A State-of-the-Art Toolkit for Document Clustering

Derek Greene

A thesis submitted to the University of Dublin, Trinity College

in fulfillment of the requirements for the degree of

Doctor of Philosophy

March 2007

Abstract

Cluster analysis refers to a family of procedures which are fundamentally concerned with automatically arranging data into meaningful groups. These procedures are increasingly being employed in knowledge discovery tasks to assist in the exploration and interpretation of large datasets. Since users may often be unfamiliar with the exact contents of a dataset, clustering can provide a means of introducing some form of organisation to the data, which can also serve to highlight significant patterns and trends.

Cluster analysis methods have recently become an important part of commercial and industrial applications for mining data in a variety of domains. In the past, these methods have also been employed to facilitate the discovery of knowledge from large collections of unstructured text. Renewed interest in document clustering has been prompted by the exponential growth in the size of digital document collections, including web pages, e-mail messages and news articles. Another motivating factor has been the increased availability of computing resources, which has encouraged researchers to reconsider the application of machine learning methods to larger corpora.

A variety of techniques for generating and evaluating clusterings of text data have been proposed in the literature, which differ significantly in terms of their theoretical foundations and practical implementation. A significant obstacle for researchers interested in harnessing this work is the absence of a comprehensive framework for comparing and extending these techniques. In this thesis, we introduce the *Text Clustering Toolkit* (TCT), a state-of-the-art framework supporting the development of applications for unsupervised text mining tasks. The toolkit covers all phases of the cluster analysis process, from the preprocessing of raw documents to the interpretation of a final clustering solution. As well as allowing researchers to evaluate popular learning algorithms, TCT also provides a flexible test-bed for the design and the development of novel clustering procedures.

The primary focus of a significant body of research in document clustering has been concerned with the production of solutions that are “accurate” in the sense that they succeed in revealing the underlying structure of a dataset. This thesis proposes a selection of novel clustering algorithms, which frequently succeed in accurately identifying the natural trends and groups in document collections. We also describe new strategies to improve the accuracy afforded by existing algorithms, such as those based on kernel learning. A secondary objective, which is frequently overlooked in this area, is the provision of information to help a user interpret the output of an analysis procedure. To support the extraction of knowledge from clustering solutions, techniques are described for producing summary information, in the form of human-interpretable cluster labels.

The increase in computing power available to researchers has opened up many new possibilities, such as the study of methods that involve aggregating information obtained from multiple clusterings. This information can often provide additional insight regarding the group structures in a dataset. However, due to the exceptionally large size and high-dimensional nature of real-world document collections, the computational cost of applying aggregation methods to text data will generally be prohibitive. To address this scalability problem, we introduce novel techniques for efficiently generating and combining a diverse collection of clusterings to produce more accurate document clustering solutions.

A particularly problematic aspect of many common unsupervised learning procedures relates to the selection of key algorithm parameters, which can greatly dictate the success of the procedure. To this end, we explore ways in which the aggregation of information derived from a large collection of clusterings can provide us with clues about the validity of a clustering model. We devote particular attention to the estimation of the number of natural groups or topics in a text corpus, which represents a fundamental issue when employing well-known document clustering algorithms.

Contents

List of Figures

List of Tables

Associated Publications

Greene, D. & Cunningham, P. (2006). Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd International Conference on Machine learning (ICML'06)*, 377–384, ACM Press.

Greene, D. & Cunningham, P. (2006). Efficient prediction-based validation for document clustering. In *Proceedings of the 17th European Conference on Machine Learning (ECML'06)*, 663–670, Springer Berlin.

Greene, D. & Cunningham, P. (2005). Producing accurate interpretable clusters from high-dimensional data. In *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'05)*, 486–494, Springer.

Greene, D., Tsymbal, A., Bolshakova, N. & Cunningham, P. (2004). Ensemble clustering in medical diagnostics. In *Proceedings of the 17th IEEE Symposium on Computer-Based Medical Systems (CBMS'04)*, 576–581, IEEE Computer Society.

Chapter 1

Introduction

1.1 Cluster Analysis

Cluster analysis refers to a family of procedures which are fundamentally concerned with automatically arranging data into meaningful groups. The grouping process, generally referred to as *clustering*, attempts to divide a set of data objects so that those assigned to the same group share common characteristics, while those assigned to different groups are conceptually unrelated. In certain situations, this form of data analysis may be used to verify whether or not a dataset contains patterns that are assumed to exist according to a particular hypothesis. For other purposes, the identification of clusters represents the initial phase in a larger application, where it may be used as a means of summarising or compressing data. However, cluster analysis is increasingly being employed as an important tool in knowledge discovery tasks. In this context, it forms an integral part of exploratory data analysis, where users may be unfamiliar with the exact contents of the data and may wish to introduce some form of organisation, or identify important trends.

In the machine learning community, cluster analysis procedures are often referred to as *unsupervised learning* methods to emphasise the absence of any form of external feedback. This is in contrast to *supervised learning* methods, which use training examples from a fixed number of predefined classes to learn a model. Clustering algorithms must attempt to identify patterns by relying solely on the intrinsic characteristics of a limited sample of data, without referring to any *a priori* class information. Consequently, the same set of data objects may be grouped in many different ways, depending upon various aspects of the clustering model employed, such as the desired number of clusters. A simple illustration

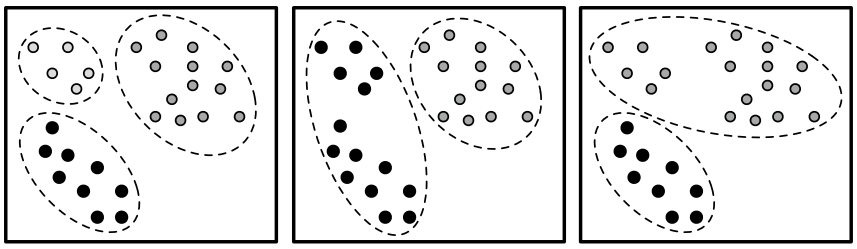


Figure 1.1: Example of three alternative clusterings for the same set of data objects.

of this is shown in Figure 1.1.

1.1.1 Common Applications

The use of cluster analysis methods can be found in the literature of many academic disciplines, ranging from geography to sociology, where they have proved useful in identifying patterns and trends in data. Recently, these methods have also become an important part of industrial and commercial applications for mining data in a variety of areas. Well-known examples include:

Bioinformatics: Microarray technology makes it possible to simultaneously examine the behaviour of thousands of genes under different experimental conditions. Cluster analysis techniques may subsequently be applied to the resulting expression data to identify functional groups of genes (Kluger *et al.*, 2003).

Image analysis : When working on images in tasks such as content-based image retrieval and object recognition, it is often desirable to identify smaller regions of interest for further processing (Lau & Levine, 2002). This is particularly useful when dealing with large, complex images such as those analysed in biomedical applications.

Multimedia signal processing: It may also be useful to identify groups of related scenes from whole video sequences, to aid browsing and navigation. Similar techniques may be employed to automate the detection of unusual activity or specific events in long sequences of video footage (Zhong *et al.*, 2004).

Marketing research: Customer records may be organised based on demographic attributes or purchasing patterns. These groups may be subsequently used to identify customers who may be interested in new products or services (Punj & Stewart, 1983).

1.2 Clustering Document Collections

In addition to the examples enumerated previously, cluster analysis has also formed an integral part of text mining applications, where it has been used to facilitate the discovery of knowledge from large collections of natural language text. In this context, the task of clustering represents an attempt to impose some form of structure on a collection by identifying interesting groups of documents sharing a common topic or theme. This problem has been studied for some time (*e.g.* Jardine & van Rijsbergen, 1971), but has seen little progress until relatively recently. However, renewed interest in document clustering has been prompted by the exponential growth in the size of unstructured digital document collections, both in the form of publicly available resources such as the World Wide Web and private, domain-specific textual databases. Another motivating factor has been the improvement in computing resources, which has encouraged researchers and users alike to reconsider the application of unsupervised learning methods in this area. Unlike text classification tasks, which require a set of labelled training examples to be assembled beforehand to provide supervision, cluster analysis procedures allow users to explore document collections without necessitating an initial training phase or requiring any form of prior knowledge regarding possible groupings.

Document clustering tasks can generally be divided into two categories: *offline* techniques that seek to cluster a static, previously compiled collection of documents, and *online* techniques that operate on an incrementally compiled set of documents. In both cases, the application of a clustering procedure is intended to allow the user to browse and explore a collection more effectively. We follow the popular trend of research in this area and focus on offline document clustering problems.

1.2.1 Cluster Analysis Workflow

Formal research concerning the problem of cluster analysis dates back over four decades (*e.g.* Forgy, 1965). However, the majority of work in the literature of this area has been contributed over recent years, during which time a vast range of algorithms have been proposed that vary greatly in their theoretical foundations and practical details. Most commonly, these algorithms involve producing a *hard clustering* of a dataset, which represents a disjoint partition where each object belongs to one and only one cluster. In some cases, the clusters are arranged in a nested fashion to produce a tree-like model of

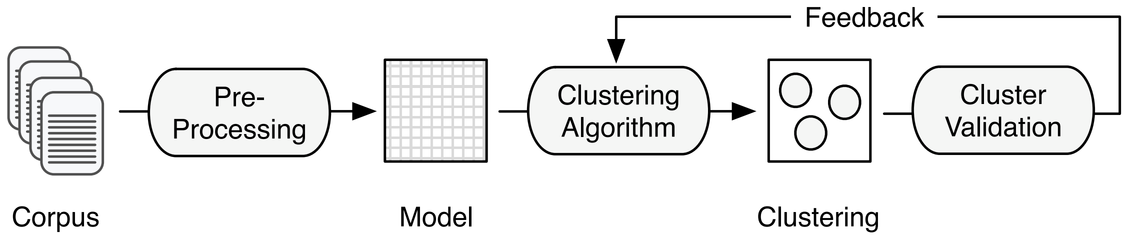


Figure 1.2: Generic document clustering workflow for a knowledge discovery task, which consists of three fundamental phases: *preprocessing*, *clustering* and *validation*.

concepts (Voorhees, 1986). Instead of assigning each object to a dedicated cluster, other authors have proposed methods to generate a *soft clustering* of data, which allows clusters to overlap so that objects may belong to several clusters to different degrees (Bezdek, 1981). Often soft cluster membership weights are constrained to take probabilistic values, resulting in a *fuzzy clustering* of the data. In this case a weight of 0 indicates that an object does not belong to a cluster at all, while a weight of 1 indicates that an object belongs entirely to a cluster. In general, a soft clustering may be converted to a hard clustering by assigning each object to the cluster for which it has the highest membership weight.

For many applications, it is customary to represent a collection of data objects in terms of a fixed number of attributes or features, so that each object can be viewed as a point in a multi-dimensional *feature space*. In the case of text corpora, these data objects represent individual documents and their features generally correspond to unique words or phrases from the vocabulary of the corpus. As an alternative, other clustering algorithms have been proposed that operate on a representation which describes each object in terms of its pairwise relations with other objects. In practice, this representation will typically take the form of a similarity or dissimilarity matrix. A benefit of this approach is that the repeated computation of similarity values in the original space can be avoided, although the clustering algorithm may no longer have access to the raw feature values.

While individual clustering algorithms may differ significantly, offline document clustering tasks applied in the context of knowledge discovery typically proceed as shown in Figure 1.2. The three fundamental phases in this workflow may be summarised as follows:

1. *Preprocessing:* Transform the collection of raw, unstructured text documents into a suitable model for clustering. It may often be necessary to adjust this representation

to improve clustering performance, such as by applying normalisation techniques or by removing unnecessary features from the model.

2. *Clustering:* Select a suitable clustering algorithm and a corresponding set of parameter values. Apply the chosen algorithm to the data model to produce a clustering solution.

3. *Validation:* Quantitatively evaluate the quality of the newly generated clustering solution. It may also be useful at this stage for a user to subjectively examine and interpret the clusters.

In many real-world scenarios it may be necessary to repeatedly adjust parameter values and reapply the clustering algorithm until a useful or interesting solution is obtained.

1.2.2 Goals and Challenges

The primary focus of a significant body of research in cluster analysis has been on producing *accurate* clusterings of data. Unlike supervised tasks, where classification accuracy is a well-defined concept, the lack of a definitive set of classes means that there is no categorical definition of what constitutes a correct clustering. However, it is generally agreed that, where possible, an accurate clustering should reveal the "natural classes" present in the data (Strehl, 2002). For a text corpus, this could correspond to groups of documents corresponding to a set of topics or themes. In practice, it may be the case that several meaningful groupings of the same data exists, depending upon the number of clusters chosen. Another aspect of cluster analysis that is often overlooked is the provision of information to facilitate the human interpretation of the output of a clustering algorithm. When working with text data, the ability to generate cluster labels that summarise the salient concepts present in groups of documents can help users to effectively understand a clustering solution.

Prior to the advent of machine learning techniques for document clustering, collections had to be manually organised by domain experts, a time-consuming task that is generally impractical for all but the smallest corpora. Although improved computing resources has made the automation of such tasks feasible, the development of algorithms that successfully achieve the goals of document clustering has continued to present computer scientists with a variety of challenges, with the most prominent of these being:

Scalability: Even with the availability of significant processing power, an obstacle that is constantly present when applying machine learning techniques to textual databases is that of dealing with the sheer volume of data available, with some repositories containing millions of document. Current trends suggest that the growth in the size of data repositories will continue to outstrip computational power. Thus, an acceptable trade-off must be found between clustering accuracy and algorithm running time.

Dimensionality: In natural language corpora, the size of the vocabulary is often extremely large. Scalability problems may be exacerbated by the high dimensionality of text documents when they are represented as points in a feature space. Specifically, each unique word or phrase will typically require an additional dimension, and the time required to run many classical clustering techniques can increase rapidly as the number of dimensions increases. The sparsity of the feature space can also impact upon an algorithm's ability to produce accurate clusterings.

Robustness: Another problem derives from the fact that most real-world data will contain *noise*. For text documents, this could be due to the presence of irrelevant terms and errors introduced by misspellings, typographical mistakes or OCR recognition faults. Noise can also take the form of outlying documents that do not fit into any of the underlying groups in the data. In presence of such noise, many algorithms will not be effective in producing an accurate, definitive clustering solution.

Model selection: A particularly problematic aspect of many clustering procedures is the selection of key algorithm parameters, such as the desired number of clusters. These decisions can greatly influence the outcome of the procedure. However, in many cases it is unclear as to how an appropriate clustering model should be determined.

Validation: A related problem is that of determining how to produce a quantitative evaluation of the quality of a given clustering solution. As there is no universally accepted definition of what constitutes an accurate clustering, it may be difficult to automatically distinguish between a solution consisting of groups that accurately reflect the patterns in the data and one that does not provide a user with any useful insight.

1.3 Contributions

In this section we summarise the specific contributions of this thesis.

1.3.1 Toolkit for Document Clustering

A major output of our work has been the development of the *Text Clustering Toolkit* (TCT), a state-of-the-art framework providing data analysts with access to a range of well-know classical and contemporary document clustering procedures. The toolkit includes implementations of techniques covering all phases of the cluster analysis process, from the application of preliminary preprocessing procedures to the final assessment and interpretation of a newly generated clustering solution. While the primary focus of our work has been on text data, an important design goal for TCT has been to support the deployment of learning applications in other domains. Therefore, the design of the toolkit is structured around a layer-based architecture, containing modular components that can be readily reused in different tasks.

