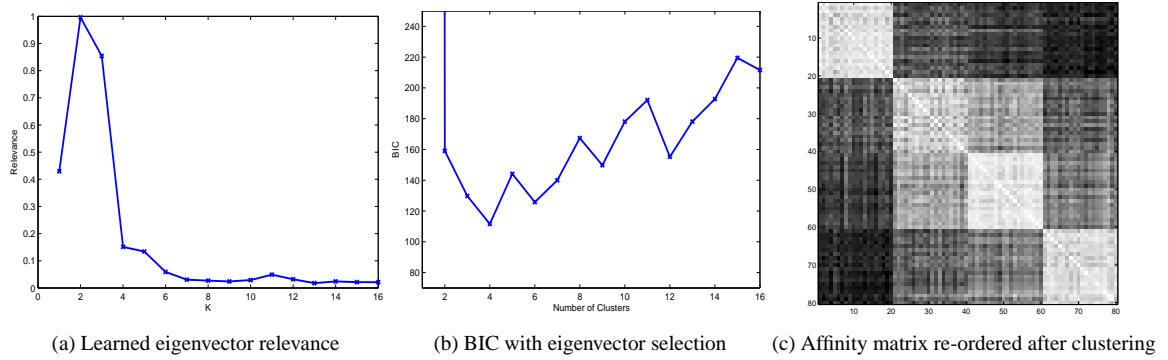


Figure 8: Clustering a synthetic data set with 5 percent outliers using our spectral clustering algorithm. (a): the learned relevance for the $K_m = 16$ largest eigenvectors. The second and third largest eigenvectors were determined as being relevant using Eqn. (5). (b) shows the BIC model selection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

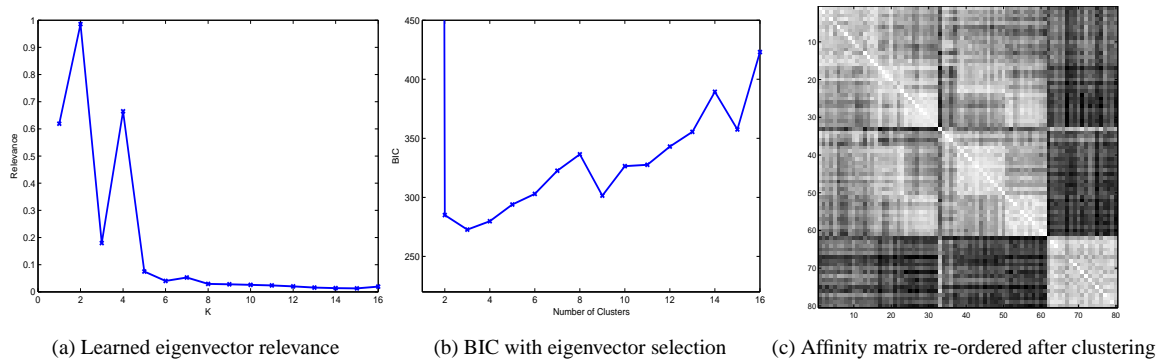


Figure 9: Clustering a synthetic data set with 20 percent outliers using our spectral clustering algorithm. (a): the learned relevance for the $K_m = 16$ largest eigenvectors. The first, second, and fourth largest eigenvectors were determined as being relevant using Eqn. (5). (b) shows the BIC model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

In the second experiment, 20 percent of the data points were substituted using noise. In this experiment, the number of clusters was determined as 3 (see Figure 8(b)). Figure 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 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 pair-wise affinity matrix \mathbf{A} is constructed for an image based on the Intervening Contours method introduced in [11]. First, for the i th 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

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

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.

Figure 10 illustrates in detail how our algorithm works for image segmentation. Given the original image in Figure 10(a), the maximum number of segments was set to 20. The associated eigenvalues are shown in Figure 10(k). Note that all the top 20 eigenvectors have an eigenvalue of magnitude close to 1. Figures 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. $\mathbf{e}_{13}, \mathbf{e}_{14}, \mathbf{e}_{17}, \mathbf{e}_{20}$). Figure 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 Figure 10(m)). The segmentation result in Figure 10(j) indicates that the image is segmented into meaningful

