

Clustering of Sparse Binary Data using a Minimum Description Length Approach

Mark D. Plumbley

Department of Electronic Engineering

Queen Mary, University of London

Mile End Road, London E1 4NS, UK

Email: mark.plumbley@elec.qmul.ac.uk

25 September 2002

Abstract

We consider the problem of analyzing binary data, arising for example from market basket data or term presence in text documents, using a minimum-description length (MDL) approach. An encoding is constructed based on a product of binary matrices, and by imposing certain restrictions on our allowed encoding, we derive a clustering system with a cluster merge distance based on the size of the minimal covers of pre-merge and post-merge clusters. In the case of singleton clusters, this corresponds to the Hamming distance between documents. We illustrate this clustering system with an agglomerative hierarchical clustering of newsgroup documents, producing representative (basis) keywords and discriminative (conjugate) keywords for each cluster. Finally we consider extensions of this approach that may be possible by relaxing the restrictions which give our strict cover-based clustering.

Index Terms

Clustering, Text mining, Minimum Description Length, Boolean attributes, Cluster Cover, Bayesian modeling, Concept vectors.

I. INTRODUCTION

One method used to find structure in data sets is that of *clustering*, where data points are grouped so that points within a cluster are ‘close’ to each other in some sense. In this paper we consider clustering of data where the data points have Boolean attributes, a special case of *categorical attributes*, and can therefore be represented by binary vectors.

Typical applications for binary data clustering include *market basket* clustering, and *text document* clustering. For the former, each vector might be the purchases made at a particular visit to a store, where each attribute indicates whether or not any of a particular product was purchased [1]. For the latter, each vector might represent the set of words present in a document, with a 1 to indicate any of a given word, and 0 for none of that word [2]. In either case the count of the number of items is ignored.

One feature of this type of application is that the data is typically very *sparse*. A customer will only buy a small fraction of the possible product lines on any visit to a store, and a given document will only contain a small fraction of the possible number of words used in the dataset. For concreteness, we will use text document clustering as our theme in this paper.

In text document clustering, suppose that each document k is represented by a vector $\mathbf{x}_k = (x_{1k}, \dots, x_{mk})$ representing the set of terms present in that document:

$$x_{ik} = \begin{cases} 1 & \text{if term } i \text{ is present in document } k \\ 0 & \text{otherwise .} \end{cases} \quad (1)$$

The matrix $\mathbf{X} = [\mathbf{x}_{ik}]$ is then an $m \times p$ “term \times document” matrix where m and p are typically very large, but \mathbf{X} itself is sparse. For example, we may have $m \approx p \approx 5000$, and less than 1% of non-zero entries in \mathbf{X} .

Dealing with data in such a high-dimensional space can be difficult. One popular approach to deal with this high-dimensionality problem is Latent Semantic Indexing (LSI) [3] which reduces the term dimensionality p by applying principal component analysis (PCA) to project each document vector \mathbf{x}_k into a lower-dimensional space of e.g. 20 dimensions.

In contrast, in this paper we present a method which allows us to take advantage of the sparsity of the data. We will construct a method based on a minimum description length (MDL) approach [4], which will allow us to cluster the data based on a “centroid-like” method. We simplify the coding structure by making certain restrictions on our code length scheme which result in a simple and efficient cluster merge distance. We then build this into a straightforward hierarchical agglomerative clustering method to find the clusters for any n clusters, and demonstrate the approach on clustering of newsgroup data.

II. CLUSTERING AND SIMILARITY MEASURES FOR BINARY DATA

Many clustering algorithms are based on the concept of *similarity* or *distance* between data vectors. The aim is then to cluster the data vectors so that vectors within a cluster have high similarity (low distance) to each other, while having a lower similarity (higher distance) with documents in a different cluster.

Consider a pair of m -dimensional binary vectors \mathbf{x} and \mathbf{y} . An obvious distance measure for these vectors is the *Hamming distance*

$$H(\mathbf{x}, \mathbf{y}) = \sum_i x_i \oplus y_i = |\mathbf{x} \vee \mathbf{y}| - |\mathbf{x} \wedge \mathbf{y}| \quad (2)$$

where \oplus is the exclusive or operator, \wedge, \vee are the AND and OR operators respectively, and $|\mathbf{a}|$ measures the number of 1’s in \mathbf{a} .

It is often considered helpful to use a measure of *similarity* between documents, in the range 0 to 1, rather than use a distance measure. For example, the Jaccard similarity measure [5], [1] is

$$J(\mathbf{x}, \mathbf{y}) = \frac{|\mathbf{x} \wedge \mathbf{y}|}{|\mathbf{x} \vee \mathbf{y}|} = 1 - \frac{H(\mathbf{x}, \mathbf{y})}{|\mathbf{x} \vee \mathbf{y}|} \quad (3)$$

and the Rand index [6] is

$$\begin{aligned} R(\mathbf{x}, \mathbf{y}) &= 1 - (|\mathbf{x} \vee \mathbf{y}| - |\mathbf{x} \wedge \mathbf{y}|)/m \\ &= 1 - H(\mathbf{x}, \mathbf{y})/m \end{aligned} \quad (4)$$

One significant disadvantage of the Hamming distance, and the Jaccard and Rand indexes, is that they only measure the distance between individual points in a cluster, not between clusters themselves. This means that centroid-based hierarchical clustering schemes cannot be used directly

[1]. Instead, it is common to use either a minimum spanning tree (MST), merging the pair of clusters containing the most similar points, or to use a group average, merging cluster with highest average similarity between documents. However, MST is known to be sensitive to outliers, while the use of group average tends to split large clusters, since the average similarity between subclusters is small. Alternatively, the ROCK algorithm of Guha et al [1] addresses this problem by introducing the concept of *links* between data points.

In this paper, we will instead derive a measure of cluster distance for binary data using the concept of a *cover*, which has Hamming distance as a special case. This cover, together with a count of points inside the cluster, can be used in place of a centroid for the cluster. Therefore a centroid-based hierarchical clustering algorithm can be used, and information about the individual data points can be discarded once merged into a cluster.

III. CODING FRAMEWORK

We will explicitly construct an encoding scheme that could be used to transmit the information in the data \mathbf{X} . We will then search for an encoding that gives us the shortest code length. This approach is known as the *minimum description length* (MDL) approach [4], and is conceptually similar to minimum message length (MML) [7] and complexity minimization [8].

In fact, the MDL/MML approach is a Bayesian method: code lengths and code structures in the coding model are equivalent to negative log probabilities and probability structure assumptions in the Bayesian approach [7]. Nevertheless, we find the coding model more appealing in that we do not need to *assume* that the data has any given probability structure or generative model. We will know, however, that the encoding scheme will have a good fit to the data if our coding model does correspond to the process whereby the data is generated, i.e. if our corresponding *generative model* is correct.

Let us therefore start from the encoding that we wish to perform. To transmit \mathbf{X} to a remote observer without any complex encoding, we would need to send the $m \times p$ binary values of \mathbf{X} directly. This would take mp bits to transmit if we use 1 bit for each 1 or 0. Alternatively, if we are allowed to use different code lengths for 1 and 0, the minimum code length achievable would be

$$L^1(\mathbf{X}) = N_1^{\mathbf{X}} l_1^x + N_0^{\mathbf{X}} l_0^x = -mp(q \log_2 q + (1 - q) \log_2(1 - q)) \quad (6)$$

where $N_1^{\mathbf{X}}$ (respectively $N_0^{\mathbf{X}}$) is the number of 1's (0's) in \mathbf{X} , $l_1 = -\log_2 q$ and $l_0 = -\log_2(1 - q)$, with $q = N_1^{\mathbf{X}}/mp$. Let us call this the 'default' encoding.