1.3.2 Improving Accuracy and Interpretability

Accuracy and interpretability represent two fundamental objectives in document clustering. In this thesis, we introduce several novel clustering algorithms designed to produce accurate clusterings of large document collections. Empirical evaluations on real-world datasets demonstrate that these algorithms frequently succeed in identifying the natural trends and groupings in the data. To facilitate the discovery of knowledge from the resulting clustering solutions, we propose complementary strategies for generating readily interpretable summary information, in the form of both *descriptive* and *discriminative* cluster labels.

1.3.3 Aggregating Information from Multiple Clusterings

Recent work in machine learning has involved the study of methods that involve generating many different clusterings of the same dataset and aggregating these solutions in a manner that extracts the most information about the underlying structures in the data (Fred, 2001). However, due to the exceptionally large size and high-dimensional nature of many document collections, scalability issues have meant that such aggregation methods have rarely been applied to this type of data. To address these issues, we propose efficient algorithms for combining information from multiple clusterings to produce a more accurate, definitive solution. We also demonstrate that, by analysing the level of agreement between a large set of clusterings, we can gauge the suitability of a given clustering model. This work is particularly relevant to the problem of estimating the number of clusters in a dataset, which is a fundamental question when employing document clustering algorithms. As always, we seek to perform this analysis in a manner that is not prohibitively expensive when working with large corpora.

1.3.4 New Benchmark Datasets

In the task of evaluating novel clustering and validation methods, the provision of annotated and well-understood corpora is highly useful. To this end, during the course of our research we have constructed two new high-quality text corpora, the *bbc* and *bbcsport* collections. These are composed of news articles relating to a variety of themes. An advantage of producing corpora of this type stems from the fact that the data and any generated clusterings may be interpreted by users without the need for specialist domain knowledge. In addition, we have constructed 84 artificial datasets, assembled from existing resources, which are specifically designed to evaluate the ability of cluster analysis methods to perform successfully on data containing natural classes of differing complexity and structure.

1.4 Thesis Organisation

Chapter 2: Algorithms for Document Clustering presents a comprehensive review of classical and state-of-the-art algorithms that have previously been applied for the task of document clustering. In addition, we examine related areas such as techniques for preprocessing raw document collections and methods for reducing the dimensionality of the resulting data model.

Chapter 3: Cluster Validation Methods for Text Data focuses on the general task of quantitatively assessing the quality of clustering solutions. We provide a survey of existing methods which may be employed to compare the relative accuracy of document clustering algorithms. We also examine strategies intended to select the most suitable clustering model from among several possible candidates.

Chapter 4: Text Clustering Toolkit introduces TCT, a new Java-based toolkit for document clustering, which provides researchers with the ability to compare and extend popular cluster analysis procedures. We describe the architectural design and functionality of the software, and provide practical details about its implementation and usage.

Chapter 5: Baseline Analysis contains a description of the experimental procedures and datasets used in empirical evaluations throughout this thesis. Furthermore, we provide a comparison of classical document clustering and validation methods, with a discussion of their advantages and limitations. This evaluation also provides us with a baseline for the assessment of approaches proposed in later chapters.

Chapter 6: Improving Accuracy and Interpretability presents novel techniques to improve the performance of modern clustering algorithms on high-dimensional data, while also ensuring that the resulting solutions may be easily understood by a user.

Chapter 7: Aggregating Information from Multiple Clusterings explores ways in which multiple clusterings may be generated and combined to provide users with a greater insight into a collection of documents. Specifically, we describe both clustering and validation techniques, based on this concept, which are practical for use on large text datasets. A full comparison and analysis of the performance and efficiency of these techniques is included.

Chapter 8: Conclusions summarises the findings of this thesis and discusses potential avenues for future research.

Chapter 2

Algorithms for Document Clustering

2.1 Introduction

A vast array of algorithms for grouping data have been proposed, not only by members of the machine learning community, but also by researchers in many other academic disciplines. These algorithms vary significantly in terms of their theoretical foundations and the details of their practical implementation, and many are not directly applicable when working with text data. In this chapter we provide a comprehensive survey of similarity-based algorithms, both classical and contemporary, which are relevant to the task of document clustering. These algorithms can be divided into five broad categories:

1. *Partitional algorithms* involve the generation of a flat grouping of the data objects, typically by performing an iterative refinement process that attempts to optimise a given objective function. The generated clusterings may either by disjoint or overlapping, and the algorithms often contain a stochastic element, where the optimisation process begins from a randomly generated initial solution.

2. *Hierarchical algorithms* produce a tree-like structure of nested clusters, which may be constructed using either a top-down or bottom-up strategy. These algorithms are often deterministic in that, for a given dataset, they will consistently produce the same single, definitive solution.

3. *Matrix decomposition methods* apply well-known techniques from linear algebra, such

as spectral analysis or sparse matrix multiplication, to produce a reduced representation of the data model, from which an accurate clustering solution can be more easily derived.

4. *Kernel methods* apply functions to implicitly transform the data to a new, possibly high-dimensional space where non-linear relationships between data objects may be more readily identified.

5. *Ensemble clustering methods* involve generating a diverse collection of clusterings on the same data and subsequently combining these clusterings to produce a superior solution.

The first pair of categories represent well-established approaches that have been widely used in the literature for several decades, while the latter three represent alternative, modern approaches that have recently become popular. In this chapter, we examine all five categories in detail, describing their respective advantages and limitations. In addition, we discuss a variety of issues that are relevant when applying clustering methods to text data, ranging from the generation of a suitable data model to problems pertaining to scalability.

2.2 Text Preprocessing

Before introducing the clustering algorithms themselves, it is necessary to examine the preliminary phase of the cluster analysis process, which is concerned with producing a machine-interpretable representation from a document collection. As we shall see, the nature of the preprocessing techniques that are applied in this context can greatly influence the outcome of any subsequent clustering procedure.

2.2.1 Document Parsing

Given a new collection of unstructured text documents $\{d_1, \ldots, d_n\}$, the first task is to apply a once-off parsing process, where the set of raw documents is transformed into a data model which can be subsequently analysed by a machine learning algorithm. This task generally involves applying a chain of procedures to each document:

Tokenisation: This initial procedure transforms the content of a document into a sequence of terms, representing words or phrases, which will subsequently be used to characterise the document. In some cases it may be useful to preserve information regarding the relative ordering of terms, depending upon the choice of data model.

Stemming: To reduce the number of unique terms, it is generally useful to stem terms to their roots. For the English language, the standard procedure is to apply the Porter suffix stripping algorithm (Porter, 1980) to eliminate common morphological and inflectional endings (*e.g.* "programming"$\rightarrow$"program"). A variety of open and commercial techniques are available for stemming documents written in other languages.

Stop-word removal: It will often be the case that it is not necessary to include all terms from the original corpus vocabulary in the data model. Notably, in text mining tasks it is extremely common to remove basic functional words (*e.g.* "the", "if") which occur so frequently in documents that they have no discriminating power and can be considered to be noise (Rijsbergen, 1975).

2.2.2 Vector Space Model

The choice of a suitable model to express document-term relations is fundamental to the success of text mining tasks. The *vector space model* (Salton *et al.*, 1975), also referred to as the "bag of words" approach, has been the dominant method for representing documents in information retrieval, text classification and clustering problems. In this model, each document d_j is represented by a vector $x_j = \{f_1, \ldots, f_m\}$ in a m-dimensional term space, where m is the total number of unique terms across all documents in the corpus and f_i indicates the frequency of occurrence of the i-th term in d_j. Once an entire corpus of n documents has been transformed to a corresponding set of feature vectors $\{x_1, \ldots, x_n\}$, the documents can be clustered based on the similarities or dissimilarities between vectors. For convenience, the complete model is usually stored as a single *term-document matrix*

$$\mathbf{A} = [x_1 \; x_2 \; \ldots \; x_n] \in \mathbb{R}^{m \times n}$$

where the entry A_{ij} indicates the frequency of the i-th term in the document vector x_j.

When employing this model, a significant amount of information about the original documents is lost, including information describing the ordering and context of terms. Some authors have suggested adapting the bag of words approach to use features constructed from phrases or n-grams, which consist of sequences of n consecutive characters extracted from text strings. However, the former can greatly exacerbate problems related to sparsity, while the latter reduces the interpretability of the model. Other radically different approaches for modelling text have been proposed (*e.g.* Park *et al.*, 2004), but none of these have been widely adopted. While the lack of spatial information and the sparse, high-dimensional nature of the term space do represent significant drawbacks, the vector space model remains the most popular choice due to its simplicity and the fact that traditional clustering algorithms working on numerical feature vectors can be directly applied to this representation. The underlying assumption here is that, if a pair of vectors are close to one another in the high-dimensional term space, the corresponding documents will share similar concepts.

2.2.3 Term Weighting

When working with text data, it is common to employ an additional preprocessing step, which involves replacing the original raw term frequency values with weighted frequencies calculated according to some normalisation function. This function is typically composed of two components: *term frequency* (tf) and *inverse document frequency* (idf). The former has the effect of increasing the impact of terms that occur frequently in a single document, while the latter seeks to reduce the influence of terms occurring in many documents, which may not be helpful in discriminating between the underlying classes in the data.

A wide variety of *tf-idf* weighting schemes were formally described by Salton & Buckley (1987). Among these, the *ltc* variant is most frequently employed in the document clustering literature, which applies logarithmic normalisation to both term frequency and document frequency values. Formally, the weighted frequency value for the i-th term in the document d_j is defined as:

$$tfidf(i,j) = ltf(i,j) \cdot \left(\log \frac{n}{df_i}\right) \quad \text{where} \quad ltf(i,j) = \begin{cases} 1 + \log f_i & \text{if } f_i > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

where f_i is the number of occurrences of the i-th term in d_j and df_i is the total number of documents in the dataset which contain that term.

2.2.4 Similarity Measures

The choice of a suitable measure for quantifying the strength of association between pairs of documents is essential to the successful discovery of accurate groupings. For some clustering algorithms, a single pairwise similarity or dissimilarity matrix can be constructed as part of the preprocessing phase to avoid unnecessary computations during the subsequent clustering process. In other cases, the measure will be employed to pairs of documents during the execution of the algorithm itself. While a wide range of techniques for assessing similarity have been proposed in different application fields, we consider here three metrics that are interesting from the perspective of researchers working with text datasets.

Euclidean distance

The most popular measure for working with real-valued feature vectors is *Euclidean distance*, which involves computing the squared L2 norm between two vectors:

$$ed(x_i, x_j) = ||x_i - x_j|| = \left(\sum_{l=1}^{m} (x_{i,l} - x_{j,l})^2 \right)^{\frac{1}{2}} \tag{2.2}$$

This metric has been widely used in machine learning problems due to its intuitive approach to capturing the concept of distance and its applicability to many different types of data. For similarity-based algorithms, a common approach for converting Euclidean distances to similarity values is to use the exponential function (Ghosh, 2003):

$$sed(x_i, x_j) = e^{-||x_i - x_j||^2} \tag{2.3}$$

A generalisation of Eqn. 2.2, often referred to as the Minkowski distance measure, employs the L_p norm, where the value of the exponent p is a user-defined parameter.

Cosine similarity

Although Euclidean distance is useful in many domains, it has frequently been shown that it does not work well for high-dimensional data, due to the importance it places on absent values (Strehl, 2002). As an alternative, the most commonly used method to compute the similarity between two documents when employing the vector-space model has been to measure the cosine of the angle between their corresponding vectors (Salton & McGill, 1983):

$$cos(x_i, x_j) = \frac{\langle x_i, x_j \rangle}{||x_i|| \cdot ||x_j||} \tag{2.4}$$

Note that the numerator is the dot product between the two vectors, while $||x_i||$ in the denominator indicates the length of x_i. This normalisation ensures that pairs of documents which differ in length, but have term frequencies in equal proportions, are considered to be identical. In that case, the value of Eqn. 2.4 is one, while a value of zero indicates that a pair of documents do not share any common terms. For algorithms that make use of dissimilarity values, *cosine distance* may be computed by simply using:

$$dcos(x_i, x_j) = 1.0 - cos(x_i, x_j) \tag{2.5}$$

Extended Jaccard similarity

The Jaccard coefficient (Jaccard, 1912), which assesses the level of agreement between two sets based on the ratio of union and intersection between the pair, has often been used as a measure of similarity on data represented by binary features. Strehl *et al.* (2000) recast this coefficient for use on non-negative, real-valued data, where the similarity between two feature vectors is computed by:

$$ej(x_i, x_j) = \frac{\langle x_i, x_j \rangle}{||x_i||^2 + ||x_j||^2 - \langle x_i, x_j \rangle} \tag{2.6}$$

As with the cosine measure, Eqn. 2.6 gives more emphasis to the presence of a term than to its absence, making it potentially useful when working with a sparse vector space model. However, it is sensitive to document length and has not been widely used in the context of document clustering.

2.3 Partitional Clustering Methods

We now consider the problem of clustering itself, and begin by examining popular partitional clustering methods, which attempt to directly decompose a dataset into a flat partition consisting of a fixed number of clusters, denoted $\mathcal{C} = \{C_1, \ldots, C_k\}$. These methods generally seek to produce a local approximation to a global objective function, which is identified by iteratively refining an initial solution.

2.3.1 Standard k-Means Algorithm

Standard k-means is the most widely used partitional clustering algorithm. It employs an iterative relocation scheme to produce a k-way hard clustering that locally minimises the distortion between the data objects and a set of k cluster representatives. Each representative, referred to as a *centroid*, is computed as the mean vector of all objects assigned to a given cluster. In the classical version of the algorithm, distortion is measured using

```
1. Create an arbitrary initial clustering with centroids $\{\mu_1, \dots, \mu_k\}$.
2. For each object $x_i \in \mathcal{X}$:
    1. Compute $||x_i - \mu_c||$ for $1 \leq c \leq k$.
    2. Reassign $x_i$ to the cluster corresponding to the nearest centroid.
3. Update cluster centroids.
4. Repeat from Step 2 until a termination criterion is satisfied.
```

Figure 2.1: Standard batch k-means algorithm.

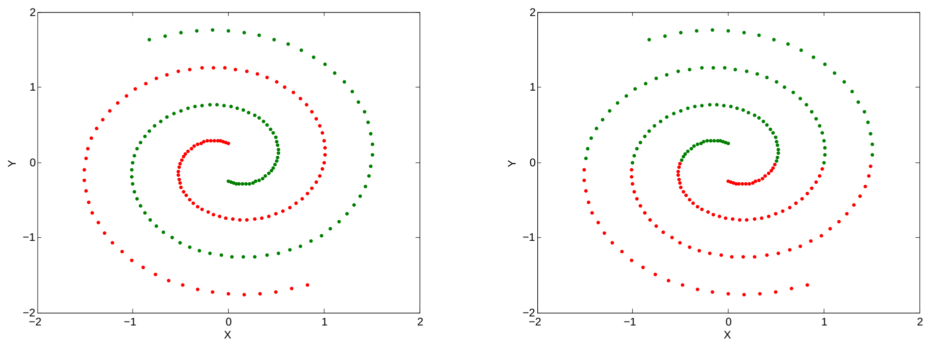
Euclidean distance, so that the goal of the clustering process becomes the minimisation of the *sum-of-squared error* (SSE) between the objects and cluster centroids $\{\mu_1, \dots, \mu_k\}$:

$$SSE(\mathcal{C}) = \sum_{c=1}^{k} \sum_{x_i \in C_c} ||x_i - \mu_c||^2 \quad \text{where} \quad \mu_c = \frac{\sum_{x_i \in C_c} x_i}{|C_c|} \tag{2.7}$$

While many variations of the basic algorithm exist, the most frequently applied version for offline clustering is the *batch k-means* algorithm, generally attributed to Forgy (1965), which involves a two-step process as shown in Figure 2.1. In the first step, each object is reassigned to the closest cluster centroid. Once all objects have been processed, the centroid vectors are updated to reflect the new cluster assignments. The iterative refinement process is repeated until a given termination criterion is satisfied. Typically this occurs when the assignment of objects to clusters no longer changes from one iteration to another. Alternatively, the procedure may be terminated if the change in the evaluation of Eqn. 2.7 between two successive iterations is less than a user-defined threshold.