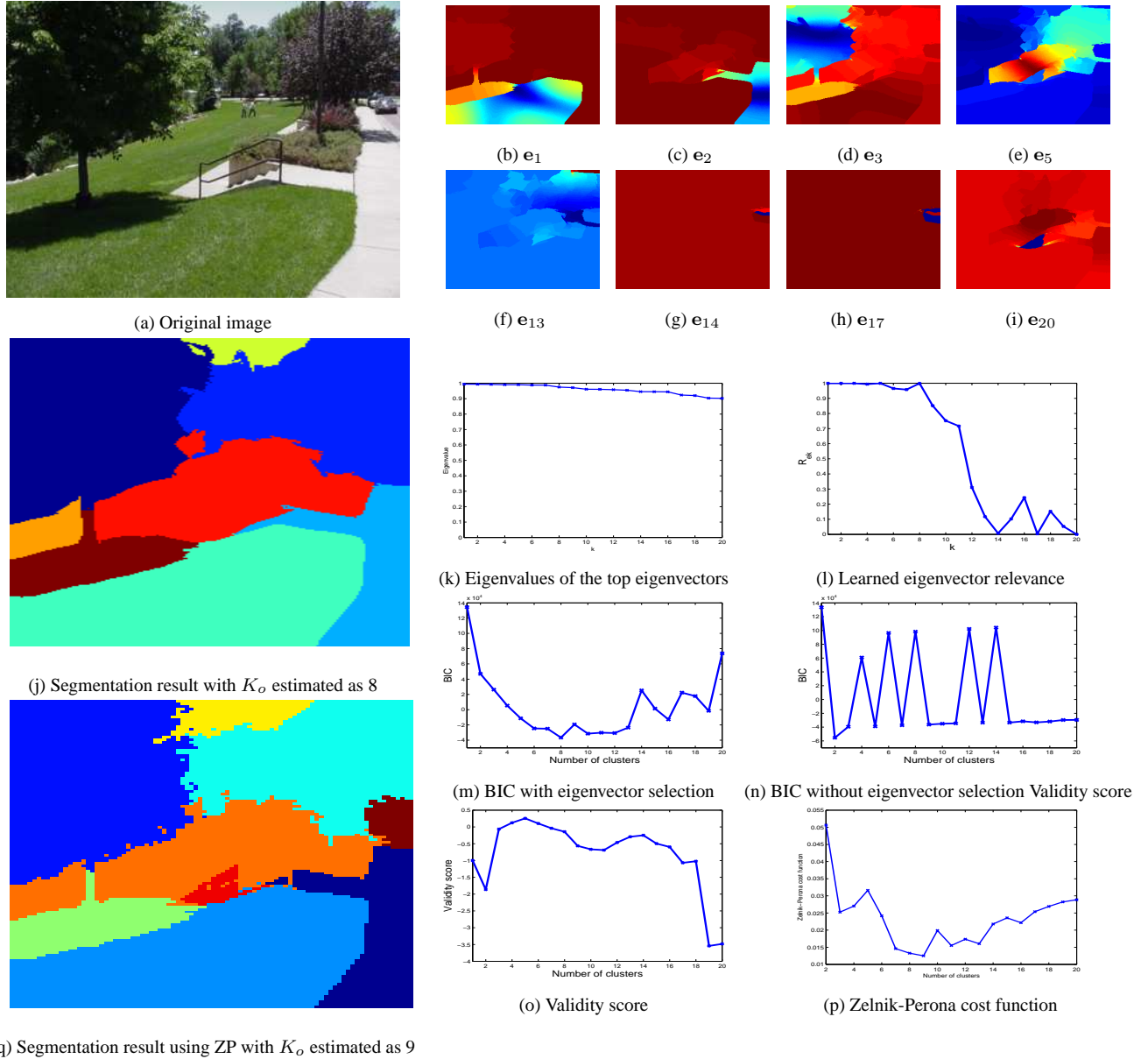


Figure 10: An example image shown in (a) is segmented as shown in (j). The corresponding eigenvalues of the top 20 eigenvectors are shown in (k). The learned relevance for the 20 largest eigenvectors is shown in (l). (b)-(e) and (f)-(i) show the top 4 most relevant and irrelevant eigenvectors among the 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.

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 [26]⁴ yielded comparable results to ours on this particular image (see Figures 10(p)&(q)).