We note that $L^1(\mathbf{X})$ is the negative total log likelihood of the data values, and is mp times the entropy of the elements of \mathbf{X} , under an assumption that the elements x_{ik} were independent and identically distributed with probability $\Pr(x_{ik} = 1) = q$. Of course we hope that this i.i.d. assumption is incorrect in any data that we expect to analyze, since this would imply that the data had no structure with which to construct an encoding model, but nevertheless the coding model is still a valid one.

Now suppose that we realize that there is some dependency structure that we can take advantage of. Specifically, we consider that \mathbf{X} may have been generated as a binary product of binary matrices, i.e. that $\mathbf{X} \approx \mathbf{A}^* \circ \mathbf{S}^*$. We wish to find an encoding based on an approximate factorization $\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{A} \circ \mathbf{S}$ where \mathbf{A} is an $m \times n$ binary matrix and \mathbf{S} is an $n \times p$ binary matrix. The product $\mathbf{A} \circ \mathbf{S}$ is an OR-of-ANDs operation, i.e.

$$\hat{x}_{ik} = \bigvee_{j=1}^n a_{ij} s_{jk} = \begin{cases} 1 & \text{if } \sum_j a_{ij} s_{jk} \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

Instead of encoding the elements of \mathbf{X} alone, we then encode the three matrices \mathbf{A} , \mathbf{S} and $(\mathbf{X} | \hat{\mathbf{X}})$. Thus we assign different code lengths $q_{x\hat{x}}$ to the values of the elements x_{ik} depending on the values of the corresponding estimates \hat{x}_{ik} . (For a complete encoding of this system we will also need to send the numbers m, n and p , but we shall ignore these for the purposes of this paper, under the assumption that the coding required for these will be small in comparison to the matrices.) If this were a *topic model* [9] each j might be called a *topic*, where $s_{jk} = 1$ if document k is ‘significantly’ concerned with topic j , and $a_{ij} = 1$ if term i is ‘likely’ to be found in topic j .

This is equivalent to a Naïve Bayes system, specifically a *binary independence model* (BIM) [10]. In the Bayesian framework, this corresponds to the assumption that the data values \mathbf{X} are independent given a topic choice \mathbf{S} , and it is known that this assumption almost never holds for natural data [10]. However, as mentioned earlier, in an MDL framework we are making an *encoding choice* rather than relying on our initial *assumption*, so we avoid having to assume something which is not known to hold.

Now, for any given (\mathbf{A}, \mathbf{S}) matrix pair, we need to choose the code lengths $l_{\alpha\beta}^x$ to use to encode $(\mathbf{X} | \hat{\mathbf{X}})$. For $\alpha, \beta \in \{0, 1\}$, let

$$N_{\alpha\beta} = \sum_{ik} \delta(x_{ik}, \alpha) \delta(\hat{x}_{ik}, \beta) \quad (8)$$

be the number of elements of \mathbf{X} which have value α where the corresponding value for $\hat{\mathbf{X}}$ is β , and let

$$N_{\cdot\beta} = \sum_{ik} \delta(\hat{x}_{ik}, \beta) \quad (9)$$

be the number of elements of $\hat{\mathbf{X}}$ which have value β . Then the optimum code length to use for any element $x_{ik} = \alpha$ given knowledge of $\hat{x}_{ik} = \beta$ is $l_{\alpha\beta} = -\log_2 q_{\alpha\beta}$ where $q_{\alpha\beta} = N_{\alpha\beta}/N_{\cdot\beta}$. This will give us a code length for \mathbf{X} given $\hat{\mathbf{X}}$ of

$$L(\mathbf{X}|\hat{\mathbf{X}}) = \sum_{\alpha\beta} N_{\alpha\beta} l_{\alpha\beta} \quad (10)$$

with an overall code length of

$$L = L(\mathbf{A}, \mathbf{S}, \mathbf{X}) = L(\mathbf{A}) + L(\mathbf{S}) + L(\mathbf{X}|\hat{\mathbf{X}}) \quad (11)$$

where some suitable encoding is used for $L(\mathbf{A})$ and $L(\mathbf{S})$.

We therefore wish to search for a binary matrix pair (\mathbf{A}, \mathbf{S}) which will minimize the overall encoding length $L(\mathbf{A}, \mathbf{S}, \mathbf{X})$. In a Bayesian framework, $L(\mathbf{A})$ and $L(\mathbf{S})$ are negative log priors for \mathbf{A} and \mathbf{S} , while $L(\mathbf{X}|\hat{\mathbf{X}})$ is a negative log likelihood of \mathbf{X} given \mathbf{A} and \mathbf{S} (and hence given $\hat{\mathbf{X}}$).

IV. SIMPLIFIED MDL FOR SPARSE DATA

This full encoding framework is very powerful, as is the equivalent Bayesian approach, but a solution for the general case will involve a search over the product space of the matrix pairs (\mathbf{A}, \mathbf{S}) to find a minimum of L . Often this is achieved using the EM algorithm [11].

However, in the case of sparse binary data we can make some restrictions to our coding model which will result in a particularly simple form. In particular, we will derive a clustering model where we no longer need to measure the true code length, but find a simple cluster merge distance which can be efficiently computed for sparse data.

Restriction 1: Given some n , the coding lengths $L(\mathbf{A})$ and $L(\mathbf{S})$ are fixed.

For example, this would result from using one bit to encode each element of \mathbf{A} and \mathbf{S} irrespective of the number of 1's and 0's in each matrix. Under this restriction, to minimize $L(\mathbf{A}, \mathbf{S}, \mathbf{X})$ at any n , we simply need to minimize $L(\mathbf{X}|\hat{\mathbf{X}})$. In the corresponding Bayesian scheme, this is equivalent to assuming that the prior probabilities of all the elements of \mathbf{A} and \mathbf{S} are $1/2$.

Restriction 2: Each column \mathbf{s}_k of \mathbf{S} has exactly one non-zero element.

This introduces an element of asymmetry into the coding system. We might write

$$s_{jk} = \begin{cases} 1 & \text{if } j = c_k \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

where $\mathbf{c} = (c_1, \dots, c_p)$ is a vector giving the ‘cluster’ to which each vector \mathbf{x}_k has been assigned. In text analysis, each document k is therefore encoded using one column \mathbf{a}_j of \mathbf{A} only.

This restriction transforms our task into one of cluster analysis, with cluster ‘centroids’ \mathbf{a}_j and cluster assignment indicators \mathbf{s}_k . We therefore wish to adjust these ‘centroids’ and assignments to minimize $L(\mathbf{X}|\hat{\mathbf{X}})$. One further restriction will give us a simpler way to view this.

Restriction 3: For the encoding of \mathbf{X} given $\hat{\mathbf{X}}$, fix $l_{00} = 0$.

In other words, if we know that $\hat{x}_{ik} = 0$, then we pay nothing (i.e. have zero code length) if $x_{ik} = 0$. Alternatively, we could have generated these by fixing $q_{00} = 1$ hence requiring $q_{10} = 0$. This is actually quite a strong restriction: it implies that $l_{10} = \infty$, and consequently any matrices \mathbf{A} and \mathbf{S} which generate $\hat{\mathbf{X}}$ such that there is any element $\hat{x}_{ik} = 0$ where $x_{ik} = 1$ will result in an infinite-length code. The result of this restriction is that for any non-infinite length encoding, $\hat{\mathbf{X}}$ must *cover* \mathbf{X} , that is, $x_{ik} = 1 \Rightarrow \hat{x}_{ik} = 1$.

Consider one column \mathbf{x}_k of \mathbf{X} . Under restriction 2, \mathbf{s}_k has exactly one non-zero entry $s_{j_k k} = 1$. Thus the k th column $\hat{\mathbf{x}}_k$ of $\hat{\mathbf{X}}$ is given by $\hat{\mathbf{x}}_k = \mathbf{a}_{j_k}$. Consequently, given \mathbf{X} and \mathbf{S} , a_{ij} must be 1 if there exists a k for which $x_{ik} = 1$ and $s_{jk} = 1$.