Limitations

The SSE function (2.7) implicitly assumes that the clusters approximate a mixture of Gaussians, such that each cluster is spherical in shape and data objects are largely concentrated near its centroid. Consequently, k-means will often fail to identify a useful partition in cases where the clusters are non-spherical or differ significantly in size. As an example, we consider the synthetic *2-spirals* dataset (Jain & Fred, 2002a), which consists of two elongated inter-woven clusters as shown in Figure 2.2(a). Due to the non-convexity of these structures, k-means will tend to simply bisect them, resulting in a poor clustering such as that given in Figure 2.2(b).



(a) Natural classes (b) K-means clustering

Figure 2.2: Example where k-means fails to produce an accurate clustering when applied to the *2-spirals* dataset, due to the presence of complex cluster shapes.

The traditional objective for k-means can also give undue influence to outlying objects. Their effect in centroid construction can lead to vectors that are not representative of the underlying groups in the data, resulting in highly skewed clusters. Some authors have proposed the introduction of an "outlier cluster", which is used to hold objects that do not fit well in any other cluster (Fischer & Buhmann, 2003). Others have suggested repeatedly applying the clustering algorithm and removing poorly clustered data after each run (Hautamäki *et al.*, 2005). However, both approaches require the introduction of an arbitrary threshold to determine whether an object is far enough from its current centroid to be deemed an outlier. Another problem occurs when the iterative refinement process results in the formation of empty clusters. A common strategy to deal with this is to assign the most outlying object (*i.e.* furthest from its current centroid) to the empty cluster. However, if the problem persists, it is more likely that the fault may lie with the choice of clustering model, such as the use of an unsuitable value for k.

A well-documented issue relating to partitional algorithms in general is their sensitivity to the choice of initial clusters (Creator *et al.*, 1984). Since algorithms such as k-means can easily become trapped at a local minimum, the starting point for the clustering process can greatly affect the accuracy of the final solution. When the starting point is chosen using a strategy that includes a stochastic element, this can often lead to the generation of a variety of significantly different solutions over multiple runs on the same dataset, making it difficult for a user to identify a single, definitive grouping of the data.

2.3.2 Related Algorithms

Since the k-means algorithm was originally introduced, numerous variations and extensions have been proposed in the literature. We now summarise several notable variants that have been proposed to address the short-comings of k-means discussed previously.

Generalised k-means

Rather than using Euclidean distance, the k-means algorithm may be extended to work with other cost functions that encode some notion of object-representative similarity. As noted previously, the cosine measure represents a more suitable metric for working with high-dimensional text data. Using this measure, the clustering objective becomes to maximise the cohesion of the k clusters, which is equivalent to the sum of document-centroid similarities (Zhao & Karypis, 2002):

$$Coh(\mathcal{C}) = \sum_{c=1}^{k} \sum_{x_i \in C_c} cos(x_i, \mu_c) \tag{2.8}$$

Partitioning Around Medoids

To reduce the influence of outliers, many authors have suggested alternatives ways of forming cluster representatives. A well-known example, the Partitioning Around Medoids (PAM) algorithm (Rousseeuw, 1987), uses the most centrally located data object or *medoid* of each cluster as a representative. After selecting k arbitrary initial medoids, the algorithm tries to find an optimal set of representatives by repeatedly attempting to swap pairs of objects, where one object is currently a medoid and one is not. Although this approach can be more robust in some circumstances, the cost of required for each iteration is $O(k(n-k)^2)$. This additional computational expense renders it unsuitable for large values of n or k. Consequently, approximation methods have been proposed, such as CLARA (Ng & Han, 1994), which operates on a small sample of the original data.

Spherical k-means

Dhillon & Modha (2001) suggested a slight variation of batch k-means suitable for application to text data, where both feature vectors and centroids are L2-normalised. The vectors may then be viewed as points lying on a high-dimensional unit sphere, while the normalised centroids, referred to as "concept vectors", can be viewed as representing the topics present in each cluster. The algorithm seeks to maximise document-centroid similarities based on dot products, which gives rise to the clustering objective:

$$\Theta(\mathcal{C}) = \sum_{c=1}^{k} \sum_{x_i \in C_c} {x_i}^{\mathsf{T}} {\mu_c}' \quad \text{where} \quad {\mu_c}' = \frac{\sum_{x_i \in C_c} x_i}{||\sum_{x_i \in C_c} x_i||} \quad \text{and} \quad ||x_i|| = 1 \tag{2.9}$$

After generating a clustering solution, the terms corresponding to the highest values in the concept vectors can be used to provide interpretable cluster labels.

First variation k-means

Dhillon *et al.* (2002a) observed that centroid-based algorithms such as spherical k-means often become quickly trapped at a poor local solution. To increase clustering accuracy, the authors proposed incorporating a local search technique into the standard batch k-means algorithm. Specifically, this involves examining all "first variations", which refers to the set of possible partitions that may be produced from an existing partition by re-assigning a single object to an alternative cluster. The variation leading to the largest improvement in the clustering objective function is subsequently chosen as the partition for the next iteration. This concept can be expanded to consider the search for a chain of first variations that lead to the greatest overall increase in the objective function. For practical purposes, a "ping-pong" strategy can be employing that alternates between spherical k-means and the first variation approach. The use of the latter allows the clustering procedure to move away from poor local solution, leading to a better final clustering. However, the application of multiple first variation iterations can be very computationally expensive when working with large datasets (Zhong, 2005).

Fuzzy c-means

Dunn (1974a) proposed a generalisation of standard k-means, the Fuzzy c-means (FCM) algorithm, which allows objects to belong to different clusters to certain degrees as expressed by probabilistic weights. These weights may be represented in the form of a $n \times k$ matrix $\mathbf{V}$, where $V_{ij} \in [0, 1]$ denotes the degree of membership of the object x_i in cluster C_j, and $\sum_j V_{ij} = 1$. Once again, the task of clustering is to minimise the distortion between objects and centroids, which is now measured by the fuzzy criterion function

$$F(\mathcal{C}, \mathbf{V}) = \sum_{i=1}^{n} \sum_{j=1}^{k} {V_{ij}}^{m} \, ||x_i - \mu_j||^2 \tag{2.10}$$

where the exponent $m > 1$ controls the fuzziness of object memberships. In this algorithm, centroids are computed using:

$$\mu_j = \frac{\sum_{i=1}^{n} {V_{ij}}^m x_i}{\sum_{i=1}^{n} {V_{ij}}^m} \tag{2.11}$$

As noted previously, Euclidean distortion is often inappropriate for text data. To address this problem in the context of fuzzy clustering, Kummamuru *et al.* (2003) proposed a fuzzy version of the spherical k-means algorithm, which seeks to maximise intra-cluster cosine similarities on unit length document vectors.

EM clustering

Another well-known soft partitional clustering technique is the Expectation Maximisation (EM) algorithm (Dempster *et al.*, 1977). Unlike the other techniques described here, this algorithm takes a model-based approach to identifying groups in data. Formally, EM clustering is based on the assumption that the data objects are generated using a model θ which consists of a mixture of k underlying probability distributions $\{\theta_1, \ldots, \theta_k\}$. The task of clustering can then be viewed as the problem of determining the most likely parameters for the model, where each component in the mixture represents a cluster. The likelihood of an object x_i is given by:

$$P(x_i|\theta) = \sum_{c=1}^{k} P(C_c)P(x_i|C_c)$$

In the standard formulation of the algorithm, the k distributions are assumed to be Gaussians, so that the problem becomes the approximation of the mean and covariance of each component. In practice, the algorithm begins with an initial estimate for the model parameters and subsequently applies an iterative optimisation approach that alternates between two steps: firstly identify the expected value of the log likelihood with respect to the current parameter estimates, then find new parameter values to maximise this likelihood. Once the algorithm has converged to a local solution, each data object is probabilistically assigned to each cluster based on the estimated distributions. As with standard k-means, the choice of initial clusters can have a considerable effect on the accuracy of the final solution. EM clustering has been widely used in many applications, including document clustering (Liu *et al.*, 2002).

2.3.3 Initialisation of Partitional Algorithms

As noted previously, the choice of a suitable initialisation strategy for a given clustering task is highly important when employing k-means. A number of different strategies have been proposed in the literature, the most common of which are summarised here.

Random initialisation: Most frequently, initialisation is performed by simply randomly dividing the data into k disjoint subsets. However, this can often lead to highly inconsistent results over many trials.

Forgy initialisation: In this approach, k data objects are selected at random as seeds, and the remaining objects are assigned to the nearest seed (Forgy, 1965).

MacQueen initialisation: A similar strategy was described by MacQueen (1967), which also involves randomly selecting k objects to act as initial centroids. Each remaining object is then assigned to the nearest centroid. However, in this case the centroid vectors are recalculated after each reassignment.

Furthest-first initialisation: This popular strategy is based on the assertion that, since clusters represent distinct groups in a feature space, a set of cluster seeds should be chosen to be as well-separated as possible. To achieve this, an initial seed is chosen at random and each additional seed is determined by finding the unassigned object which is furthest from the previously selected seeds (Hochbaum & Shmoys, 1985). The process continues until k seeds have been selected. Variations of this approach have been proposed that involve selecting the first seed to be the mean of the dataset, the object closest to the mean or the object whose feature vector has the maximum norm (Katsavounidis *et al.*, 1994).

Subset furthest-first initialisation: In general, the furthest-first method tends to be sensitive to the presence of outliers, since the selection of objects that are most distant will often result in the selection of seeds that are not representative of the true clusters. To address this, Turnbull & Elkan (2005) proposed randomly sampling a subset of the data to remove as many outliers as possible, and applying the furthest-first technique to the selected subset.

While many of the more complex initialisation strategies can lead to improvements on certain types of data, in other situations they can produce clusterings that are on average less accurate than those produced using simple random initialisation.

2.4 Hierarchical Clustering Methods

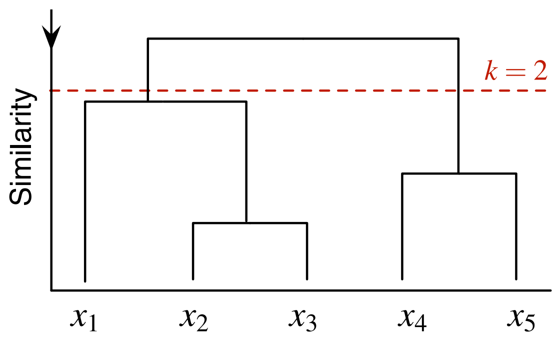
Instead of generating a flat partition of data, it may often be useful to construct a hierarchy of concepts by producing a set of nested clusters that may be arranged to form a tree structure. While partitional clustering methods have received more attention in recent literature, hierarchical clustering algorithms represent the traditional choice for performing document clustering, since text collections often contain broad themes that may be naturally sub-divided into more specific topics. Hierarchical algorithms are generally organised into two distinct categories:

Agglomerative: Begin with each object assigned to a singleton cluster. Apply a bottom-up strategy where, at each step, the most similar pair of clusters are merged.

Divisive: Begin with a single cluster containing all n objects. Apply a top-down strategy where, at each step, a chosen cluster is split into two sub-clusters.

In either case, the resulting hierarchy may be presented visually using a tree-like structure referred to as a *dendrogram*, which contains nodes for each cluster constructed by the clustering algorithm, together with cluster relations illustrating the merge or split operations that were performed during the clustering process. Figure 2.3 provides a simple example of an agglomerative clustering process applied to a set of five data objects, together with the corresponding cluster assignments. It is worth noting that, as each merge operation is performed, the similarity between the chosen pair of cluster decreases.

Unlike the requirement in most partitional algorithms to specify a value for the number of clusters k in advance, hierarchical algorithms support the construction of a tree from



$$C = \{\{x_1, x_2, x_3, x_4, x_5\}\}$$
$$C = \{\{x_1, x_2, x_3\}, \{x_4, x_5\}\}$$
$$C = \{\{x_1\}, \{x_2, x_3\}, \{x_4, x_5\}\}$$
$$C = \{\{x_1\}, \{x_2, x_3\}, \{x_4\}, \{x_5\}\}$$
$$C = \{\{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}, \{x_5\}\}$$

Figure 2.3: Example dendrogram representing an agglomerative clustering of five data objects, together with the corresponding cluster memberships.

which a user may manually select k by examining the resulting dendrogram and identifying an appropriate cut-off point (Milligan & Cooper, 1985). For instance, by cutting the tree in Figure 2.3 at the level indicated, we can derive a clustering of the data for $k = 2$ from the two leaf nodes at that level.

2.4.1 Agglomerative Algorithms

Agglomerative hierarchical clustering (AHC) involves the construction of a tree of clusters from the bottom upwards. A variety of agglomerative algorithms have been proposed, such as BIRCH (Zhang *et al.*, 1996) and CURE (Guha *et al.*, 1998), which are suitable for specific types of data. However, we focus on the standard formulation that has widely been used in document clustering (Voorhees, 1986), which proceeds as follows:

1. Assign each object to a singleton clusters.
2. Update the pairwise inter-cluster similarity matrix.
3. Identify and merge the most similar pair of clusters.
4. Repeat from Step 2 until a single cluster remains or a given termination criterion has been satisfied.

When an estimation for the number of clusters k is given in advance, the algorithm may be terminated when the required number of leaf nodes remain in the dendrogram.

A variety of *linkage* strategies exist for determining which pair of clusters should be merged from among all possible pairs. While these strategies are typically expressed in terms of distances, they may be easily adapted to use similarity values such as those produced by the cosine measure. Given a symmetric matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$, where S_{ij} denotes the similarity between a pair of objects x_i and x_j, the most popular linkage strategies for document clustering are defined as follows:

Single linkage: The most common strategy, also known as the nearest neighbour technique, defines the similarity between two clusters (C_a, C_b) as the maximum similarity between an object assigned to C_a and an object assigned to C_b:

$$sim(C_a, C_b) = \max_{x_i \in C_a, x_j \in C_b} S_{ij}$$

While this approach is widely used, it can often produce clusters of poor quality as it is subject to the phenomenon of "chaining", where singletons are repeatedly merged with an existing cluster, resulting in one large, elongated cluster with highly dissimilar objects at either end.

Complete linkage: The similarity between two clusters (C_a, C_b) is defined as the minimum similarity between an object assigned to C_a and an object assigned to C_b:

$$sim(C_a, C_b) = \min_{x_i \in C_a, x_j \in C_b} S_{ij}$$

This strategy tends to favour strongly compact, tightly coupled clusters and is often highly sensitive to the presence of outliers.

Average linkage: The similarity between a pair of clusters (C_a, C_b) is calculated as the mean similarity between objects assigned to C_a and objects assigned to C_b:

$$sim(C_a, C_b) = \frac{\sum_{x_i \in C_a} \sum_{x_j \in C_b} S_{ij}}{|C_a|\,|C_b|}$$

This strategy is often referred to as *unweighted pair group method using arithmetic averages* (UPGMA), since normalising by cluster size has the effect of giving equal weights to objects that are assigned to clusters of different sizes.

Min-max linkage: Ding & He (2002) suggest the use of a novel graph partitioning objective, the *min-max* criterion (see Section 2.6.1), as a means of assessing the similarity between two clusters:

$$s(C_a, C_b) = \sum_{x_i \in C_a} \sum_{x_j \in C_b} S_{ij} \quad \text{and} \quad sim(C_a, C_b) = \frac{s(C_a, C_b)}{s(C_a, C_a) s(C_b, C_b)}$$

In agglomerative clustering, this linkage strategy has the tendency of merging similar clusters with low self-similarities to produce larger, more balanced clusters, with high self-similarity.