Our algorithm has been tested on a variety of natural images. Figures 11&12 show some segmentation

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

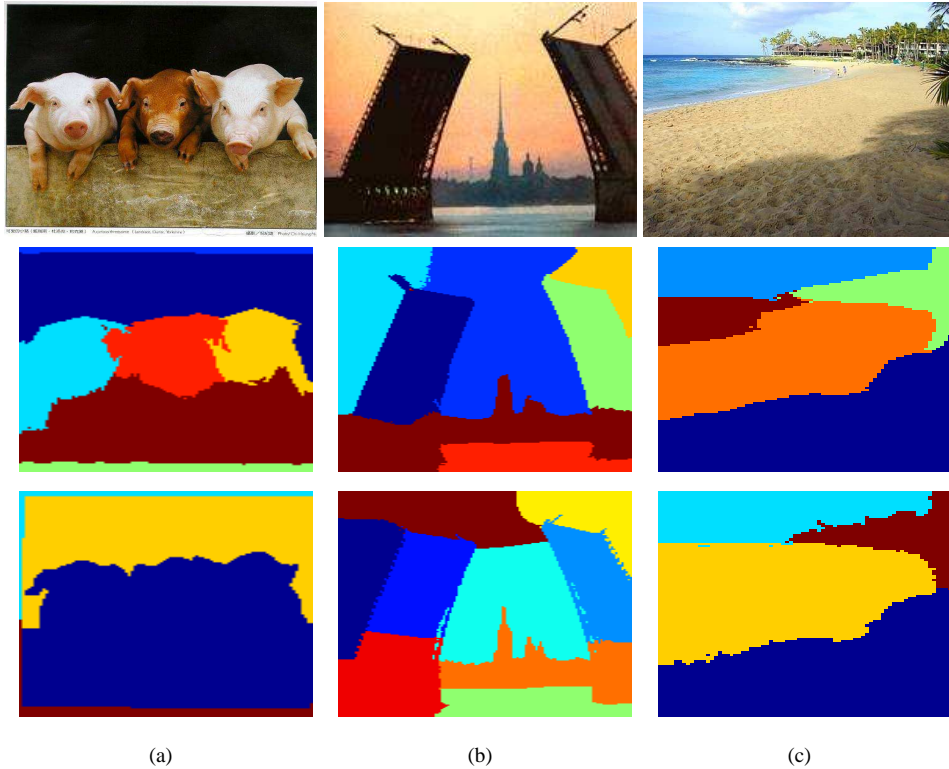


Figure 11: Further examples of image segmentation. The segmentation results using the proposed algorithm and Zelnik-Manor and Perona’s self-tuning spectral clustering algorithm are shown in the middle and bottom row respectively. From left to right, the optimal number of segments K_o were determined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

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 Figures 11&12. This supports our argument that selecting the relevant eigenvectors is critical for spectral clustering. The proposed algorithm was also compared with the Self-Tuning Spectral Clustering approach introduced in [26]. It can be seen from Figures 11&12 that in comparison, our approach led to more accurate estimation of the number of image segments and better segmentation. In particular, in Figures 11(a)&(c) and Figure 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 Figure 11(b) and Figures 12(a)&(b), although the two approaches obtained similar numbers of clusters, the segmentation results obtained using our algorithm are still superior.



Figure 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.

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 $\mathbf{V} = \{\mathbf{v}_1, \dots, \mathbf{v}_n, \dots, \mathbf{v}_N\}$ so that each segment contains approximately a single behaviour pattern. The n th video segment \mathbf{v}_n consisting of T_n image frames is represented as $\mathbf{v}_n = \{\mathbf{I}_{n1}, \dots, \mathbf{I}_{nt}, \dots, \mathbf{I}_{nT_n}\}$, where \mathbf{I}_{nt} is the t th image frame. 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 [22] can be adopted. Alternatively, the video can be simply sliced into overlapping segments with a fixed time duration [27].