The code length is now computed from two terms rather than four. For a non-infinite code we have $N_{10} = 0$ and hence $N_{11} = N_{\cdot 1} = \sum_{ik} \delta(x_{ik}, 1)$ which is independent of \mathbf{A} and \mathbf{S} . Therefore we have $N_{01} = N_{\cdot 1} - N_{11}$ where $N_{\cdot 1} = \sum_{ik} \delta(\hat{x}_{ik}, 1)$ so $q_{11} = N_{11}/N_{\cdot 1}$ and $q_{01} = 1 - q_{11}$.

Since q can be regarded as a probability estimator for \mathbf{X} , we know that for the best possible fit we wish N_{01} to be as small as possible, i.e. for $\hat{\mathbf{X}}$ to be a *minimal cover* for \mathbf{X} . It is also straightforward to verify that the code length L increases monotonically with N_{01} since N_{11} is fixed. It is therefore sufficient to search for a matrix pair (\mathbf{A}, \mathbf{S}) which minimises N_{01} , such that $N_{10} = 0$.

We can see that our required \mathbf{A} is now determined exactly from \mathbf{X} and \mathbf{S} . Specifically we have

$$a_{ij} = \begin{cases} 1 & \text{if } \exists k \text{ s.t. } x_{ik} = 1 \text{ and } s_{jk} = 1 \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

or more compactly $\mathbf{A} = \mathbf{X} \circ \mathbf{S}^T$. Any other choice of \mathbf{A} would give us a larger value for N_{10} unnecessarily. For the text clustering application, we can interpret this as follows: the set of terms i included in cluster j must be the union of the sets of terms in all documents k assigned to cluster j .

It remains to find the \mathbf{S} which minimizes the code length. In this paper we will use an agglomerative hierarchical clustering method, which has the advantage in this exposition in that it is conceptually simple, and requires us to compute cluster merge costs only. However, it does have complexity at best p^2 with the number of documents p so may not be applicable to very large document sets (our example later is on a set of 5000 documents each of almost 4695 terms).

V. HIERARCHICAL CLUSTERING METHOD

We start with $n = p$ clusters, each containing a single document, and successively merge pairs of ‘close’ clusters until we reach the desired number of clusters n^* . This will be a ‘centroid’-based method, where each column \mathbf{a}_j of \mathbf{A} , together with a count of the number of documents in the cluster, will be sufficient to fully represent the cluster without requiring any information about the individual documents within the cluster. The BIRCH algorithm by Zhang et al [12] introduced a similar concept of *clustering features* (CFs) to summarize non-binary data in a cluster.

Note that, although we use the term ‘centroid’-based here, each \mathbf{a}_j is a *cover* for the documents within a give cluster, not a centroid (or even a medoid [13]) the use of the term ‘centroid’, which suggests \mathbf{a}_j is somehow in the ‘middle’ of the clustered documents, may be misleading.

In outline, a hierarchical agglomerative clustering method proceeds as follows [14]

1. Begin with $n = p$ clusters, each containing one element;
2. Merge the pair of clusters with the minimum *merge distance*, giving $n - 1$ clusters;
3. Repeat from 2 until the desired number of clusters n^* is reached.

It is therefore sufficient to define a *merge distance*, or *merge cost*, between clusters in order to use this approach.

In our system, we wish to minimize the code length of the matrix triple $(\mathbf{A}, \mathbf{S}, (\mathbf{X}|\hat{\mathbf{X}}))$. Under the set of restrictions that we consider in this paper, we have seen in the previous section that it is sufficient to minimize the number of 1’s in $\hat{\mathbf{X}}$ which correspond to 0’s in \mathbf{X} , while ensuring that $\hat{\mathbf{X}}$ covers \mathbf{X} .

Let us consider merging two clusters \mathbf{a}_1 and \mathbf{a}_2 containing n_1 and n_2 elements each, giving us a new cluster \mathbf{a}_3 with $n_3 = n_1 + n_2$ elements. We assume that \mathbf{a}_1 and \mathbf{a}_2 are minimal covers for the elements they contain. For \mathbf{a}_3 to be a minimal cover for the merged cluster, we must have $a_{3j} = a_{1j} \vee a_{2j}$, i.e. the set of elements of \mathbf{a}_3 with value 1 is the union of the set of value-1 elements of \mathbf{a}_1 and \mathbf{a}_2 . We will also have a merged \mathbf{S} matrix, whereby rows \mathbf{s}_1 and \mathbf{s}_2 are replaced by a new merged row $\mathbf{s}_3 = \mathbf{s}_1 \vee \mathbf{s}_2 = \mathbf{s}_1 + \mathbf{s}_2$, where ‘or’ and addition are equivalent since the columns of \mathbf{S} each contain only one non-zero entry.

Let us consider the effect of this cluster merge on $\hat{\mathbf{X}}$. Before the merge, any columns (documents) in clusters 1 and 2 were represented by \mathbf{a}_1 and \mathbf{a}_2 respectively in those columns of $\hat{\mathbf{X}}$. After the merge, these columns are all represented by $\mathbf{a}_3 = \mathbf{a}_1 \vee \mathbf{a}_2$ in the corresponding columns of the $\hat{\mathbf{X}}_{\text{new}}$. Therefore the number of 1’s in $\hat{\mathbf{X}}_{\text{new}}$ has increased by

$$D(1, 2) = n_3|\mathbf{a}_3| - (n_1|\mathbf{a}_1| + n_2|\mathbf{a}_2|) \quad (14)$$

where we use $|\mathbf{a}_j|$ to denote the number of non-zero values in \mathbf{a}_j . The quantity $|\mathbf{a}_j|$ is also the 1-norm of \mathbf{a}_j , since it is also the sum of all the (1-valued) elements.

Thus $D(j_1, j_2)$ represents the merge cost or ‘merge distance’ between clusters j_1 and j_2 . It is clear that we do not need to keep track of the individual elements (documents) inside each cluster to calculate $D(\cdot, \cdot)$, just the cluster ‘covers’ \mathbf{a}_j and the number of elements n_j in each cluster. The terms $|\mathbf{a}_1|$, $|\mathbf{a}_2|$, and $|\mathbf{a}_3| = |\mathbf{a}_1 \vee \mathbf{a}_2|$, are efficient to calculate for data which is held in a sparse matrix already, which means that $D(\mathbf{a}_1, \mathbf{a}_2)$ is efficient to calculate for sparse data.

In contrast to some approaches to clustering, we have thus constructed a merge cost for *clusters*, rather than an explicit distance or similarity measure between *documents*. Of course, we could simply define a distance between documents (columns of \mathbf{X}) \mathbf{x}_{i_1} and \mathbf{x}_{i_2} to be the cost of merging the singleton clusters holding those documents. In this case, we obtain

$$D_1(i_1, i_2) = 2|\mathbf{x}_{i_1} \vee \mathbf{x}_{i_2}| - (|\mathbf{x}_{i_1}| + |\mathbf{x}_{i_2}|) \quad (15)$$

$$= |\mathbf{x}_{i_1} \vee \mathbf{x}_{i_2}| - |\mathbf{x}_{i_1} \wedge \mathbf{x}_{i_2}| \quad (16)$$

$$= H(\mathbf{x}_{i_1}, \mathbf{x}_{i_2}) \quad (17)$$

i.e. the Hamming distance between the two documents, which we have seen is closely related to the Jaccard and Rand similarity measures.

VI. RESULTS

This algorithm was applied to a subset of the 20-newsgroup dataset. A total 4695 documents were taken from 4 newsgroups: sci.crypt, sci.space, sci.med and soc.religion.christian. The data was pre-processed as in [2] to remove message headers and stop words, forming a binary matrix \mathbf{X} of $m = 5000$ terms by $p = 4695$ documents.