Clearly, the choice of linkage strategy can significantly affect the structure of the clusters that are generated by AHC. As a consequence, the prior selection of a suitable strategy for a given dataset may represent a non-trivial parameter selection problem. In practice, a user may generate several hierarchies using different approaches, and manually inspect the results to choose the most appropriate solution.

Limitations

A substantial drawback of standard agglomerative algorithms is that poor decisions made early in the clustering process can greatly influence the accuracy of the final solution.

Without the use of a global objective function, many potential mergers at these stages may appear to be equally valid. Once a merging decision has been made, there exists no facility to rectify an erroneous choice at a later stage. On the contrary, the adverse effects of these decisions are often exaggerated as the clustering process continues. In addition to deficiencies in clustering accuracy, hierarchical clustering algorithms are generally considerably more computationally costly than their partitional counterparts, typically having time complexity $O(n^3)$.

2.4.2 Divisive Algorithms

In contrast to agglomerative methods, divisive hierarchical clustering involves building a cluster tree from the root node downwards.

1. Assign all data objects to a single cluster.
2. Select a cluster to split.
3. Replace the selected cluster with two new sub-clusters.
4. Repeat from Step 2 until k leaf clusters have been generated or a given termination criterion has been satisfied.

Several authors have empirically shown divisive algorithms to be superior to agglomerative techniques on text data (Ding & He, 2002). In addition, these algorithms are often less time consuming than traditional bottom-up clustering. However, in general, they have been employed less frequently due to the non-trivial problems of selecting a cluster to split and finding the optimal sub-division of the chosen cluster. We now describe two representative divisive algorithms that have been primarily employed for document clustering.

Bisecting k-means

As a representative example of divisive clustering, we consider the algorithm proposed by Steinbach *et al.* (2000) for use on text data, which combines aspects of hierarchical and partitional clustering. Initially, all documents are assigned to a single root cluster. The algorithm involves repeatedly selecting an existing cluster and splitting the cluster into two sub-clustering using the generalised k-means algorithm with cosine similarity. The process is repeated until k clusters have been obtained. To split a cluster, a fixed number of randomly-initialised bisections τ may be performed, from which the best candidate

1. Assign all n objects to a single cluster.
2. Select a cluster C_c to split according to a chosen splitting criterion.
3. Generate τ 2-way partitions of the cluster C_c using randomly initialised k-means.
4. Replace C_c with the best pair of clusters as determined by a given clustering criterion.
5. Repeat from Step 2 until k leaf clusters have been generated.

Figure 2.4: Bisecting k-means (BKM) algorithm.

is selected. This choice is determined by a cluster evaluation criterion, such as mean document-centroid cosine similarity:

$$Cen(C_c) = \frac{\sum_{x_i \in C_c} cos(x_i, \mu_c)}{|C_c|} \tag{2.12}$$

A larger value for τ renders the algorithm less sensitive to the choice of initial clusters than the partitional algorithms described in Section 2.3, although it does increase the computational cost of applying the algorithm. A summary of the complete procedure is given in Figure 2.4.

Several strategies have been proposed to identify the most appropriate cluster to split. A naïve approach is to divide the largest cluster into two sub-clusters at each stage (Steinbach *et al.*, 2000). However, this may be inappropriate when working with text corpora, which frequently contain "unbalanced" clusters that differ in their relative proportions. An alternative strategy is to split the cluster with maximal distortion or minimal cohesion, as measured by Eqn. 2.8. For instance, Ding & He (2002) suggested selecting the cluster with the minimum mean intra-cluster pairwise similarity, computed by:

$$Intra(C_c) = \frac{\sum_{x_i, x_j \in C_c} cos(x_i, x_j)}{|C_c|^2} \tag{2.13}$$

Given k potential candidates for splitting, Zhao & Karypis (2002) proposed evaluating all possible candidates and selecting the split that leads to the best subsequent partition containing $k+1$ clusters. However, this approach requires significantly more computational time than the standard formulation of the algorithm.

Principal Direction Divisive Partitioning

The well-known Principal Direction Divisive Partitioning (PDDP) algorithm (Boley, 1998) provides an efficient means of producing a hierarchy of clusters using a non-iterative ap-

1. Assign all n objects to a single cluster.
2. Select a cluster C_c to split, such that the cluster has the highest level of intra-cluster scatter.
3. Calculate the principal direction u_c of the matrix representation of the cluster C_c, and its centroid μ_c.
4. Partition C_c into two sub-clusters (C_a, C_b) such that, for each original $x_i \in C_c$:

$$x_i \in \begin{cases} C_a & \text{if } fc(x_i) \leq 0 \\ C_b & \text{otherwise.} \end{cases} \qquad \text{where the projection} \quad fc(x_i) = {u_c}^{\mathsf{T}}(x_i - \mu_c)$$

5. Repeat from Step 2 until k leaf clusters have been generated.

Figure 2.5: Principal Direction Divisive Partitioning (PDDP) algorithm.

proach based on the concepts employed in Principal Component Analysis (PCA) (see Section 2.6 for more details regarding PCA). The *principal direction* of a cluster of documents C_c, denoted by u_c, is defined as the leading eigenvector of the covariance matrix of C_c. In practice, given a sparse term-document matrix $\mathbf{A}$ representing an entire dataset, it proves more efficient to compute the leading left singular vector of the normalised matrix constructed according to:

$$\mathbf{M}_c = \mathbf{A}_c - \mu_c \mathbf{1}^{\mathsf{T}}$$

where $\mathbf{A}_c$ is a sub-matrix of $\mathbf{A}$ with columns corresponding to the document vectors in C_c, μ_c is the cluster centroid, and $\mathbf{1} = [1, \ldots, 1]$ is a n-dimensional vector of ones. Note that each document vector is assumed to be scaled to unit length.

The algorithm commences by splitting the entire document collection into two clusters. This is done by projecting each document vector onto the principal direction of $\mathbf{A}$ and examining the resulting values. A binary tree of clusters is subsequently constructed by recursively splitting clusters in the same manner until the desired number of clusters is obtained. At each stage, the cluster for splitting is chosen to be the candidate C_c having the highest intra-cluster scatter, as measured by the Frobenius norm $||\mathbf{M}_c||_{\mathsf{F}}^2$. A summary of the complete algorithm is provided in Figure 2.5.

PDDP has been widely used for document clustering, although recent work has shown that the algorithm can often perform poorly in the presence of overlapping clusters (Kruengkrai *et al.*, 2004) and that clustering on a larger number of singular vectors can often lead to superior results (Shi & Malik, 2000). As with bisecting k-means, the selection of an appropriate cluster for splitting can significantly affect the quality of the final clustering solution.

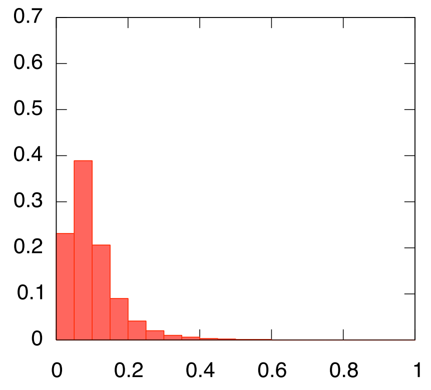
2.5 Unsupervised Dimension Reduction

2.5.1 Motivation

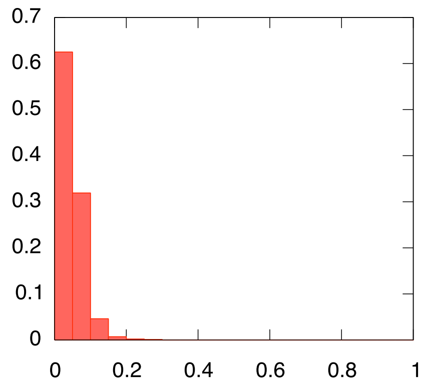
Like many machine learning methods, clustering algorithms are susceptible to the "curse of dimensionality" (Bellman, 1961), as additional information in the form of a larger feature set does not necessarily improve their ability to accurately partition data. Rather, algorithm performance often deteriorates significantly in the presence of many irrelevant or redundant features. As a result, classical clustering techniques working on document collections are faced with major obstacles due to the high-dimensional nature of this type of data. Some of these problems are due to lexical ambiguities which are endemic in natural languages, while others arise from the choice of representation or algorithm used in the clustering process. The most commonly observed problems are:

1. *Sparsity.* As discussed in Section 2.2.2, documents in text corpora are generally represented by large vectors, where each entry corresponds to a unique feature. While a typical feature vector will often contain 10,000 or more entries, for real-world corpora the vast majority of these values will be zero (typically 95-99%) since most terms will occur in very few documents. The inherent sparsity of documents when represented in a high-dimensional space can make it difficult for an algorithm to find any structure in the data. In particular, as the number of dimensions increases, document vectors tend to become equally similar to one another, thereby impairing an algorithm's ability to correctly discriminate between documents that are truly related and those that have little or no conceptual relation.

   To illustrate this, Figure 2.6 provides histograms for intra-class and inter-class pair-wise similarity values for the *bbc* dataset (see Section 5.2.1 for details regarding this corpus). The vector space model for this corpus contains 5570 unique dimensions after preprocessing. Due to the sparsity of the representation, the cosine similarities between the majority of pairs of documents are very low, even for pairs that belong to the same natural class, thus making the successful identification of the classes difficult.



(a) Intra-class similarities



(b) Inter-class similarities

Figure 2.6: Histograms for intra-class and inter-class pairwise cosine similarity values generated on the *bbc* corpus, which illustrate the problem of sparsity in text data.

2. *Synonymy.* The problem of sparsity can become significantly worse when using the vector space model, due to the assumption that terms are independent of one another. In reality this is rarely the case, as it is common for natural languages to use many different words to refer to identical or closely related concepts (*e.g.* 'residence' and 'abode'). Some authors have suggested the use of synonym lists from a thesaurus such as the WordNet[1] database to provide external information regarding relations between terms (Sedding & Kazakov, 2004). However, these resources are usually limited to a general-purpose vocabulary and may have little effect when working with specialised document collections, such as biomedical literature, where the majority of the discriminating terms will not be present in the thesaurus. The cost of constructing an appropriate list of synonyms for a specific domain may out-weigh the benefits of its use.

3. *Polysemy and Homonymy.* A more complex problem occurs when a single word has either multiple related meanings (polysemy) or has a number of different meanings that are completely unrelated (homonymy). For certain languages, the use of context information is often crucial in distinguishing between the various senses of a word. While this occurs less frequently in the English language, both polysemy (*e.g.* 'review' as a noun or as a verb) and homonymy (*e.g.* 'pupil' referring to a school student or a part of the eye) will still present significant problems for text mining algorithms. Since the standard vector space model does not retain context infor-

[1]`http://wordnet.princeton.edu/`

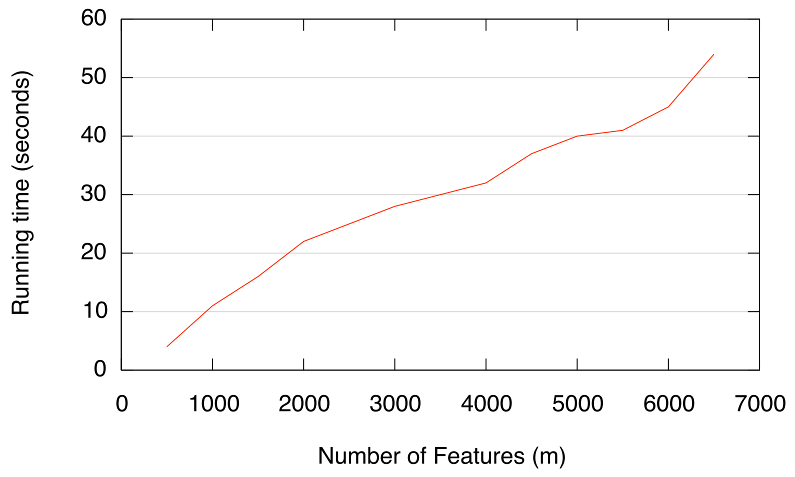


Figure 2.7: Mean running time (in seconds) for the k-means algorithm when applied to the *bbc* dataset. Observe that the computational cost increases significantly as additional features are added to the data model.

mation, algorithms that operate directly on a term-document matrix will often be unable to disambiguate word senses.

4. *Scalability.* The computational cost of many popular text mining procedures is largely dependent upon the number of dimensions m in the data. To illustrate this, Figure 2.7 shows the mean running time in seconds for a single execution of the standard k-means algorithm on the *bbc* dataset. As the number of features used to represent the documents increases, the time required also significantly increases. This effect is magnified when employing more computationally complex algorithms. While sparse matrix representations may be used to significantly increase algorithm efficiency, the level of improvement is highly dependent upon the sparsity of a dataset's term-document matrix. In addition, the overhead required to store the matrix can sometimes be prohibitive, so that for larger datasets it may become necessary to load only a small segment of the full matrix into memory at any one time.

To address these issues, *dimension reduction* techniques have been widely employed in machine learning tasks to reduce the number of features used to represent data. In cluster analysis, they commonly constitute part of a preprocessing phase applied prior to running a clustering algorithm. Potential benefits of reducing the dimensionality of text data include:

1. *Improved performance.* Removing redundant or noisy terms may provide a clearer picture of the true relations between documents in a corpus, leading to higher clustering accuracy. By compensating for linguistic problems such as synonymy and polysemy, we can also discover hidden semantic relations in the data that are not apparent from examining the original term frequency values.

2. *Economy of representation.* Dimensionality reduction can be viewed as a compression procedure, providing a more compact representation of the data. This can greatly improve the scalability of computationally complex clustering algorithms by reducing running time and storage requirements.

3. *Visualisation.* The output of a clustering algorithm can often be impossible to visualise due to the inherent problem of presenting high-dimensional spaces to humans. To address this, dimension reduction techniques have been widely used to aid visualisation in data exploration applications. In practice, this involves selecting 2–3 relevant dimensions, or projecting the original data to a lower 2-dimensional or 3-dimensional space, which can often help to illustrate the spatial relations between documents and cluster structures.

We now outline two general approaches used in many machine learning problems to reduce the dimensionality of data.

2.5.2 Feature Selection

Feature selection is concerned with locating the "best" minimum subset of the original dimensions. By eliminating unnecessary features, the size of the model used to represent the data can be significantly reduced. Feature selection has been widely employed in supervised learning tasks, such text classification (see Dash & Liu, 1997). These methods typically incorporate a search strategy for exploring the space of feature subsets, and can generally be divided into two broad categories based on the approach used to evaluate subsets (John *et al.*, 1994): *wrapper* schemes make use of the learning algorithm itself to choose a set of relevant features, while *filter* schemes attempt to remove features prior to applying the algorithm. The popularity of these techniques has motivated the development of analogous methods for unsupervised learning, including wrappers around partitional clustering algorithms (Dy & Brodley, 2000) and information theoretic filter strategies (Dash & Liu, 2000). However, the absence of class information or any definitive subset evaluation criterion can significantly limit the usefulness of these techniques. In addition, the application of a costly optimisation procedure may often be prohibitive for large, very high-dimensional datasets such as text collections. For example, although the filter approach proposed by Dash & Liu (2000) is based on a heuristic search procedure, this process requires time $O(mn^2)$, which will impractical for all but the smallest corpora. Consequently, the use of complex subset search selection has been very limited for document clustering tasks.