A discrete event based approach is then adopted for behaviour representation [9, 23]. Firstly, an adaptive Gaussian mixture background model [20] is adopted to detect foreground pixels which are modelled using Pixel Change History (PCH) [24]. 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

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

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 Gaussian Mixture Model (GMM). The number of scene-event classes K_e captured in the videos is determined by automatic model order selection based on Bayesian Information Criterion (BIC) [17]. 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 n th video

segment \mathbf{v}_n is represented as a feature vector \mathbf{P}_n , given as

$$\mathbf{P}_n = [\mathbf{p}_{n1}, \dots, \mathbf{p}_{nt}, \dots, \mathbf{p}_{nT_n}], \quad (10)$$

where T_n is the length of the n th video segment and the t th element of \mathbf{P}_n is a K_e dimensional variable:

$$\mathbf{p}_{nt} = [p_{nt}^1, \dots, p_{nt}^k, \dots, p_{nt}^{K_e}]. \quad (11)$$

\mathbf{p}_{nt} corresponds to the t th image frame of \mathbf{v}_n where p_{nt}^k is the posterior probability that an event of the k th event class has occurred in the frame given the learned GMM. If an event of the k th class is detected in the t th image frame of \mathbf{v}_n , we have $0 < p_{nt}^k \leq 1$; otherwise, we have $p_{nt}^k = 0$. 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 $\mathbf{D} = \{\mathbf{P}_1, \dots, \mathbf{P}_n, \dots, \mathbf{P}_N\}$ consisting of N behaviour patterns, where \mathbf{P}_n is the n th 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 [4, 8]. Similar to the synthetic data case (see Section 2), the affinity between \mathbf{P}_i and \mathbf{P}_j is then computed as:

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

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.

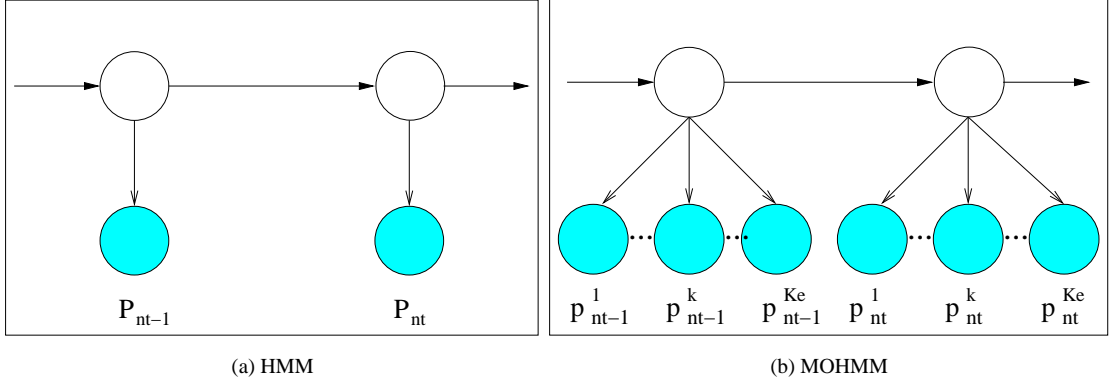


Figure 13: Modelling a behaviour pattern $\mathbf{P}_n = \{\mathbf{p}_{n1}, \dots, \mathbf{p}_{nt}, \dots, \mathbf{p}_{nT_n}\}$ where $\mathbf{p}_{nt} = \{p_{nt}^1, \dots, p_{nt}^k, \dots, p_{nt}^{K_e}\}$ using a HMM and a MOHMM. Observation nodes are shown as shaded circles and hidden nodes as clear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

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) [9] 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_{nt}^k) is independent of each other. Consequently, the number of parameters for describing a MOHMM is much lower than that for a HMM ($2K_e N_s + N_s^2 - 1$ for a MOHMM and $(K_e^2 + 3K_e)N_s/2 + N_s^2 - 1$ for a 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 k th behaviour class using a MOHMM \mathbf{B}_k . The parameters of \mathbf{B}_k , $\theta_{\mathbf{B}_k}$ are estimated using all the patterns in the training set that belong to the k th class. A behaviour model \mathbf{M} is then formulated as a 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

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

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

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

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

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

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

4.4 Experiments

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

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

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 Figure 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 Figure 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 hours of video totalling 432000 frames captured at 20Hz 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.



Figure 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)&(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.