The clustering algorithm was run on Matlab 5, taking 1.3×10^9 flops to reduce $n = 4695$ singleton clusters to $n = 1$ cluster containing all the documents. Fig. 1 shows the code length required to

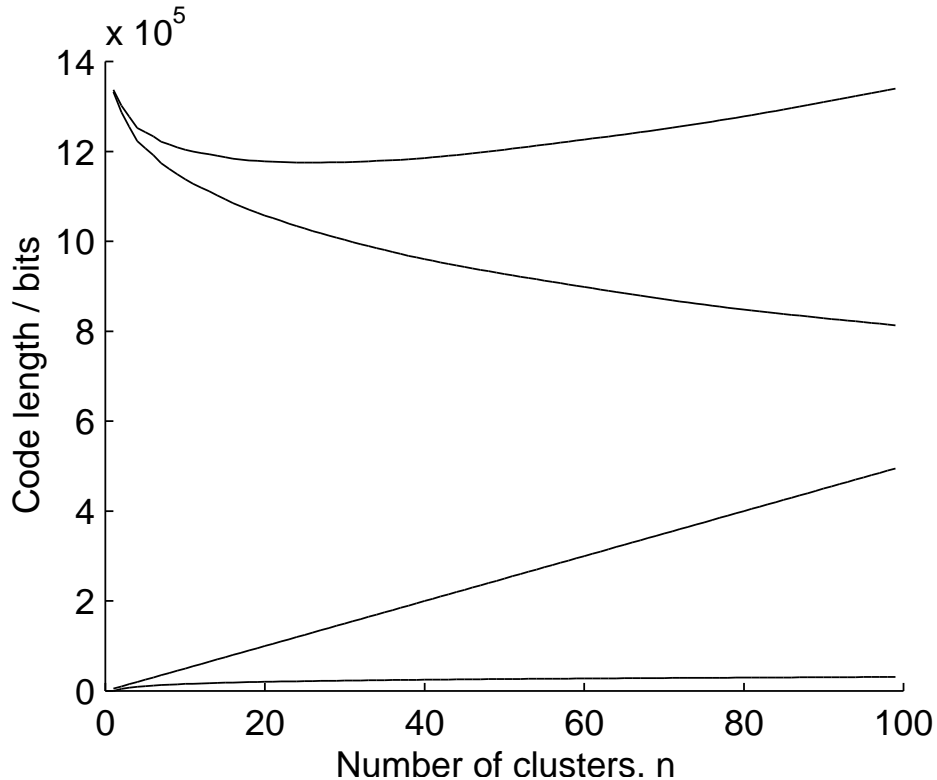


Fig. 1. Code lengths curves for different number of clusters, $1 \leq n < 100$. The figure shows the number of bits required to encode \mathbf{S} (bottom curve), \mathbf{A} (2nd from bottom), and ' \mathbf{X} given $\hat{\mathbf{X}}$ ' (3rd from bottom). The top curve shows the sum of these code lengths.

encode \mathbf{S} , \mathbf{A} and \mathbf{X} for different number of clusters $n < 100$. Recall that under restriction 1 each element of \mathbf{A} requires 1 bit to encode, so its code length reduces linearly with n . The minimum is achieved for $n = 26$, with code length

$$L(\mathbf{A}) + L(\mathbf{S}) + L(\mathbf{X}|\hat{\mathbf{X}}) = (0.130 + 0.022 + 1.023) \times 10^6 \quad (18)$$

$$= 1.175 \times 10^6 \quad (19)$$

corresponding to 5.70×10^6 zeros in $\hat{\mathbf{X}}$ which are not present in \mathbf{X} . The total code length is within 1% of this value between $n = 16$ and $n = 40$, indicating that this does not give a particularly strong indication of how many clusters are ‘best’.

The documents came from known newsgroups (although the algorithm did not have access to the document labels) so we can compare the clustering found by the algorithm with the labelling assigned by the human by posting the message on that particular newsgroup. To see how interpretable the results are, we shall examine the clustering results for a small number of clusters, $n = 10$ and $n = 4$.

A. Result for $n = 10$ clusters

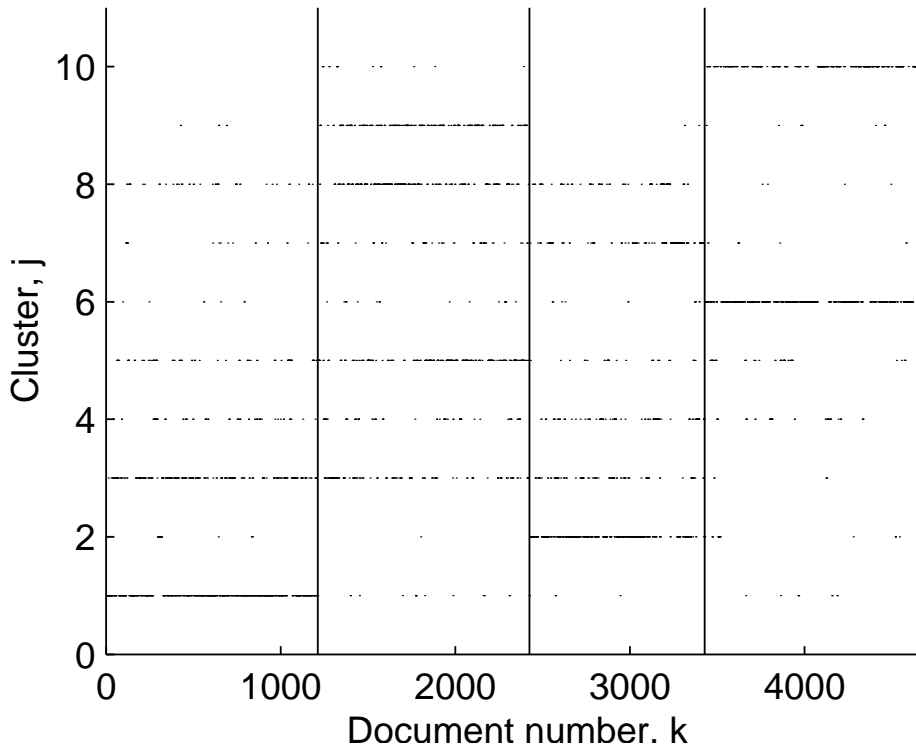


Fig. 2. Scatter diagram for cluster assignments for $n = 10$. Each dot on the figure corresponds to one document in a particular cluster. The vertical lines indicate the cluster boundaries for visual interpretation: this information was not available to the clustering algorithm.

Fig. 2 is a scatter plot giving an overview the clustering achieved at $n = 10$, with Table I showing the corresponding confusion matrix. Some of these clusters contain documents almost exclusively from a single newsgroup: Cluster 1 from sci.crypt; Cluster 9 from sci.med; Cluster 2 from sci.space; Clusters 6 and 10 from soc.religion.christian. Others are more mixed: for example, cluster 3 contains approximately a ratio of 5:3:2 of documents from sci.{crypt:med:space}. If clusters were allocated to the lowest-error newsgroup, the misclassification error would be $1077/4695 = 22.9\%$. This represents ‘training error’ so would be an underestimate of the misclassification error on unseen documents. These results are comparable with the middle range of performances reported by Bingham [2] for linear Independent Component Analysis (ICA) time-series algorithms on the same data.

Table II shows the terms which are present in most documents in each cluster. These give a good sense of the topic covered by the cluster, confirming that e.g. cluster 1 is about encryption, clusters

Cluster	Newsgroup			
	1	2	3	4
1	679	17	3	9
2	26	1	444	22
3	260	152	117	5
4	67	78	119	53
5	96	283	34	51
6	7	21	13	652
7	18	80	168	9
8	52	262	103	5
9	7	301	3	11
10	0	17	0	450

TABLE I

CONFUSION MATRIX FOR $n = 10$. NEWSGROUPS ARE: 1 = SCI.CRYPT; 2 = SCI.MED; 3 = SCI.SPACE; 4 = SOC.RELIGION.CHRISTIAN.