Simpler, more efficient ranking strategies have been more frequently applied to improve the efficiency of text mining algorithms. These avoid the need for computationally expensive search procedures by assigning weights to individual terms indicating their relevance, and subsequently selecting from a ranked list those terms whose weight is above a certain threshold. In classification tasks, popular term ranking criteria, such as *information gain* (Quinlan, 1993) and the χ^2 statistic (Yang & Pedersen, 1997), can produce weights that discriminate between a fixed number of categories in a training set. For clustering, the lack of class information requires that evaluations are carried out using less sophisticated measures, which are often based upon the *tf-idf* weighting functions described in Section 2.2.3. For instance, Yang & Pedersen (1997) suggested simply weighting features inversely based on their *document frequency* values, while Tang *et al.* (2005) computed the *mean tf-idf* value for each term across all documents in a corpus under the assumption that terms with a higher average value will have more discriminating power. Several authors have employed *term variance quality* (Dhillon *et al.*, 2002b), which rewards features that exhibit high variance in frequency across different documents. Given a term-document matrix $\mathbf{A}$ containing raw frequency values, this criterion computes a weight for the i-th term using the expression:

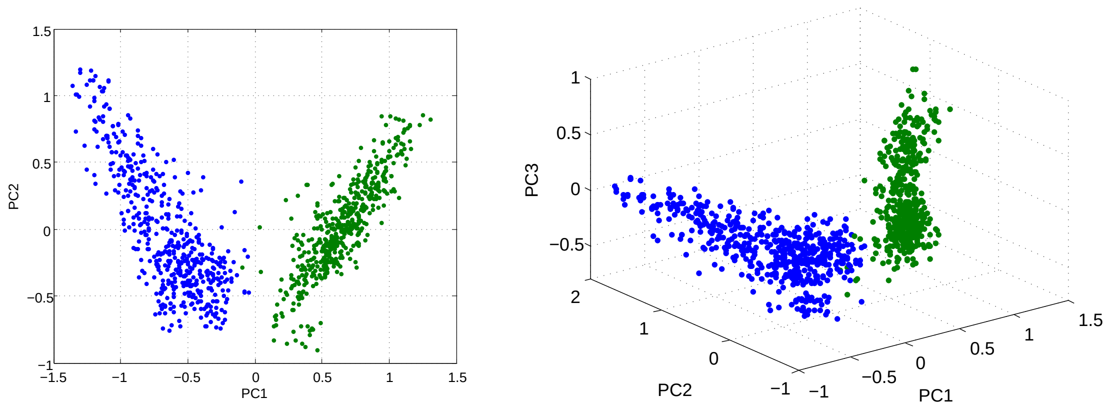
$$w_i = \sum_{j=1}^{n} {A_{ij}}^2 - \frac{1}{n}\left[\sum_{j=1}^{n} A_{ij}\right]^2 \tag{2.14}$$

Empirical research has shown that, without the provision of information to discriminate between terms and classes, unsupervised techniques will often perform poorly when terms a removed beyond a certain point (Liu *et al.*, 2003a). In general, the performance of ranking strategies is often highly dependent on a user selecting an appropriate threshold or cut-off point below which terms are rejected.

2.5.3 Feature Extraction

Feature extraction methods take an alternative approach to dimension reduction by applying a linear or non-linear transformation to map the original data to a new, low-dimensional representation, while attempting to preserve as much information as possible about the structure of the original data. The most popular unsupervised approach, *Principal Component Analysis* (PCA) (Pearson, 1901), is a linear technique that projects data onto a reduced set of orthogonal dimensions. Formally, given a covariance matrix $\mathbf{C}$, the new dimensions, referred to as principal components (PCs), are constructed from the eigenvectors of $\mathbf{C}$ such that the j-th PC is given by the eigenvector corresponding to the j-th largest eigenvalue of $\mathbf{C}$. By identifying the set of projection directions containing the largest variance and discarding the rest, PCA can reduce redundancy while minimising the reconstruction error. An example is provided in Figure 2.8, which shows 2-dimensional and 3-dimensional embeddings for a two class subset of the *bbc* corpus. By projecting the documents from the original high-dimensional vector space to the leading principal components, it is possible to identify two well-defined clusters corresponding to the topics.

PCA has proved useful in many domains, ranging from gene expression analysis (Raychaudhuri *et al.*, 2000) to image processing applications such as face recognition (Sirovich & Kirby, 1987). A related technique, *Latent Semantic Indexing* (LSI) (Deerwester *et al.*, 1990), has been widely employed for dimension reduction in information retrieval to uncover hidden patterns in text collections, which would otherwise be obscured by the linguistic problems described previously. Other well-known methods in this area include



(a) Projection onto top 2 leading PCs. (b) Projection onto top 3 leading PCs.

Figure 2.8: Visualisation of the ‘business’ and ‘sport’ topics from the *bbc* corpus, based on an embedding constructed from 2–3 leading principal components (PCs).

Independent Component Analysis (ICA) (Comon, 1994), which attempts to map the data to a set of statistically independent variables that are not necessarily orthogonal, and *Random Projection* (RP) (Bingham & Mannila, 2001), which applies an efficient linear transformation to produce a small set of approximately orthogonal dimensions. However, these latter approaches have not been frequently applied to text data.

One notable drawback of extraction techniques such as PCA and LSI is the loss of interpretability resulting from the fact that the newly formed dimensions may not necessarily have any physical meaning. From a knowledge discovery perspective, this may not be acceptable as it limits the degree to which a clustering solution can be explained, either with regard to the content of the clusters or to how the clusters were derived. We discuss this issue further in Section 6.2. In general, the computational and performance advantages of extraction-based reduction have made such methods far more prevalent than feature selection strategies in recent clustering literature. In the next two sections we provide detailed descriptions of two state-of-the-art extraction approaches that have been successfully applied to text data.

2.6 Spectral Clustering

Motivated by work in graph theory, unsupervised feature extraction methods have been developed that employ well-known techniques from linear algebra to analyse the spectral properties of a graph representing a dataset. In practice, this involves constructing a reduced dimensional space from the eigenvalue decomposition (EVD) of a matrix form of the graph. Existing clustering algorithms may subsequently be applied in the reduced space to uncover the underlying classes in the data. Spectral clustering methods have been widely used due to their efficiency and applicability in a variety of tasks, including image segmentation (Shi & Malik, 2000), gene expression analysis (Kluger *et al.*, 2003) and document clustering (Dhillon, 2001). In this section we examine the theoretical underpinnings of spectral clustering and consider several algorithms that may be relevant in reducing the dimensionality of text data.

2.6.1 Graph Partitioning

As noted previously, a common way of expressing the relations between pairs of data objects is to use a symmetric similarity or affinity matrix $\mathbf{S}$, where S_{ij} denotes the association between the objects x_i and x_j. The task of producing a disjoint clustering may then be modelled as a graph partitioning problem, where $\mathbf{S}$ becomes the adjacency matrix for a weighted undirected graph $G(\mathcal{V}, \mathcal{E})$. In this model, the set of vertices $\mathcal{V}$ represents the data objects and the set of edges $\mathcal{E}$ represents pairwise similarities between objects. Given this graph-theoretic formulation, we seek to find a k-way partition $\mathcal{C} = \{C_1, \ldots, C_k\}$ of the vertices of G that optimises a particular objective function or cost criterion. We now enumerate several criteria that have been commonly used in graph partitioning, which are also relevant to the theoretical motivations for spectral clustering.

Minimum cut: The simplest approach for evaluating the quality of a two-way partition, the *minimum cut* criterion, measures the weight of the edges crossing the partition. The optimisation of this objective may be viewed as the task of identifying a separator such that as few edges as possible must be removed from the graph to produce two disconnected sub-graphs (*i.e.* two disjoint clusters). Formally, we seek a bi-partition (C_1, C_2) of the graph vertices such that $C_1 \cup C_2 = \mathcal{V}$, which minimises the sum of the weights of edges connecting the two clusters, as denoted by:

$$s(C_1, C_2) = \sum_{i \in C_1, j \in C_2} S_{ij} \tag{2.15}$$

This expression shows that the weight of the cut is directly proportional to the number of edges that join the two sub-graphs. Consequently, the criterion favours small groups of isolated vertices. This makes it sensitive to outliers and often leads to highly unbalanced clusterings.

Ratio cut: To address the shortcomings of Eqn. 2.15, the *ratio cut* criterion (Pothen *et al.*, 1990) has frequently been employed for bi-partitioning. This measure seeks to minimise the edge cut, while balancing the sizes of the clusters in the partition.

$$Rcut(C_1, C_2) = \frac{s(C_1, C_2)}{|C_1|} + \frac{s(C_1, C_2)}{|C_2|} \tag{2.16}$$

Chan *et al.* (1994) generalised the two-way ratio cut metric to evaluate a partition $\mathcal{C}$ consisting of k clusters:

$$KRcut(\mathcal{C}) = \sum_{i=1}^{k} \frac{s(C_i, \mathcal{V} \backslash C_i)}{|C_i|} \tag{2.17}$$

Normalised cut: Shi & Malik (1997) proposed a more robust measure for assessing bi-partitions, the *normalised cut* criterion, which measures the degree of association between a cluster and the remaining vertices, relative to the total association within that cluster. This normalisation makes the criterion less sensitive to the presence of outlying objects. Formally, the criterion can be calculated using the expression:

$$Ncut(C_1, C_2) = \frac{s(C_1, C_2)}{s(C_1, \mathcal{V})} + \frac{s(C_1, C_2)}{s(C_2, \mathcal{V})} \tag{2.18}$$

Yu & Shi (2003) subsequently generalised this objective for k-way partitioning:

$$KNcut(\mathcal{C}) = \sum_{i=1}^{k} \frac{s(C_i, \mathcal{V} \backslash C_i)}{s(C_i, \mathcal{V})} \tag{2.19}$$

Min-max cut: Ding *et al.* (2001) proposed a graph partitioning criterion which aims to evaluate the common clustering objectives of maximising intra-cluster associations while simultaneously minimising inter-cluster associations. These objectives may be satisfied by minimising the expression:

$$MMcut(C_1, C_2) = \frac{s(C_1, C_2)}{s(C_1, C_1)} + \frac{s(C_1, C_2)}{s(C_2, C_2)} \tag{2.20}$$

This measure favours clusters of approximately equal size. It may be extended to k clusters by computing the sum of Eqn. 2.20 over all pairs of clusters, which can be simplified to:

$$KMMcut(\mathcal{C}) = \sum_{i=1}^{k} \frac{s(C_i, \mathcal{V} \backslash C_i)}{s(C_i, C_i)} \tag{2.21}$$

2.6.2 Spectral Bi-partitioning

Unfortunately, the problem of finding an optimal partition according to the criteria described in the previous section is NP-complete. While traditional techniques such as the Kernighan-Lin algorithm (Kernighan & Lin, 1970) have been used to produce local approximations, such methods often have drawbacks in terms of partition accuracy. Rather than directly attempting to optimise a given criterion, many authors have sought to transform the optimisation task into a generalised eigenvalue problem. Given a symmetric adjacency matrix $\mathbf{S}$, this involves computing its eigenvalue decomposition:

$$\mathbf{S} = \mathbf{V \Lambda V}^{\mathsf{T}}$$

where the diagonal entries of $\mathbf{\Lambda}$ represent the set of eigenvalues and the columns of $\mathbf{V}$ are a corresponding set of orthogonal eigenvectors. Unlike local partitioning methods, analysing the spectrum of $\mathbf{S}$ allows grouping to be performed based on global information describing the structure of the corresponding graph.

Early work in this area was performed by Donath & Hoffman (1973) and Fiedler (1973), who first suggested a connection between the problem of finding vertex separators for a graph and the eigenvalue decomposition of its corresponding *Laplacian matrix* $\mathbf{L} = \mathbf{D} - \mathbf{S}$, where $\mathbf{D}$ denotes a diagonal degree matrix such that $D_{ii} = \sum_{j=1}^{n} S_{ij}$. The Laplacian of a graph G containing n vertices is symmetric positive semi-definite, with non-negative eigenvalues $0 = \lambda_1 < \lambda_2 < \cdots \leq \lambda_n$ and corresponding eigenvectors $\{v_1, \ldots, v_n\}$. The most important common observation made by these authors was that the spectrum of a graph provides useful structural information that may indicate how best to partition its vertices. The use of spectral partitioning was popularised by the proposal of a formal technique by Pothen *et al.* (1990). Following the work of Hall (1970), a bi-partition (C_1, C_2) of G may be represented by a membership indicator vector $q = \{q_1, \ldots, q_n\}$ such that:

$$q_i = \begin{cases} +1 & \text{if } i \in C_1 \\ -1 & \text{if } i \in C_2 \end{cases}$$

If the adjacency matrix $\mathbf{S}$ has a block-diagonal structure (*i.e.* the rows can be reorganised by cluster membership to form a checker-board pattern), we can optimise the minimum cut (2.15) by finding a clustering that minimises the sum of the weights in the off-diagonal blocks. The problem can be formulated as the search for a vector q that minimises:

$$\arg\min_{q} \sum_{i,j=1}^{n} S_{ij}(q_i - q_j)^2 \quad \text{such that} \quad \sum_{i=1} q_i^2 = 1$$

This objective can also be expressed in quadratic form using the Laplacian $\mathbf{L}$:

$$\arg\min_{q} \; q^{\mathsf{T}} \mathbf{L} q$$

Rather than solving this as a complex combinatorial problem, a solution may be found by relaxing the requirement on q to contain discrete values, so that the assignment of each vertex is continuous, with membership weights taking real values in the range $[-1, 1]$. The partition approximating the minimal cut can then be found by examining the eigenvectors of $\mathbf{L}$. Specifically, the relaxed membership weights are calculated as the components of the eigenvector v_2 corresponding to the second smallest eigenvalue λ_2 of the Laplacian (*i.e.* the first non-trivial eigenvector), which is often referred to as the *Fiedler vector*.

A variety of justifications for partitioning based on v_2 are given in the literature. Fiedler (1973) showed the association between its corresponding eigenvalue λ_2 and the edge connectivity of a graph, while Pothen *et al.* (1990) demonstrated a relationship

Class	Doc	Content
business	b_1	record company profits
	b_2	company profits fall
	b_3	company takeover bid
football	f_1	world cup football
	f_2	champions league football
	f_3	world cup soccer

Table 2.1: Small dataset consisting of six short text fragments relating to two topics.

between the edge separator induced by v_2 and the *isoperimetric number* of a graph, which represents the value of the smallest possible edge cut over all candidate separators. The latter has motivated several popular spectral bi-partitioning algorithms. In these, the vertices are sorted according to the values in v_2, and those vertices with values below a chosen threshold, such as 0, the mean value or the median value, are assigned to one cluster, with the remaining vertices assigned to the second cluster.

To illustrate this approach, we consider the small dataset shown in Table 2.1, which consists of six short text fragments belonging to two disjoint classes which relate to business and football/sport respectively. While these documents are trivial to group by manual inspection, the example raises several interesting problems for automatic clustering. Specifically, even with such a limited number of terms, the problems of synonymy and sparsity are apparent. For instance, while the terms "football" and "soccer" are often used inter-changeably, this knowledge is unavailable to the clustering algorithm and the terms will be treated independently when using the vector space model. Consequently, the documents f_1 and f_3 do not achieve a perfect similarity score using the cosine measure, although they are conceptually identical. Likewise, the documents f_2 and f_3 will achieve

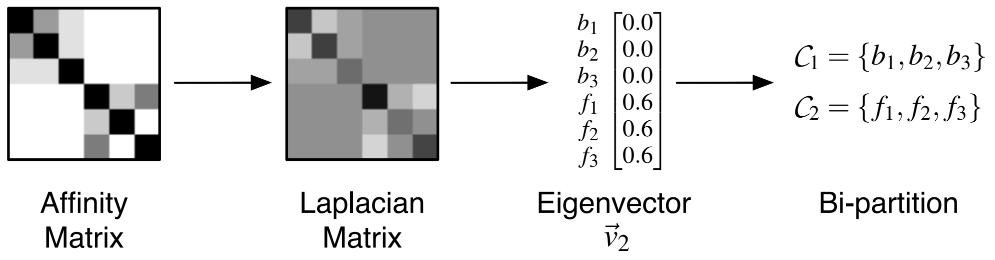


Figure 2.9: Example illustrating the spectral bi-partitioning process for the small dataset described in Table 2.1.

a similarity score of 0 as they share no common terms, even though they are related to the same topic. To uncover the indirect associations between the documents, spectral bi-partitioning may be applied as described previously. The entire bi-partitioning process is shown in Figure 2.9. Observe that, by viewing the values in the Fiedler vector v_2 as continuous membership indicators, the correct partition can be easily obtained by choosing a separator based upon the mean value ($\bar{v_2} = 0.3$).