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 con-

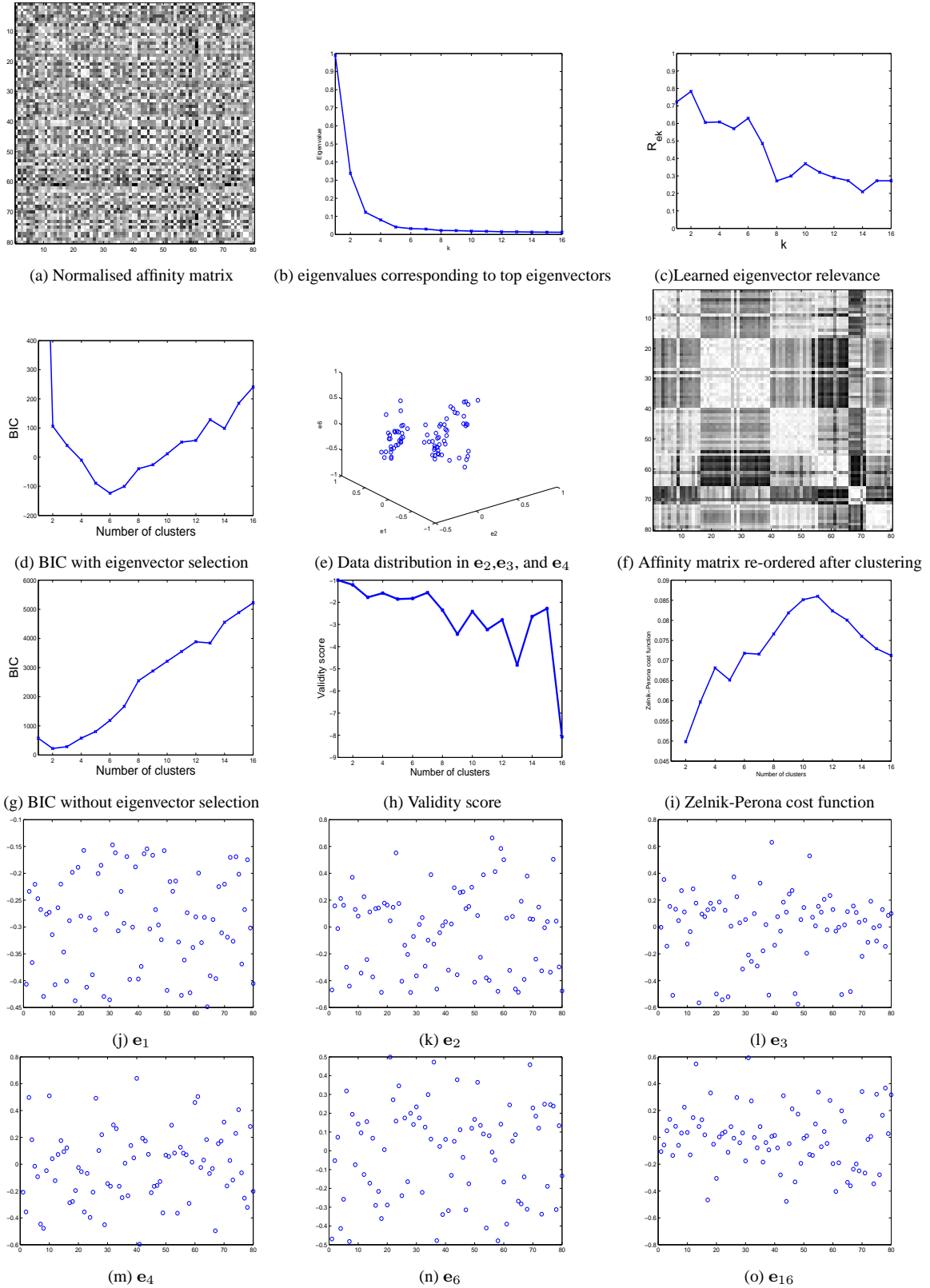


Figure 15: An example of behaviour pattern clustering. (c) shows that the top 6 largest eigenvectors were determined as relevant features for clustering. (d) and (g) show the number of behaviour classes was determined as 6 and 2 using BIC with and without relevant eigenvector selection respectively. (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.

stituents 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. Figure 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 Figures 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.