Cluster									
1	2	3	4	5	6	7	8	9	10
kei	space	govern	program	food	christian	space	univers	effect	god
chip	launch	system	univers	research	god	soviet	medic	food	christian
encrypt	system	kei	space	problem	church	object	gener	studi	christ
govern	orbit	david	call	case	homosexu	venu	organ	result	sin
clipper	earth	secur	nasa	univers	christ	show	call	problem	church
secur	develop	point	kei	reason	sin	univers	bank	reason	lord
law	cost	clipper	research	lot	paul	inform	chastiti	human	jesu
system	nasa	mail	system	scienc	jesu	earth	intellect	system	faith
inform	satellit	encrypt	data	scientif	bibl	mission	skeptic	scienc	life
phone	plan	group	design	effect	point	planet	inform	test	psalm
escrow	project	case	scienc	reaction	life	scienc	surrend	diet	bibl
number	commerci	inform	point	group	approv	energi	problem	treatment	live
nsa	support	public	includ	result	love	system	shame	case	book
commun	fund	reason	inform	test	scriptur	surfac	system	eat	love
algorithm	base	world	capabl	studi	discuss	probe	doctor	point	doctrin

TABLE II

COMMONLY-OCCURING KEYWORDS FOR $n = 10$ CLUSTERS

2 and 7 about space, and clusters 6 and 10 about religion. We can see that it is possible for these keywords to appear in more than one cluster: for example, ‘encrypt’ is common in clusters 1 and 3.

We can also see that term ‘system’ is a common term in all clusters except 5, 6, and 10: if we investigate further we find that ‘system’ is in fact present in *all* of the ten clusters. Consequently for $n = 10$, the term ‘system’ is completely non-discriminative: the presence or absence of this term in a given document would not affect the cluster it was assigned to. We might therefore regard this as an emerging stopword (function word), much like ‘a’, ‘the’, and so on, which are commonly removed before document analysis is attempted. In fact, there are 327 non-discriminative words for $n = 10$, with the most common being: god (in 624 documents), system, point, call, gener, problem, reason, case, univers, space, inform, fact, govern, includ, and lot (in 348 documents). Note that while these are non-discriminative for $n = 10$, this does not mean that we would get the same

Cluster				
1	2	3	4	5
pen	lawson	expn	jstmp	vaction
mykotronx	rimsat	csrc	sirtf	delani
jhan	vlbi	edt	brener	workshop
copperud	transpond	sendmail	lili	tangi
concaten	trajectori	ncsl	srt	barbequ
myk	sternberg	sombodi	flower	anaphylact
cylink	schwarzenegg	telnet	internship	skip
scif	swedish	polio	modesti	uvn
schulman	pacastro	biopsi	spacelab	simmon
snark	quantum	vrify	gehrel	unscientif
itar	usaf	smtpt	tuition	weber
unlock	cheapli	sunlight	puriti	stung
crc	srb	alia	unclutt	psychologist
phonecal	blip	codebreak	emblem	mathematician
disarm	standdown	baffl	voltaic	puke
Cluster				
6	7	8	9	10
incarn	venera	andersom	phenylalanin	wrath
prostitut	astrosun	reciproc	sporanox	satisfact
fornic	lazio	ininaugur	excitotox	wedlock
crossroad	cyclotron	grandchildren	toricelli	psalmist
leviticu	venerean	crackdown	prog	beza
deuterocanon	hallei	nitrosiamin	excitotoxin	comprehend
effemin	spike	vagina	daydream	forsak
nrsv	celsiu	gravit	deleter	bernadett
malakoi	fahrenheit	lens	sweeten	unborn
sodomit	innermost	annal	pku	bastard
apocrapha	redshift	cfv	impec	disobei
beget	acf	astrophysicist	aspart	bathsheba
pagan	autocorrel	exam	pasteur	hammerslag
abound	electrophoresi	ppl	ventricular	chapt
lazaru	jelinek	ucsu	toxicologist	rebuk

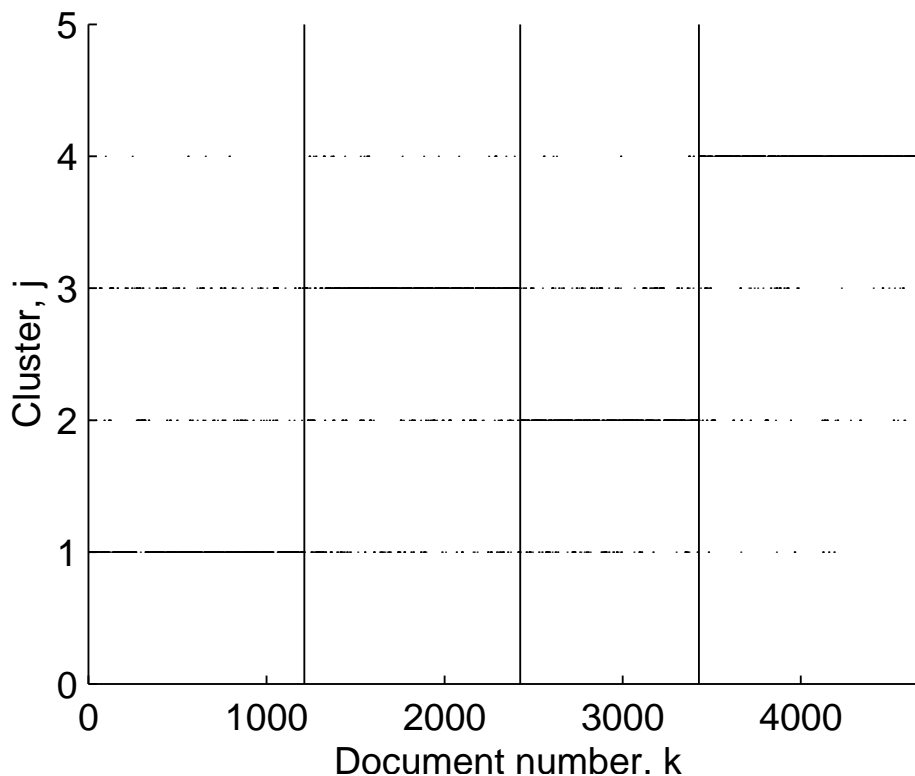
TABLE III
DISCRIMINATIVE KEYWORDS FOR $n = 10$ CLUSTERS.

clusters if we had omitted these at the beginning of the hierarchical clustering procedure: just that once we have reached 10 clusters, removal of these terms would have no effect on maintaining the cluster assignments.

On the other hand, Table III shows what we might call the *discriminative* terms for each cluster: these are terms which appear in the most documents, but in that cluster only. We can immediately see that a large number of technical terms, acronyms and jargon words appear in this table. For example ‘crc’ (acronym for Cyclic Redundancy Check) appears only in cluster 1, ‘quantum’ only in cluster 2, and ‘Pasteur’ only in cluster 9. If we wished to determine the cluster to assign a new document, these would be the critical terms to use in determining the best match.

B. Result for $n = 4$ clusters

For $n = 4$, where the number of clusters is equal to the number of newsgroups, the document scatter diagram is shown in Fig. 3 with the corresponding confusion matrix in Table IV. If the clusters were assigned to the ‘best’ newsgroup, we would have the same error (22.9%) as for $n = 10$,

Fig. 3. Scatter diagram for cluster assignments for $n = 4$.