Spectral methods have also been developed to optimise other, more robust graph partitioning criteria. Notably, Shi & Malik (1997) suggested a spectral approach to finding a bisection that minimises the normalised cut criterion (2.18). A good approximation may be identified in a manner similar to that described previously, but rather using the spectrum of the *normalised Laplacian matrix* of the graph, which is defined as $\mathbf{L_n} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{D}-\mathbf{S})\mathbf{D}^{-\frac{1}{2}}$. Two clusters are formed from the normalised Fiedler vector by sorting the entries and choosing a splitting point along the vector which results in the minimal value for Eqn. 2.18.

2.6.3 K-Way Spectral Clustering

In most cases, we will typically want to partition a dataset into more than two clusters. Two general approaches have been proposed in the literature to extend spectral bi-partitioning to the problem of k-way clustering. The first involves recursively applying spectral bi-partitioning to hierarchically divide each resulting sub-graph until k clusters have been recovered (e.g. Shi & Malik, 1997). However, if k is not a power of 2, it is unclear as to how to choose which segments should be sub-divided.

A more effective approach involves directly producing a k-way partition by constructing an embedding from multiple eigenvectors of the affinity matrix. However, rather than using those vectors corresponding to the smallest eigenvalues, clustering may be performed using the eigenvectors associated with the largest eigenvalues, which also contain structural information. A formal justification for the benefits of clustering in the reduced space formed from these vectors was given in the *polarisation theorem* proposed by Brand & Huang (2003). This theorem asserts that, as an affinity matrix $\mathbf{S}$ is projected onto smaller subsets of successive leading eigenvectors, the angles between highly similar objects are least distorted, while the angles between dissimilar objects tend to greatly increase. Consequently, by magnifying the similarities between objects that belong to the same natural class and attenuating the associations between objects belonging to different classes, the clustering problem will often become easier to solve.

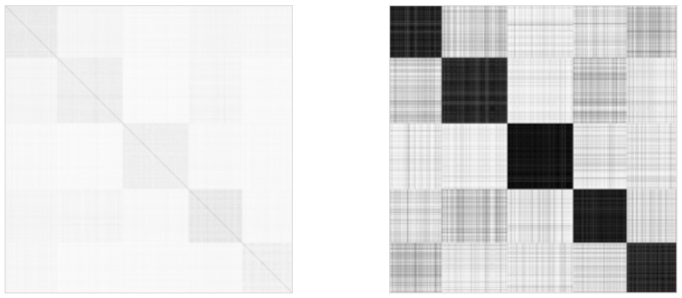
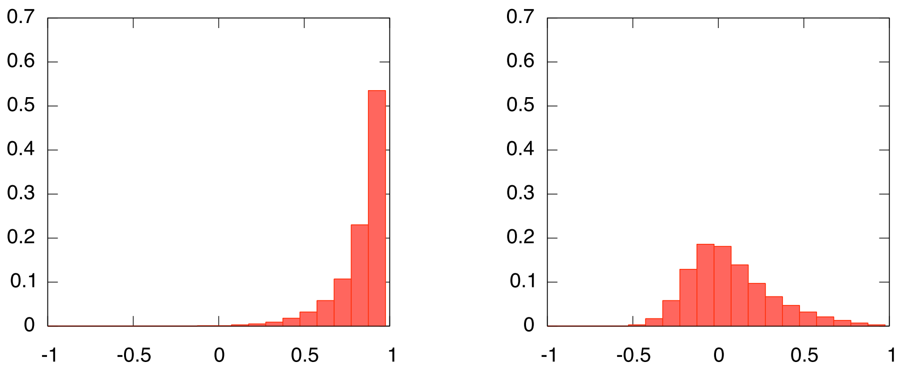

(a) Matrix for the original high-dimensional data.
(b) Matrix for the k-way spectral embedding ($k = 5$).

Figure 2.10: Pairwise cosine similarity matrices generated on the *bbc* corpus, illustrating the exaggeration of the block diagonal structure resulting from k-way spectral embedding.

As an example, we consider the construction of a k-way embedding for the *bbc* dataset. Figure 2.10(a) shows the block-diagonalised cosine similarity matrix generated on the original high-dimensional feature space. Figure 2.10(b) shows the corresponding matrix generated on the spectral embedding formed from the five leading eigenvectors of the original similarity matrix. While the values in the latter are higher in general due to the far lower level of sparsity in the embedded dimensions, it is particularly apparent that there is a significant increase in intra-class similarity as evident from the prominent blocks representing the natural classes in the data. To confirm this, Figure 2.11 provides histograms for the intra-class and inter-class cosine similarity values computed using the embedding. Note that these values now lie in the range $[-1, 1]$ due to the presence of negative values in the embedded dimensions. However, it is clear that the trends in these



(a) Intra-class similarities
(b) Inter-class similarities

Figure 2.11: Histograms of pairwise cosine similarity values for the k-way spectral embedding of the *bbc* corpus ($k = 5$), demonstrating the increase in intra-class similarity values after the application of dimension reduction.

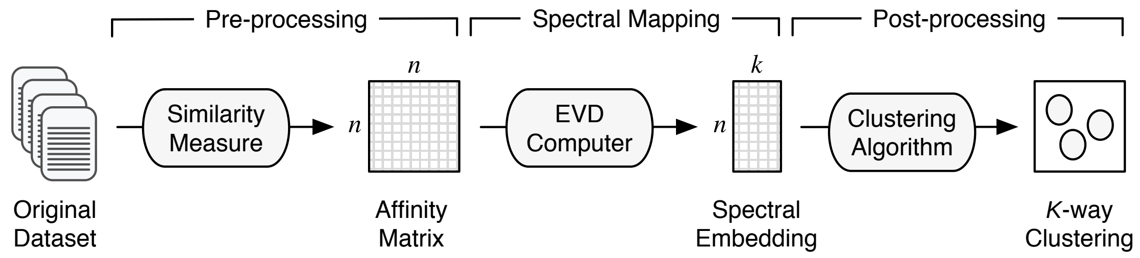


Figure 2.12: Workflow for the k-way spectral clustering process, which consists of three phases: *pre-processing*, *spectral mapping* and *post-processing*.

histograms contrast sharply with those in the histograms produced on the original data as shown in Figure 2.6. The exaggeration of intra-class and inter-class relations makes the goal of accurately uncovering the underlying classes far more attainable.

K-way spectral clustering techniques generally consist of three principal phases, as shown in Figure 2.12: preprocessing, spectral mapping and post-processing (Verma & Meila, 2003). We now describe each of these phases individually and summarise the most popular approaches that have been used to implement them.

Preprocessing: Initially, an affinity matrix $\mathbf{S}$ is constructed from the original data using an appropriate metric, such as the Gaussian kernel function (see Section 2.8.2) for image data or cosine similarity for text documents. As with bi-partitioning, various normalisation techniques may be applied to $\mathbf{S}$ to support the optimisation of different partitioning criteria. Most commonly, an approximation to the k-way normalised cut (2.19) has been used, which is found by computing the truncated EVD of the normalised affinity matrix given by:

$$\mathbf{S_n} = \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}} \tag{2.22}$$

When using this objective, some authors have observed that removing the influence of the diagonal values by setting $S_{ii} = 0$ prior to decomposition results in improved accuracy (*e.g.* Ng *et al.*, 2001).

Spectral mapping: The second phase of the spectral clustering process involves computing the eigenvalue decomposition of the normalised affinity matrix. Ng *et al.* (2001) showed that, when partitioning data into k clusters, the use of eigenvectors corresponding to the k largest eigenvalues affords the best discriminating power.

By stacking these vectors in columns to form $\mathbf{Y} \in \mathbb{R}^{n \times k}$, a reduced-dimensional representation is produced for the original n objects.

Post-processing: The columns of a k-dimensional spectral embedding $\mathbf{Y}$ can be viewed as a set of k semantic variables. However, since these variables may take negative values, they are not immediately interpretable as clusters. In simple cases where the affinity matrix is approximately block diagonal, it may be possible to identify a partition by inspecting the values in $\mathbf{Y}$. However, for real-world data such as text corpora some form of post-processing will be required to extract the final cluster assignments.

A popular approach is to treat the rows $\{y_1, \ldots, y_k\}$ as points in a geometric space $\mathbb{R}^k$ and apply a partitional algorithm, such as standard k-means, to cluster these points. A final clustering of the original dataset may be derived by simply assigning the object x_i to the cluster C_j which contains the corresponding embedded point y_j. It has been shown the quality of this partition may often be improved by normalising the rows of $\mathbf{Y}$ to L2 unit length prior to clustering (Ng *et al.*, 2001). Several other authors have focused on directly decomposing the selected eigenvectors into a set of k clusters without the need for the subsequent application of a clustering algorithm (e.g. Yu & Shi, 2003).

To illustrate how the three phases fit together, a summary of a popular representative algorithm, Ng-Jordan-Weiss (NJW) clustering (Ng *et al.*, 2001), is given in Figure 2.13.

2.6.4 Bipartite Spectral Co-clustering

The techniques described previously in this section focus solely on the problem of grouping the objects in a dataset. However, in certain situations it may be useful to perform *co-clustering*, where both objects and features are assigned to groups simultaneously. Such techniques are related to the *principle of the duality of clustering objects and features*, which states that a clustering of objects induces a clustering of features while a clustering of features also induces a clustering of objects (Dhillon, 2001). The co-clustering problem may be viewed as the task of partitioning a weighted bipartite graph. Formally, given a text corpus, we build a graph $G(\mathcal{V}, \mathcal{E})$ such that $\mathcal{V} = \mathcal{V}_X \cup \mathcal{V}_T$, where $\mathcal{V}_X$ is a set of vertices representing the n documents and $\mathcal{V}_T$ is a set of vertices representing the m terms. An edge (i, j) only exists if the j-th term occurs at least once in the document x_i, where the weight on that edge indicates the number of occurrences. We may conveniently represent such a graph using a term-document matrix $\mathbf{A}$.

While the methods in the previous section involve analysing the eigendecomposition of an affinity matrix, for the bipartite case several authors have suggested the use of the related *singular value decomposition* (SVD), which may be applied to rectangular matrices. Formally, this involves decomposing a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ into the product of three factors:

$$\mathbf{A} = \mathbf{U\Sigma V}^{\mathsf{T}} \tag{2.23}$$

The columns of the matrix $\mathbf{U} \in \mathbb{R}^{m\times m}$ are referred to as the left singular vectors, the rows of $\mathbf{V} \in \mathbb{R}^{n\times n}$ are the right singular vectors, and the diagonal entries of $\mathbf{\Sigma} \in \mathbb{R}^{m\times n}$ are called the singular values of $\mathbf{A}$. Note that the left singular vectors are equivalent to the eigenvectors of $\mathbf{AA}^{\mathsf{T}}$, the right singular vectors are the eigenvectors of $\mathbf{A}^{\mathsf{T}}\mathbf{A}$ and their identical sets of eigenvalues are given by the diagonal of $\mathbf{\Sigma}^2$.

Dhillon (2001) suggested that an approximation for the optimal normalised cut of a bipartite graph represented by a matrix $\mathbf{A}$ may be obtained by analysing the $l = \log_2 k$ leading singular vectors of the degree-normalised matrix given by:

$$\mathbf{A_n} = \mathbf{D_1}^{-\frac{1}{2}}\mathbf{A}\mathbf{D_2}^{-\frac{1}{2}} \tag{2.24}$$

where $\mathbf{D_1}$ and $\mathbf{D_2}$ are diagonal matrices such that

$$[D_1]_{ii} = \sum_{j=1}^{n} A_{ij}\,, \quad [D_2]_{jj} = \sum_{i=1}^{m} A_{ij} \tag{2.25}$$

---

1. Construct an affinity matrix $\mathbf{S} \in \mathbb{R}^{n\times n}$ on the original data $\mathcal{X}$, and set $S_{ii} = 0$.
2. Form the normalised affinity matrix:
$$\mathbf{S_n} = \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}}$$
3. Decompose $\mathbf{S_n}$ and construct an embedding $\mathbf{Y} \in \mathbb{R}^{n\times k}$, such that the columns are given by the eigenvectors corresponding to the k largest eigenvalues.
4. Normalise the rows of $\mathbf{Y}$ to L2 unit length.
5. Apply standard k-means to the rows of $\mathbf{Y}$ to generate a k-way clustering $\mathcal{C}$.
6. Produce a clustering of $\mathcal{X}$ by assigning each object x_i to the j-th cluster if $y_i \in C_j$.

Figure 2.13: Ng-Jordan-Weiss (NJW) spectral clustering algorithm.

If $\mathbf{A}$ is a term-document matrix, the rows of the left truncated vectors $\mathbf{U_l}$ represent a l-dimensional embedding of the terms, while the columns of the right truncated vectors $\mathbf{V_l}$ represent an embedding of the documents. By selecting the leading vectors of the spectral decomposition, we can produce a reduced-dimensional space that amplifies the natural structures in the data. In this case, a unified embedding $\mathbf{Z} \in \mathbb{R}^{(m+n)\times l}$ is constructed by normalising and arranging the truncated factors as follows:

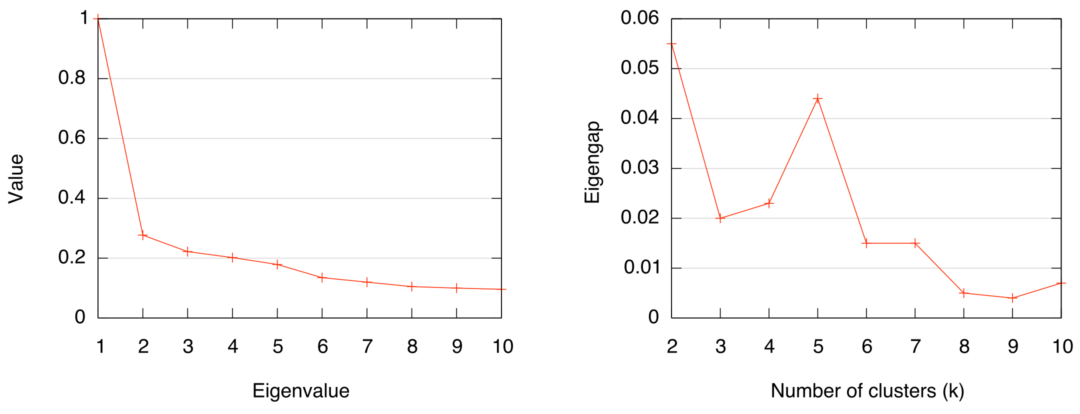
$$\mathbf{Z} = \begin{bmatrix} \mathbf{D_1}^{-1/2}\mathbf{U_l} \\ \mathbf{D_2}^{-1/2}\mathbf{V_l} \end{bmatrix}$$

A partitional clustering algorithm, such as k-means, is then applied in the geometric space $\mathbf{Z}$ to produce a simultaneous k-way partitioning of both documents and terms.