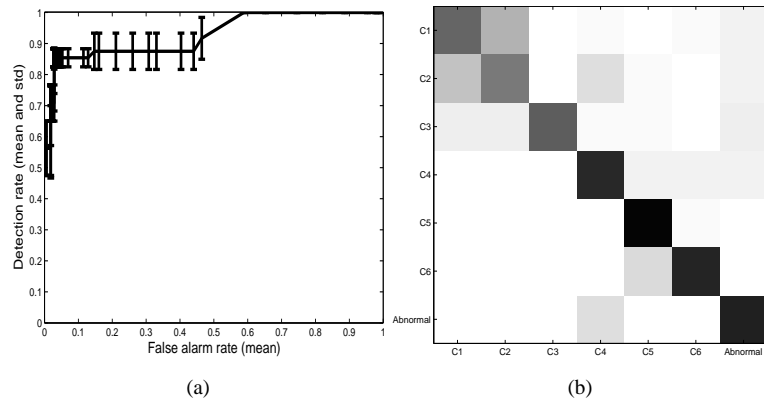


Figure 16: The performance of abnormality detection and behaviour recognition for the corridor scene. (a): The mean and ± 1 standard deviation of the ROC curves for abnormality detection obtained over 20 trials. (b): Confusion matrix for behaviour recognition. Each row represents the probabilities of 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].

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 a ROC curve. Figure 16(a) shows that high detection rate and low false alarm rate have been achieved. Th_A (see Eqn.(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 is illustrated as a confusion matrix shown in Figure 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 Figure 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 a 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 choose 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 100000; 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 [16, 7]. In particular, in a high-dimensional eigenspace spanned by the top K_m eigenvectors, the Gaussian Mixture Model (GMM) used for modelling data distribution in the eigenspace would suffer 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 [15] and [26] 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 [7, 1]. The ongoing work includes investigating the effect of employing different model selection criteria on the performance of our algorithm.

References

- [1] C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):719–725, 2000.
- [2] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245–271, 1997.
- [3] F. Chung. *Number 92 in CBMS Regional Conference Series in Mathematics*. American Mathematical society, 1997.
- [4] A. Dempster, N. Laird, and D. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.
- [5] J. Dy, C. Brodley, A. Kak, L. Broderick, and A. Aisen. Unsupervised feature selection applied to content-based retrieval of lung images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:373–378, 2003.
- [6] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journals*, 23:298–305, 1973.
- [7] M. Figueiredo and A.K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):381–396, 2002.
- [8] Z. Ghahramani. Learning dynamic bayesian networks. In *Adaptive Processing of Sequences and Data Structures. Lecture Notes in AI*, pages 168–197, 1998.

- [9] S. Gong and T. Xiang. Recognition of group activities using dynamic probabilistic networks. In *IEEE International Conference on Computer Vision*, 2003.
- [10] L.R.Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [11] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *International Journal of Computer Vision*, pages 7–27, 2001.
- [12] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, 2001.
- [13] A. Panuccio, M. Bicego, and 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*, pages 734–742, London, UK, 2002. Springer-Verlag.
- [14] 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.
- [15] F. Porikli and T. Haga. Event detection by eigenvector decomposition using object and frame features. In *IEEE conference on Computer Vision and Pattern Recognition workshop*, pages 114–121, 2004.
- [16] S. Roberts, D. Husmeier, I. Rezek, and W. Penny. Bayesian approaches to Gaussian mixture modelling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1133–1142, 1998.
- [17] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.
- [18] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- [19] P. Smyth. Clustering sequence with hidden markov models. In *Advances in Neural Information Processing Systems*, pages 648–654, 1997.
- [20] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–758, August 2000.
- [21] Y. Weiss. Segmentation using eigenvectors: a unifying view. In *IEEE International Conference on Computer Vision*, pages 975–982, 1999.
- [22] T. Xiang and S. Gong. Activity based video content trajectory representation and segmentation. In *British Machine Vision Conference*, 2004.
- [23] T. Xiang and S. Gong. Beyond tracking: Modelling activity and understanding behaviour. *International Journal of Computer Vision*, 67:21–51, 2006.
- [24] T. Xiang, S. Gong, and D. Parkinson. Autonomous visual events detection and classification without explicit object-centred segmentation and tracking. In *British Machine Vision Conference*, pages 233–242, 2002.

- [25] S. Yu and J. Shi. Multiclass spectral clustering. In *IEEE International Conference on Computer Vision*, pages 313–319, 2003.
- [26] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems*, 2004.
- [27] H. Zhong, J. Shi, and M. Visontai. Detecting unusual activity in video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 819–826, 2004.