Cluster	Newsgroup			
	1	2	3	4
1	939	169	120	14
2	111	159	731	84
3	155	846	140	67
4	7	38	13	1102

TABLE IV

CONFUSION MATRIX FOR $n = 4$. NEWSGROUPS ARE: 1 = SCI.CRYPT; 2 = SCI.MED; 3 = SCI.SPACE; 4 = SOC.RELIGION.CHRISTIAN.

indicating that the cluster merging from 10 down to 4 has been consistent with the ‘best’ newsgroup allocations that we would have made for $n = 10$ given the document newsgroup assignment information. These results are again in the middle of those obtained by this task by Bingham [2] using linear ICA approaches.

It is particularly clear from Table IV that cluster 4 is well separated from the others, as we might expect, since the other newsgroups are all science groups. Specifically, only 5% of documents in cluster 4 are not from soc.religion.christian (false positives), with 13% of documents in soc.religion.christian assigned to other clusters (false negatives).

Tables V and VI give the terms in most documents in a given cluster, and the most common discriminative words for each cluster. The commonly occurring terms (Table V) give a good sense of the topic of the cluster, and it is clear from this which cluster corresponds to which newsgroup.

Cluster			
1	2	3	4
kei	space	effect	god
govern	launch	problem	christian
encrypt	system	food	church
chip	orbit	univers	christ
clipper	nasa	reason	sin
secur	univers	case	jesu
system	develop	research	life
law	earth	gener	bibl
inform	program	scienc	love
point	data	result	homosexu
david	cost	lot	point
phone	project	call	lord
escrow	gener	system	approv
nsa	plan	medic	scriptur
public	scienc	studi	faith

TABLE V
COMMONLY-OCCURRING KEYWORDS FOR $n = 4$ CLUSTERS

Cluster			
1	2	3	4
amanda	meter	placebo	testam
pen	galact	allergi	cor
laptop	venera	lyme	incarn
safeguard	lander	carcinogen	corinthian
nist	icbm	soup	taught
paranoid	investor	glu	heresi
beckman	vega	wcsbeau	isaiah
rifl	cosmolog	vitamin	righteou
militia	dock	amino	augustin
mykotronx	mcdonnell	phenylalanin	reincarn
infring	revenu	charcoal	jerom
jhan	interplanetari	chen	prostitut
eavesdrop	esa	sporanox	mormon
dse	astrosun	glucos	inherit
dsg	lazio	albican	fornic

TABLE VI
DISCRIMINATIVE KEYWORDS FOR $n = 4$ CLUSTERS.

However, it is also interesting to consider the discriminative terms (Table VI). These do not ‘represent’ the topics to the same extent that the common words do, and a casual glance does not give the same ‘feel’ of the topics as Table V does. However, as discriminative words they indicate the ‘difference’ between clusters: ‘paranoid’ appears only in cluster 1 (cryptology) and no other; ‘esa’ (European Space Agency) appears only in cluster 2 (space) and no other, and so on.

For $n = 4$ there are 1312 non-discriminative terms, the most common being: god (624), system, point, christian, call, gener, problem, reason, kei, case, univers, space, inform, fact, and govern (373). Note that some of the ‘most representative’ words (words in most documents) for a particular cluster, e.g. ‘kei’ (key) for cluster 1, ‘space’ for cluster 2, and ‘god’ for cluster 4, are in fact non-discriminative. This means that we could omit these words entirely and the current cluster assignments would be unaffected, and the occurrence of these terms in a new document would give us no information about which cluster it should be merged with. This increase in non-discriminative words (function words), as the number of clusters reduces, was also observed by Dhillon et al [15].

VII. BASIS KEYWORDS VS. CONJUGATE KEYWORDS

Many papers on text analysis consider only one set of keywords for a set of documents, while here we have two, which may seem a little strange at first. However, there is a natural parallel for linear systems, such as those used in Independent Component Analysis (ICA) [16], [2], [17].

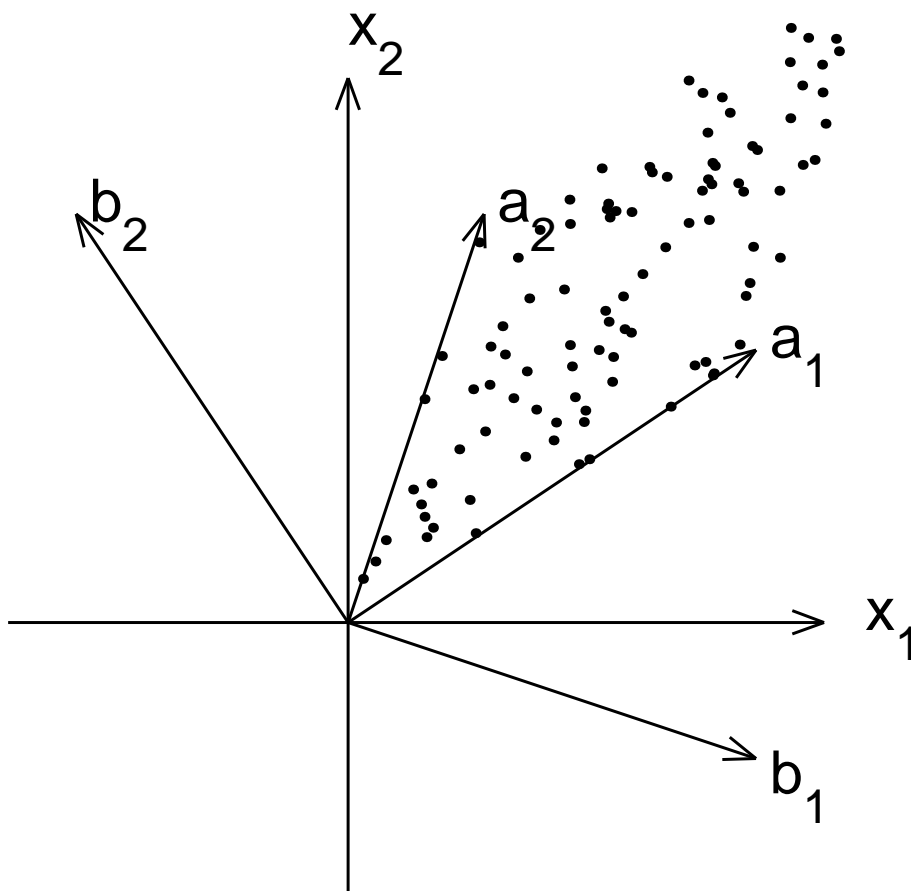


Fig. 4. Scatter plot of data points, with generating basis vectors ($\mathbf{a}_1, \mathbf{a}_2$) and conjugate basis vectors ($\mathbf{b}_1, \mathbf{b}_2$).

Fig. 4 illustrates data points \mathbf{X} being generated according to a linear generative model $\mathbf{X} = \mathbf{A}\mathbf{S}$ where \mathbf{A} is a $m \times 2$ linear matrix, and \mathbf{S} is a $2 \times p$ linear matrix with independent entries. The task in ICA is to discover the separating matrix \mathbf{B} which gives $\mathbf{S} = \mathbf{B}\mathbf{X}$, or some scaling and/or permutation of the rows of \mathbf{S} . Thus the rows \mathbf{b}_1 and \mathbf{b}_2 of \mathbf{B} form a *conjugate basis* to the columns \mathbf{a}_1 and \mathbf{a}_2 of \mathbf{A} .

We draw an analogy with this system, where the basis vectors \mathbf{a}_j correspond to the ‘common keywords’ for documents. These are the terms which may occur in documents in a cluster j , or from a generative viewpoint, these are set of terms from which a generator can select the terms which may be used in a given document. In contrast, the conjugate basis vectors \mathbf{b}_j are those that can ‘unmix’ the sources \mathbf{S} from \mathbf{X} . For the identity-permutation unscaled solution, we have $\mathbf{B}\mathbf{A} = \mathbf{I}_2$, so $\mathbf{b}_i \cdot \mathbf{a}_j = 0$ for $i \neq j$, and for any column \mathbf{x}_k of \mathbf{X} , we have $\mathbf{b}_j \mathbf{x}_k = \mathbf{s}_{jk}$. These correspond to the ‘discriminative keywords’, serving to uniquely identify which source(s) (in our application: which cluster) is active for a particular column (document) \mathbf{x}_k .