2.6.5 Limitations of Spectral Clustering

State-of-the-art k-way spectral clustering algorithms have become popular in the many domains. However, their performance is highly sensitive to the selection of an appropriate number of leading eigenvectors. A number of authors have suggested choosing k based on the largest eigengap (*e.g.* Ng *et al.*, 2001), which refers to the maximum difference between any two consecutive eigenvalues (*i.e.* $|\lambda_i - \lambda_{i+1}|$). When considering this approach, it is worth noting that both theoretical results and empirical observations suggest that spectral clustering performs best in cases where the rows of the matrix representing the dataset can be arranged in a block diagonal manner. Unfortunately, matrices representing real-world text corpora will contain overlapping structures and are therefore unlikely to fulfil this requirement. This often leads the eigengap method to produce misleading results. An example of this is shown in Figure 2.14, where analysis of the eigenvalues of the cosine similarity matrix for the *bbc* corpus suggests that $k = 2$ is suitable in place of the correct value $k = 5$, despite the fact that the classes are reasonably well separated. In general, traditional model selection approaches, such as those described in the next chapter, tend to be more useful when working with text data.

The issue of the performance of spectral methods when applied to non-block-diagonal matrices is particularly important in document clustering, where the data will frequently contain overlapping groups. It is worth noting that the generation of soft clusterings based on spectral methods has not been considered in the literature. Since soft membership weights for documents and terms can be useful in providing an insight into a newly produced clustering solution, we consider this problem in more detail in Section 6.2.



(a) Leading eigenvalues (b) Corresponding eigengaps

Figure 2.14: Example application of the eigengap method to estimate the optimal number of clusters in the *bbc* corpus, based on the examination of the differences between the leading eigenvalues.

2.7 Non-negative Matrix Factorisation

A disadvantage of spectral clustering methods stems from the presence of negative entries in the eigenvectors or singular vectors used to construct the reduced embedding. As a result, the features of a reduced representation do not have any immediate physical meaning. Therefore, the application of a post-processing technique is generally required to produce a final partition of the data. As noted in Section 2.5, this is an issue that is inherent in many feature extraction techniques. To address this problem, Lee & Seung (1999) proposed an alternative unsupervised approach for reducing the dimensionality of non-negative matrices, referred to as *Non-negative Matrix Factorisation* (NMF). Unlike spectral methods, NMF seeks to decompose the data into factors that are constrained so that they will not contain negative values. By modelling each object as the additive combination of a set of non-negative basis vectors, a readily interpretable clustering of the data can be produced without the requirement for further post-processing. In contrast to techniques that construct a reduced space from eigenvectors, these basis vectors are not required to be orthogonal, which facilitates the discovery of overlapping groups.

NMF also has a natural tendency to produce sparse basis vectors, leading it to perform well on data where structures exist in different low-dimensional subspaces of the original high-dimensional space. In such cases, global dimension reduction techniques such as spectral clustering or PCA may fail to identify these local patterns. In this respect NMF resembles *subspace clustering* techniques (Agrawal *et al.*, 1998), which aim to find clusters in various subspaces of the original feature space by performing the tasks of dimensionality reduction and data clustering simultaneously.

Formally, given a rectangular non-negative matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ such that each column represents a data object, NMF generates a reduced rank-k approximation in the form of the product of two non-negative factors:

$$\mathbf{A} \approx \mathbf{WH} \quad \text{such that} \quad \mathbf{W} \geq 0 \, , \mathbf{H} \geq 0$$

where the rows of $\mathbf{W} \in \mathbb{R}^{m \times k}$ represent a set of k basis vectors and the columns of $\mathbf{H} \in \mathbb{R}^{k \times n}$ are linear combinations of the basis vectors. Typically the number of basis vectors is a user-defined parameter, which is chosen so that $k << min(m, n)$.

While NMF has been primarily applied in bioinformatics (*e.g.* Brunet *et al.*, 2004) and image processing (*e.g.* Li *et al.*, 2001), it may also be used to decompose the term-document representation $\mathbf{A}$ of a corpus. In this context, the factor $\mathbf{W}$ can be viewed as a set of semantic variables corresponding to the topics in the data, while $\mathbf{H}$ describes the contribution of the documents to each topic. This idea of representing each document as the additive combination of several overlapping topics is highly intuitive. Furthermore, the non-negativity of the factors allows them to be directly interpreted as a soft k-way co-clustering of both documents and terms. To illustrate this, we consider the application of NMF to the 2-class subset of the *bbc* corpus described in Section 2.5.3. Table 2.2 shows the highest weighted terms in the NMF basis vectors forming the rows of $\mathbf{W}$, which immediately provide us with labels for the two clusters.

(a) Basis vector w_1

Rank	Term	Weight
1	year	0.15
2	company	0.15
3	firm	0.14
4	market	0.13
5	share	0.11
6	growth	0.11
7	bank	0.11
8	government	0.10
9	sale	0.10
10	economy	0.10

(b) Basis vector w_2

Rank	Term	Weight
1	play	0.16
2	game	0.16
3	win	0.15
4	player	0.13
5	first	0.11
6	match	0.10
7	goal	0.10
8	England	0.10
9	team	0.10
10	club	0.10

Table 2.2: Top ranked terms in the basis vectors produced by applying NMF factorisation to the ‘business’ and ‘sport’ classes from the *bbc* corpus.

2.7.1 Basic Algorithm

The choice of factors in NMF is determined by some objective function that seeks to minimise the error of the reconstruction of **A** by the product **WH**. In the original formulation proposed by Lee & Seung (1999), this involves minimising squared Euclidean distance as computed by the Frobenius norm:

$$f(\mathbf{W},\mathbf{H}) = \frac{1}{2}\,||\mathbf{A} - \mathbf{WH}||_{\mathsf{F}}^{2} \tag{2.26}$$

An approximate solution may be found by employing a diagonally re-scaled gradient descent search strategy. In practice, this involves alternating between a pair of multiplicative update rules, as shown in Figure 2.15. At each iteration, **W** and **H** are updated by multiplying the current factors by a measure of the quality of the current approximation **WH**. The authors showed that the error of this approximation decreases monotonically as these rules are applied until the search procedure converges to a local minimum.

An information theoretic formulation of NMF was also proposed by Lee & Seung (1999), based on the use of generalised *Kullback-Leibler* (KL) divergence, also referred to as *relative entropy*. The clustering objective then becomes the minimisation of the loss of information between **A** and the approximation **WH** as quantified by the expression:

$$D(\mathbf{A}||\mathbf{WH}) = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(A_{ij}\log\frac{A_{ij}}{\sum_{l=1}^{k} W_{il}H_{lj}} - A_{ij} + [\mathbf{WH}]_{ij}\right) \tag{2.27}$$

This function may also be optimised by applying a gradient descent search in the form of a corresponding pair of multiplicative update rules.

---

1. Randomly initialise **W** and **H** with positive values.
2. Update factor **H** for $1 \leq j \leq n$, $1 \leq c \leq k$:

$$H_{cj} \leftarrow H_{cj}\frac{(\mathbf{W}^{\mathsf{T}}\mathbf{A})_{cj}}{(\mathbf{W}^{\mathsf{T}}\mathbf{WH})_{cj}}$$

3. Update factor **W** for $1 \leq i \leq m$, $1 \leq c \leq k$:

$$W_{ic} \leftarrow W_{ic}\frac{(\mathbf{AH}^{\mathsf{T}})_{ic}}{(\mathbf{WHH}^{\mathsf{T}})_{ic}}$$

4. Repeat from Step 2 until convergence or maximum number of iterations have elapsed.

Figure 2.15: Euclidean-based NMF algorithm, using multiplicative update rules.

2.7.2 Variations

Recently, a number of techniques have been proposed that involve imposing additional constraints on the matrices **U** and **V** to enforce a certain degree of sparsity. While the standard NMF formulations described previously do tend to generate reasonably sparse factors, it may be desirable in some situations to increase the level of sparsity in the basis vectors to produce a more localised solution. Several algorithms have been proposed that implement sparsity constraints by including additional penalty terms in the objective function, including Local Non-negative Matrix Factorisation (LNMF) (Li *et al.*, 2001) and Sparse Non-negative Matrix Factorisation (SNMF) (Liu *et al.*, 2003b).

NMF techniques may also be applied to produce a rank-k non-negative approximation of a symmetric matrix. Given a positive semi-definite similarity matrix **S**, Ding & He (2005) showed that a solution to the problem of minimising the SSE error (2.7) may be found by performing a symmetric factorisation of **S** based on the objective function:

$$f(\mathbf{V}) = \left|\left|\mathbf{S} - \mathbf{V}\mathbf{V}^{\mathsf{T}}\right|\right|_{\mathsf{F}}^{2} \tag{2.28}$$

The optimal factor $\mathbf{V} \in \mathbb{R}^{n \times k}$ can be approximated using a single update rule

$$V_{cj} \leftarrow V_{cj} \left(1 - \beta + \beta \frac{(\mathbf{S}\mathbf{V})_{cj}}{(\mathbf{V}\mathbf{V}^{\mathsf{T}}\mathbf{V})_{cj}}\right)$$

where $0 < \beta \leq 1$ is a user-defined parameter which controls the rate of convergence.

2.7.3 Limitations

Several authors have observed the sensitivity of NMF to the choice of initial factors. The standard approach in the literature has been to initialise the factors **U** and **V** with randomly generated values. However, this can lead to inconsistent results over multiple trials in the same way as stochastically initialised k-means. We discuss this issue further in Section 6.2.

Another drawback of NMF stems from the fact that the standard multiplicative update techniques are often very slow to converge. This is particularly significant as every iteration will typically require several costly matrix multiplication operations. For instance, the update rules for the algorithm given in Figure 2.15 involve six multiplications, each requiring time $O(n^3)$. Even when sparse matrix multiplication techniques are applied to make the process more efficient, the running time can still be prohibitive for large datasets.

2.8 Kernel Clustering Methods

Kernel methods involve the transformation of a dataset to a new, possibly high-dimensional space where non-linear relationships between objects may be more easily identified. Rather than explicitly computing the transformed representation $\phi(x)$ of each data object x, the application of the "kernel trick" (Aizerman *et al.*, 1964) allows us to consider the affinity between a pair of objects (x_i,x_j) using a kernel function κ, which is defined in terms of the dot product:

$$\kappa(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \tag{2.29}$$

In practice, the function κ is represented by an $n \times n$ symmetric, positive semi-definite *kernel matrix* (or Gram matrix) $\mathbf{K}$, such that $K_{ij} = \kappa(x_i, x_j)$. By re-formulating algorithms using only dot products and subsequently replacing these with affinity values from $\mathbf{K}$, we can efficiently apply learning algorithms in the new kernel space.

Another significant advantage of kernel methods is their modularity, where every method is composed of two decoupled components: a generic learning algorithm, and a problem-specific kernel function. Consequently, it is possible to develop algorithms that can readily be deployed in a wide range of domains without requiring any customisation. Novel kernels can also be constructed in a modular fashion by chaining together multiple existing functions together.

The main focus of research in this area has been on the development of techniques for supervised tasks, notably the well-known *support vector machine* (SVM) classifier (Cristianini & Shawe-Taylor, 2000). However, kernel methods have also been shown to be effective in unsupervised problems, such as document clustering (Dhillon *et al.*, 2004b). We now describe several algorithms and kernel functions that are relevant in this context.

2.8.1 Algorithms

Kernel k-means

A variety of popular clustering techniques have been re-formulated for use in a kernel-induced space, including the standard k-means algorithm. To describe the algorithm, we firstly observe that, using the notation given above, the squared Euclidean distance between a pair of objects in the kernel space represented by a matrix $\mathbf{K}$ can be expressed as:

$$||\phi(x_i) - \phi(x_j)||^2 = K_{ii} + K_{jj} - 2K_{ij} \tag{2.30}$$

This may be used as a starting point for the identification of cluster structures. Formally, given a set of objects $\{x_1, \ldots, x_n\}$, the kernel k-means algorithm (Schölkopf *et al.*, 1998) seeks to minimise the distortion between the objects and the "pseudo-centroids" $\{\mu_1, \ldots, \mu_k\}$ in the new space:

$$\sum_{c=1}^{k} \sum_{x_i \in C_c} ||\phi(x_i) - \mu_c||^2 \quad \text{where} \quad \mu_c = \frac{\sum_{x_i \in C_c} \phi(x_i)}{|C_c|} \tag{2.31}$$

Note that this expression is analogous to the SSE objective (2.7) used in standard k-means. However, rather than explicitly constructing centroid vectors in the kernel space, distances are computed using dot products only. From Eqn. 2.30, we can formulate squared object-centroid distance by the expression:

$$||\phi(x_i) - \mu_c||^2 = K_{ii} + \frac{\sum_{x_j, x_l \in C_c} K_{jl}}{|C_c|^2} - \frac{2\sum_{x_j \in C_c} K_{ij}}{|C_c|} \tag{2.32}$$

The first term above may be excluded as it remains constant; the second is a common term representing the self-similarity of the centroid μ_c, which only needs to be calculated once for each cluster; the third term represents the affinity between x_i and the centroid of the cluster C_c.

This kernelised algorithm has a significant advantage over standard k-means in the sense that, given an appropriate kernel function, it can be used to identify structures that are not necessarily spherical or convex. In addition, once we have constructed a single matrix $\mathbf{K}$, multiple partitions may be subsequently generated without referring back to the original feature space.

---

1. Select k arbitrary initial clusters $\{C_1, \ldots, C_k\}$.
2. For each object $x_i \in \mathcal{X}$ and centroid μ_c, compute the distance:

$$d(x_i, \mu_c) = K_{ii} + \frac{\sum_{x_j, x_l \in C_c} K_{jl}}{|C_c|^2} - \frac{2\sum_{x_j \in C_c} K_{ij}}{|C_c|}$$

3. Assign each x_i to cluster corresponding to nearest centroid.
4. Repeat from Step 2 until termination criterion is satisfied.

Figure 2.16: Kernel k-means (KKM) algorithm.

Weighted kernel k-means

In the standard k-means algorithm and its kernel equivalent, each data object in a cluster makes an equal contribution towards the location the cluster centroid. An extension of the kernel k-means algorithm was proposed by Dhillon *et al.* (2004b), where each object x_i has an associated non-negative weight w_i indicating its relative importance. The objective function given in Eqn. 2.31 then becomes:

$$\sum_{c=1}^{k} \sum_{x_i \in C_c} w_i \, ||\phi(x_i) - \mu_c||^2 \quad \text{where} \;\; \mu_c = \frac{\sum_{x_i \in C_c} w_i \phi(x_i)}{\sum_{x_i \in C_c} w_i} \tag{2.33}$$

Since the weight for each object must be taken into consideration when computing object-centroid distances, the expression Eqn. 2.32 must be altered as follows:

$$||\phi(x_i) - \mu_c||^2 = K_{ii} + \frac{\sum_{x_j, x_l \in C_c} w_j w_l K_{jl}}{(\sum_{x_i \in C_c} w_i)^2} - \frac{2 \sum_{x_j \in C_c} w_j K_{ij}}{\sum_{x_i \in C_c} w_i} \tag{2.34}$$

Note that the original kernel k-means algorithm can be viewed as a specific case of the weighted variant, where $w_i = 1 \; \forall i$.

Dhillon *et al.* (2004b) showed that many of the graph partitioning criteria given in Section 2.6.1 can be shown to be equivalent to the weighted objective function (2.33) when using a particular kernel matrix and set of weights. For instance, an approximation for the normalised cut criterion may be found by applying the algorithm to the matrix

$$\mathbf{K} = \mathbf{D}^{-1}\mathbf{S}\mathbf{D}^{-1} \tag{2.35}$$

where $\mathbf{D}$ is the degree matrix of $\mathbf{K}$ and object weights are chosen so that $w_i = D_{ii}$.