Thus the basis vectors \mathbf{a}_j are ‘representative’ of the appearance of clusters or sources in \mathbf{X} , since this gives the contribution to \mathbf{X} made by each source, while the conjugate basis vectors \mathbf{b}_j are ‘discriminative’ for each cluster or source, since they find the direction in \mathbf{X} -column space (set of terms) which are orthogonal to all sources (clusters) apart from the one of interest. This suggests that we might refer to the commonly-occurring keywords as *basis keywords*, with the discriminative keywords as *conjugate keywords*.

For a single set of labels for human interpretation, it may be possible to produce a labelling which is a mixture of these two types of basis vectors. One possible approach might be to use the ‘frequent and predictive’ words cluster labelling method proposed by Popescul and Ungar [18].

VIII. DISCUSSION

It is interesting to note that the concept of a *cover*, the set of all items appearing in *any* member of a cluster, is complementary to the idea of an *itemset*, the set of items appearing in *all* members of a cluster, which is used to generate association rules for market basket transaction data [19], [20]. A similar concept of word-intersection clustering (Word-IC), using the set of words shared by all documents in a cluster, has been suggested by Zamir et. al. [21]. However, Clifton et al [22] found that entities that they expected to be grouped as a topic would not show up as a frequent itemset, i.e. that no article contained all of the words they expected. We suspect that this asymmetry is due to the sparsity of the data, meaning that itemsets (intersections) may be unsuitable for large clusters. It will take many merged documents before the cover includes all words, but it will take less documents before an itemset reduces to the empty set.

Note that the minimal covers used here should not be confused with the minimal covers used in the CLIQUE algorithm of Agrawal et al [23]. CLIQUE uses covers in subspaces of a high-dimensional numeric data space rather than over a binary data matrix as we use it here.

We saw that our developed measure reduces to the Hamming distance (L_1 norm) in the case of singleton clusters. Aggarwal et al [24] have indicated that the L_1 norm is more suitable for high dimensional data than L_p for larger values of p , such as the euclidean distance (L_2 norm). It may be interesting to compare this cover-based merge distance with a euclidean centroid-based merge distance in more detail.

We used a simple agglomerative hierarchical clustering algorithm to construct the clusters. However, while it is conceptually simple, and requires only a cluster merge distance, it does have a number of disadvantages. For example, no re-allocation of cluster members is allowed once merging has taken place (so unlike with k-means algorithms early ‘mistakes’ cannot be rectified later), and

k-means algorithms are typically more computationally efficient than the agglomerative hierarchical approach [14].

We are investigating possible methods to apply a k-means approach to the cover and merge cost used here. While this is not entirely straightforward, algorithms such as k-modes, related to k-means, are already available for categorical data [25]. It may be that a *first-variation* approach, whereby each document is moved from one cluster to another with immediate update of the cluster cover [26], may provide the approach that we need.

Using the idea of a measure of compression to drive document clustering has also been proposed by Thaper [27], although without the simplifying restrictions imposed here, and is directly related to ideas from information theory [28]. Let us consider what would happen to our system were we to relax some of the restrictions that we have imposed to create our simple system.

Restriction 3 requires $\hat{\mathbf{X}}$ to be a cover for \mathbf{X} , hence requiring that the cluster ‘centroids’ \mathbf{a}_j must each be a cover for all the documents in that cluster. If we were to relax this restriction, it might be possible to reduce the code length to assign a ‘0’ in $\hat{\mathbf{X}}$ to terms which very rarely appear in \mathbf{X} . We would then need to allocate a finite (although probably large) code length l_{10} (the code length for sending $x_{ik} = 1$ when $\hat{x}_{ik} = 0$) but this may still represent an overall saving since the number of unnecessary zeros could be reduced in $\hat{\mathbf{X}}$. It would be interesting to consider whether this is related to removal of infrequent words during pre-processing. Typically, infrequently occurring words (e.g. less than 5 occurrences) are removed from any dataset before pre-processing. Perhaps we would find that a large but non-infinite code length for l_{10} would be used for words occurring less than some small N_{\min} times, and so might be ignored in the construction of $\hat{\mathbf{X}}$, and hence in the construction of the cluster centroids \mathbf{A} .

Restriction 2 forces only a single non-zero element (i.e. only one ‘1’) in each column of \mathbf{S} , leading to a clustering algorithm where each document can be in only one cluster. If we were to allow more than one non-zero element, we would obtain something closer to the topic models observed with Latent Semantic Indexing (LSI) [3]. Each document would be considered to be ‘about’ one or more underlying topics, with the words chosen according to the set of topics represented. With binary \mathbf{A} and \mathbf{S} and restriction 3 in force, each document could be considered to be generated from the set of terms formed from the union of the sets of terms corresponding to all topics. Interestingly the coding model $\hat{\mathbf{X}} = \mathbf{A} \circ \mathbf{S}$ would now be symmetrical in \mathbf{A} and \mathbf{S} : any method developed for this should also be applicable to the document-by-term coding scheme $\hat{\mathbf{X}}^T = \mathbf{S}^T \circ \mathbf{A}^T$.

However, while many algorithms have been developed for clustering algorithms where each document appears in only one document, there has been less work on these ‘multicluster’ systems, particularly for binary data. The most relevant work here is probably that in topic models [9], where typically a linear model such as a factor model derived from LSI [3] or ICA [2], [17] is used to represent the data. However, even in these approaches it is often assumed that documents are ‘mostly’ about a single topic: the concept of ϵ -separability being one example of this [9]. Therefore it may be possible to construct an algorithm that starts from a single-topic clustering, and refines this to allow multiple topics for documents that require it. While this would be an asymmetrical approach (we would be less happy with the idea that each word could only be used in one topic) this may reflect an asymmetry in the document analysis problem itself that we could and should exploit.

As a more general approach, Sahami et al [29] used Saund’s Multiple Cause Mixture Model [30], attempting to model binary text data using a *soft-disjunction* or *Noisy Or* generative model. They reported that the model did allow documents to be included in more than one category, although the gradient-based search method was very computationally intensive.

Restriction 1 allows us to ignore the code lengths for \mathbf{S} and \mathbf{A} , in a similar way to the way that maximum likelihood methods do not need to factor in the prior probabilities used by a true Bayesian MAP approach. It would be possible to relax this restriction: the code lengths for \mathbf{S} , \mathbf{A} and $(\mathbf{X}|\hat{\mathbf{X}})$ would still only be dependent on the count of 1's and 0's (and their pairings for $(\mathbf{X}|\hat{\mathbf{X}})$) so should be reasonably efficient. However, we should notice that this scheme still proposes that the same code length will be used for all elements of \mathbf{A} , for example, irrespective of their individual frequency information. Were we to relax this condition, we may need a different code length for each element of \mathbf{A} , and we would no longer be able to assume that the number of bits required to send the encoding length itself would be negligible. To relax this restriction it may be more sensible to use a standard Bayesian approach.

In passing, we notice that the encoding framework gives us a natural way to extend the model to handle incomplete data, without requiring unknown data to be imputed [31]. We would need to encode an additional matrix to indicate if any data element were missing: for only a few missing values, this would be sparse and therefore relatively inexpensive to encode. In particular, it would not depend on the values chosen for \mathbf{A} and \mathbf{S} . Missing values would not need to be encoded, so their code length (and hence match cost for that parameter) would be ignored.

Finally, we note that it should be possible to extend the coding model to allow co-clustering [32]. Here the words are also assigned into clusters which are expected to appear in similar contexts, which allows the document clustering to be performed in a reduced dimension. To extend our model we would need an encoding structure of the form $\hat{\mathbf{X}} = \mathbf{R} \circ \mathbf{A} \circ \mathbf{S}$ where \mathbf{R} is our word clustering matrix and \mathbf{A} is now a matrix that associates word clusters with topics. For a simple model, we might choose to set each row of \mathbf{R} to contain exactly one '1', so that each word appears in only one word cluster, although a more general model could be constructed. Since the resulting method would be related to information transmission, it would be interesting to compare with the 'Information Bottleneck' method of Tishby et al [33].

IX. CONCLUSIONS

We have introduced an approach to clustering of data with Boolean attributes based on an information-theoretic minimum description length (MDL) approach. Some simplifying restrictions allowed us to derive what appears to be a novel yet simple cluster merge cost, based on the concept of a *minimum cover*.

This merge cost is efficient to calculate, and is closely related to other well-known distance and similarity measures on single documents. The clustering was demonstrated on a subset of the 20-Newsgroup corpus.

The MDL encoding framework indicates that many improvements and extensions to this model are may be possible, including more efficient clustering along the lines of k-means, relaxing the absolute cover requirement to allow for outliers, and handling of missing data values without imputation.

X. ACKNOWLEDGMENTS

The author would like to thank Ella Bingham for many discussions and contributions that led to the work in this paper, and for supplying the pre-processed data used in section VI. Part of this work was undertaken while the author was visiting the Laboratory of Computer Science at Helsinki University of Technology, supported by a Study Abroad Fellowship from the Leverhulme Trust. The author would like to thank the Laboratory of Computer and Information Science at HUT for much help and support during the visit, as well as various members of the Neural Networks Research Centre for many interesting and stimulating discussions. This work is also partly supported by

grant GR/R54620 from the UK Engineering and Physical Sciences Research Council.

REFERENCES

- [1] S. Guha, R. Rastogi, and K. Shim, "ROCK: A robust clustering algorithm for categorical attributes," *Information Systems*, vol. 25, no. 5, pp. 345–366, 2000.
- [2] E. Bingham, "Topic identification in dynamical text by extracting minimum complexity time components," in *Proceedings of the 3rd International Conference on Independent Component Analysis and Signal Separation (ICA2001)*, 9–13 December 2001, pp. 546–551.
- [3] T. Hofmann, "Probabilistic latent semantic indexing," in *ACM SIGIR 99*, 1999, pp. 50–57.
- [4] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, pp. 465–471, 1978.
- [5] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, Englewood Cliffs, New Jersey, 1988.
- [6] P. Berkhin, "Survey of clustering data mining techniques," 2002, Accrue Software Research Paper. Available online at <http://www.acrue.com/products/researchpapers.html>.
- [7] J. J. Oliver, R. A. Baxter, and C. S. Wallace, "Unsupervised learning using MML," in *Machine Learning: Proceedings of the Thirteenth International Conference (ICML 96)*. 1996, pp. 364–372, Morgan Kaufmann Publishers.
- [8] P. Pajunen, "Blind source separation using algorithmic information theory," *Neurocomputing*, vol. 22, no. 1-3, pp. 35–48, Nov 1998.
- [9] E. Bingham, H. Mannila, and J. Seppänen, "Topics in 0-1 data," in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada, July 23–26 2002, pp. 450–455.
- [10] D. D. Lewis, "Naive (Bayes) at forty: The independence assumption in information retrieval," in *Proceedings of ECML-98, 10th European Conference on Machine Learning*, C. Nédellec and C. Rouveirol, Eds., Chemnitz, DE, 1998, pp. 4–15, Springer Verlag, Heidelberg, DE.
- [11] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society Series B*, vol. 39, pp. 1–38, 1977.
- [12] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," in *Proc. 1996 ACM-SIGMOD Int. Conf. Management of Data*, Montreal, Canada, 1996.
- [13] M.-S. Chen, J. Han, and P. S. Yu, "Data mining: an overview from a database perspective," *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 866–883, Dec. 1996.
- [14] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *KDD Workshop on Text Mining*, 2000.
- [15] I. S. Dhillon and D. S. Modha, "Concept decompositions for large sparse text data using clustering," *Machine Learning*, vol. 42, no. 1/2, pp. 143–175, Jan. 2001.
- [16] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, John Wiley & Sons, 2001.
- [17] Y.-H. Kim and B. Zhang, "Document indexing using independent topic extraction," in *Proceedings of the International Conference on Independent Component Analysis and Signal Separation (ICA2001)*, San Diego, California, Lee, Jung, Makeig, and Sejnowski, Eds., December 9-13 2001, pp. 557–562.
- [18] A. Popescul and L. H. Ungar, "Automatic labeling of document clusters," 2000, Unpublished manuscript. Available at <http://www.cis.upenn.edu/~popescul/publications.html>.
- [19] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining association rules between sets of items

- in large databases,” in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, P. Buneman and S. Jajodia, Eds., Washington, D.C., May 26–28 1993, pp. 207–216.
- [20] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules,” in *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, J. B. Bocca, M. Jarke, and C. Zaniolo, Eds. 12–15 Sept. 1994, pp. 487–499, Morgan Kaufmann.
- [21] O. Zamir, O. Etzioni, O. Madani, and R. M. Karp, “Fast and intuitive clustering of web documents,” in *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, 1997, pp. 287–290.
- [22] C. Clifton and R. Cooley, “Topcat: Data mining for topic identification in a text corpus,” in *Proceedings of the 3rd European Conference of Principles and Practice of Knowledge Discovery in Databases, Prague, Czech Republic*, 1999, pp. 174–183, Updated version (with J Rennie) submitted to *IEEE Transactions on Knowledge and Data Engineering*, December 2001.
- [23] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, “Automatic subspace clustering of high dimensional data for data mining applications,” in *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data*, Seattle, Washington, June 1998, pp. 94–105.
- [24] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, “On the surprising behavior of distance metrics in high dimensional space,” in *Proc. Of 8th International Conference on Database Theory, ICDT 2001*, London, 2001, pp. 420–434.
- [25] Z. Huang, “A fast clustering algorithm to cluster very large categorical data sets in data mining,” in *Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD’97)*, 1997.
- [26] I. S. Dhillon, Y. Guan, and J. Kogan, “Refining clusters in high dimensional text data,” Jan. 7 2002, Unpublished manuscript, available at <http://www.cs.utexas.edu/users/yguan/resume.htm>.
- [27] N. Thaper, “Using compression for source based classification of text,” M.S. thesis, Dept of Electrical Engineering and Computer Science, MIT, Feb. 2001.
- [28] D. Lin, “An information-theoretic definition of similarity,” in *Proc. 15th International Conf. on Machine Learning*. 1998, pp. 296–304, Morgan Kaufmann, San Francisco, CA.
- [29] M. Sahami, M. A. Hearst, and E. Saund, “Applying the multiple cause mixture model to text categorization,” in *Proceedings of ICML-96, 13th International Conference on Machine Learning*, L. Saitta, Ed., Bari, IT, 1996, pp. 435–443, Morgan Kaufmann Publishers, San Francisco, US.
- [30] E. Saund, “A multiple cause mixture model for unsupervised learning,” *Neural Computation*, vol. 7, pp. 51–71, 1995.
- [31] C. C. Aggarwal and S. Parthasarathy, “Mining massively incomplete data sets by conceptual reconstruction,” in *Proceedings of the Seventh ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, San Francisco, California, USA, 2001, pp. 227–232.
- [32] I. S. Dhillon, “Co-clustering documents and words using bipartite spectral graph partitioning,” in *Proceedings of the Seventh ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, San Francisco, California, USA, Aug. 26–29 2001, pp. 269–274.
- [33] N. Tishby, F. C. Pereira, and W. Bialek, “The information bottleneck method,” in *Proceedings of the 37th Annual Allerton Conference on Communication, Control and Computing*, 1999, pp. 368–377.