2.8.2 Kernel Functions

The selection of a suitable kernel function for a particular learning problem is crucial to the success of kernel methods. In general, any positive semi-definite matrix can be viewed as a kernel matrix (Cristianini & Shawe-Taylor, 2000). We discuss here several popular kernel functions that may be applied to text data.

Normalised linear kernel: As noted previously, the most common choice of similarity measure for text data is cosine similarity (2.4). A corresponding kernel function is formed by computing dot products and normalising by self-similarity (Schölkopf & Smola, 2001). Formally, the affinity between two documents x_i and x_j is defined as:

$$\kappa(x_i, x_j) = \frac{\langle x_i, x_j \rangle}{\sqrt{\langle x_i, x_i \rangle \, \langle x_j, x_j \rangle}} \tag{2.36}$$

This function yields values in the range $[0, 1]$ and self-similarity values $\kappa(x_i, x_i) = 1$.

Gaussian kernel: The Gaussian Radial Basis Function (RBF) kernel is widely used in a variety of domains and applications, notable image processing. Formally, the function is defined by

$$\kappa(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{\sigma^2}\right) \tag{2.37}$$

where $\sigma \in \mathbb{R}^+$ is a user-defined smoothing parameter, often referred to as the kernel width. The choice of a suitable value for σ can greatly influence the structures that are generated by a kernel clustering algorithm and its selection represents a non-trivial parameter selection problem (Lee & Daniels, 2005).

Polynomial kernel: Given an existing kernel expressed in terms of dot products, a new kernel can be created by applying the polynomial function

$$\kappa(x_i, x_j) = \langle x_i, x_i \rangle^p \tag{2.38}$$

for some positive integer degree p. As with the Gaussian kernel, the selection of a value for the exponent can have a significant impact upon the success of the learning algorithm.

String kernels: It is worth noting that kernel functions which operate on the traditional feature vector representation of documents will also be subject to certain drawbacks of the vector space model, notably the loss of context information regarding the relative placement of terms in documents. To preserve this information, Lodhi *et al.* (2002) proposed the use of *string kernels*, which involve computing similarities between documents by finding matching non-consecutive sub-sequences of characters. Rather than making use of individual characters, Cancedda *et al.* (2003) proposed a kernel based on word sequences which proved successful in text classification problems. However, in contrast to a simple linear kernel, the construction of string kernels is often complex and computationally expensive.

2.9 Ensemble Clustering Methods

Ensemble techniques have been successfully applied in supervised learning to improve the accuracy of classification algorithms, where the rationale is that the combined judgement

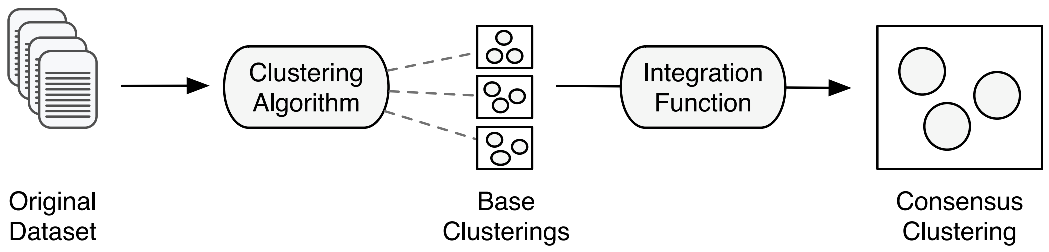


Figure 2.17: Generic ensemble clustering workflow, illustrating the *generation* and *integration* phases of the ensemble process.

of a group of predictors is frequently superior to that of an individual (Breiman, 1996). In contrast, cluster analysis methods have often involved the repeated execution of a clustering procedure, followed by the manual selection of a single solution that optimises an internal validation criterion. However, rather than merely selecting a “winning” partition, recent work has shown that combining the strengths of an ensemble of clusterings can often yield better results (*e.g.* Fred, 2001; Strehl & Ghosh, 2002b).

Given a collection of “base” clusterings generated on data originating from the same source, the primary aim of *ensemble clustering* is to aggregate the information provided the members of the collection to produce a more accurate solution. Additionally, ensemble methods can often afford greater *stability*, which refers to the ability of a clustering procedure to consistently produce similar solutions across multiple trials. Even though the underlying clustering algorithm, such as k-means with random initialisation, may generate inconsistent solutions of varying accuracy, by combining these solutions it may be possible to produce a single definitive output. Another potential benefit of these methods, which has received little attention, is in the reuse of knowledge from previously generated clusterings (Strehl & Ghosh, 2002b). Rather than constructing a new partition of the domain data from scratch, it may be preferable to aggregate legacy solutions or use them as a source of background information when producing a new clustering.

A range of approaches to ensemble clustering have been proposed in the literature, which have been referred to by different authors as “cluster combination”, “meta-clustering”, “evidence accumulation” and “cluster fusion”. However, these methods generally follow a common workflow as illustrated in Figure 2.17, which consists of two distinct phases:

1. *Generation:* Construct a collection of τ base clustering solutions, denoted $\mathbb{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_\tau\}$, which represent the members of the ensemble. This is typically done

by repeatedly applying a chosen clustering algorithm in a manner that leads to diversity among the members.

2. *Integration:* Once a collection of ensemble members has been generated, a suitable integration function is applied to combine them to produce a final *consensus clustering* $\bar{\mathcal{C}}$:

$$f : \{\mathcal{C}_1, \ldots, \mathcal{C}_\tau\} \rightarrow \bar{\mathcal{C}}$$

   In practice, this often involves the application of an additional clustering procedure to an intermediate representation of $\mathbb{C}$.

2.9.1 Generation Techniques

It has frequently been demonstrated that supervised ensembles are most successful when constructed from a set of accurate classifiers whose errors lie in different parts of the data space (*e.g.* Opitz & Shavlik, 1996). Similarly, unsupervised ensemble procedures typically seek to encourage diversity with a view to improving the quality of the information available in the integration phase. For instance, Topchy *et al.* (2005) suggested that, given a collection of poor quality partitions of a dataset, a useful consensus clustering may be produced if the collection is sufficiently diverse. In these cases, the lack of accuracy in the ensemble members is compensated for by their diversity. However, in general, empirical results suggest that it is preferable to make use of reasonably accurate clusterings that adequately cover the space of possible solutions (Fern & Brodley, 2003).

A simple approach is to rely on the inherent instability of common clustering algorithms such as standard k-means. By employing a stochastic initialisation scheme, the algorithm may often converge to different local solutions (Jain & Fred, 2002b). However, for many datasets this may not result in a sufficient level of diversity. Additionally, the effect of outlying objects may not be negated. Consequently, a variety of strategies have been used in the clustering literature to artificially introduce instabilities in clustering algorithms. We now summarise several noteworthy ensemble generation approaches:

Data sampling: The most commonly employed strategy has been to use unbiased random sampling to produce partitions on different parts of the same dataset. Many authors (*e.g.* Leisch, 1999; Dudoit & Fridlyand, 2003) have suggested the use of bootstrapping aggregation or "bagging", where subsets of the original data are produced by independently drawing with replacement. A related technique involves applying subsampling without replacement, where typically 60-80% of the data objects are included when generating each base clustering (Minaei-Bidgoli *et al.*, 2004; Fern & Brodley, 2004). Having chosen a sample of data, an ensemble member is generated by applying a suitable base clustering algorithm, such as standard k-means with random initialisation.

Feature selection: Another technique that has frequently been applied in supervised learning to introduce diversity is to use different feature subsets, so that individual ensemble members have only a partial view of each data object. A simple approach is to employ random subspacing (Ho, 1998), where each member is generated on a randomly selected subset of the original dimensions. This technique was applied to produce ensemble clusterings of medical and synthetic data by Greene *et al.* (2004).

Feature extraction: Fern & Brodley (2003) proposed the generation of a diverse set of base clusterings by randomly projecting data onto a lower dimensional subspace. Each ensemble member is generated by transforming the $n \times d$ data set to a reduced set of d' new dimensions. This transformation is defined by a $d \times d'$ matrix R, which is populated by randomly selecting values in the range $[0, 1]$ and normalising the entries in each column to sum to 1. After performing the mapping, the authors applied the EM clustering algorithm to the new data. Other feature extraction techniques could potentially be used in this context (e.g. spectral embedding). However, the computational cost and global nature of many of these methods makes them less attractive.

Random parameter selection: As noted in Section 2.3, the output of partitional algorithms such as k-means is dependent on the initial choice of the number of clusters k. This has been exploited as a source of ensemble diversity by generating clusterings using randomly selected values of k from a user-specified interval (Jain & Fred, 2002b). Ghosh *et al.* (2002) observed that better results can sometimes be achieved by combining a collection of clusterings generated at a much higher resolution than the value of k used in the final clustering.

Heterogeneous ensembles: Typically in ensemble clustering, members are generated over multiple runs of the same clustering algorithm. As an alternative, *heterogeneous* ensembles may be employed, where diversity is induced by allowing each base clustering to potentially be generated using a different algorithm (Greene *et al.*, 2004). The motivation here is that these algorithms may differ in their strengths and weaknesses, so that an ensemble can combine the best aspects of each approach.

2.9.2 Integration Techniques

Once a collection of diverse ensemble members has been generated, a strategy is required to combine these clusterings to produce a single solution. In supervised learning, it has been observed that the success of an ensemble technique depends not only on the presence of a diverse set of base classifiers, but also on the ability of the aggregating classifier to exploit the resulting diversity (Brodley & Lane, 1996). Similarly, the choice of a suitable strategy for integrating an ensemble of clusterings will greatly affect the accuracy of the final clustering solution (Greene *et al.*, 2004). We now summarise three distinct integration techniques that have been frequently used in the ensemble clustering literature.

Analysis of pairwise co-assignments

The most popular integration strategy has been to use information derived from different clusterings to determine the level of association between each pair of objects in a dataset. The fundamental assumption underlying this strategy is that pairs of objects belonging to the same natural class will frequently be co-assigned during repeated executions of a clustering algorithm. This approach was initially proposed by Fred (2001), who referred to it as "evidence accumulation". Its use was motivated by majority voting schemes commonly employed in classifier ensembles, since each pair of co-assigned objects may be viewed as a vote for the pair being assigned to the same cluster in the final partition. Strehl & Ghosh (2002b) independently proposed an equivalent strategy, which they refer to as "cluster-based similarity partitioning". This strategy was motivated by the observation that pairwise co-assignments, averaged over a sufficiently large number of clusterings, may be used to induce a new measure of similarity on the data.

In practice, integration techniques based on co-association involve the construction of a symmetric $n \times n$ matrix $\mathbf{M}$, such that M_{ij} indicates the fraction of clusterings in $\mathbb{C}$ in which both x_i and x_j were assigned to the same cluster. As suggested by Monti *et al.* (2003), when using resampling-based ensemble generation, the normalisation of $\mathbf{M}$ should only take into account the total number of clusterings in which pairs of objects were both present. Once this intermediate representation has been constructed, a standard similarity-based clustering algorithm may be applied to $\mathbf{M}$ to produce a consensus solution. Algorithms that have been employed for this purpose include single linkage agglomerative clustering (Jain & Fred, 2002a) and multi-level graph partitioning (Strehl & Ghosh, 2002b).

Integration based on cluster graphs

Rather than examining the pairwise associations between objects, several authors have suggested examining the relations between the clusters contained in all partitions in $\mathbb{C}$. Strehl & Ghosh (2002b) proposed a technique that involves modelling the set of partitions $\mathbb{C}$ as a weighted hypergraph, where each vertex represents a cluster from one of the partitions and the edge weight between a pair of vertices is computed based upon the similarity between the corresponding pair of clusters. This similarity may be computed using standard techniques for comparing disjoint partitions, such as the Jaccard index (see Section 3.3). A grouping of the clusters is then found using an appropriate k-way graph partitioning algorithm. Each resulting group of clusters can be regarded as a "meta-cluster". A final clustering may be derived by assigning each object to the meta-cluster containing the largest number of clusters to which the object has been assigned.

Fern & Brodley (2004) proposed formulating the information contained in $\mathbb{C}$ as an unweighted bipartite graph, which preserves both the associations between objects as well as those between clusters. Formally, this involves the construction of a graph $G(\mathcal{V}, \mathcal{E})$ such that $\mathcal{V} = \mathcal{V}_X \cup \mathcal{V}_C$, where $\mathcal{V}_X$ represents the n data objects and $\mathcal{V}_C$ represents the set of all individual clusters in $\mathbb{C}$. An edge (i, j) exists only if the object x_i was assigned to cluster j. A consensus clustering may be found by applying a bipartite graph partitioning algorithm such as spectral co-clustering.

Integration by cluster correspondence

The graph-based approach proposed by Strehl & Ghosh (2002b) is based on the assumption that there will be a direct relationship between individual clusters across different partitions in $\mathbb{C}$. This concept of correspondence, which is sometimes referred to as cluster voting, has been explicitly used by several authors for combining collections of clusterings. Since there is no intuitive way of finding the correspondence between all base clusterings in a single pass, Dimitriadou *et al.* (2002) proposed a heuristic approach where, for each newly generated base clustering, the new clusters are mapped to the existing clusters in the current consensus clustering. This mapping is performed by matching each pair of clusters that have the highest fraction of objects in common. The transformed cluster assignments may then be viewed as votes indicating associations between the data objects and the k clusters in the current consensus clustering. By repeatedly performing this process, the τ base clusterings may be combined to form a k-way fuzzy clustering of the data. Dudoit & Fridlyand (2003) described a similar algorithm, referred to as *BagClust1*, which involves finding the optimal alignment between the clusters in each newly generated base partition and those in the existing current clustering. Once all members have been added, a final clustering is obtained by taking the majority cluster for each data object.

2.9.3 Limitations

A serious drawback of ensemble techniques in general is the computational cost of repeatedly processing a dataset. In ensemble clustering, the generation and aggregation of many solutions can be impractical for large, high-dimensional datasets such as text corpora. While reducing the number of base clusterings appears to be an intuitive solution, this can result in an unstable solution that is little better than that produced by the base clustering algorithm. Unfortunately, the feasibility of applying the ensemble clustering techniques described previously may be greatly limited by the number of objects in the data. The corresponding storage overhead required for techniques such as bipartite graph-based integration can also be prohibitive.

It is also the case that ensemble techniques require a series of key design issues to be addressed. Firstly, how do we generate a collection of base clusterings from which an ensemble is composed? Secondly, how many base clusterings must be aggregated to produce a stable, accurate solution? Thirdly, how can we aggregate the ensemble members to produce the final consensus clustering? This raises a number of parameter selection issues in addition to those pertaining to the base clustering algorithm. In particular, we have observed that the choice of a suitable integration method can greatly dictate the success of ensemble clustering (Greene *et al.*, 2004). Although traditional clustering algorithms have often been used in the integration phase, the limitations of those algorithms will often impact upon the accuracy of the resulting consensus clustering. For instance, while the application of single linkage AHC based on a co-assignment matrix has been the most popular choice for ensemble clustering in the literature, inaccurate clusterings may still be generated due to poor merging decisions and the chaining phenomenon described in Section 2.4.1. To address this problem, Strehl & Ghosh (2002b) suggested the use of a "supra-consensus function" which combines the output of multiple integration methods to produce a more robust solution. However, such an approach naturally leads to additional computational expense.

2.10 Summary

In this chapter, we have reviewed a variety of clustering paradigms, ranging from partitional and hierarchical methods proposed in the classical clustering literature to state-of-the-art approaches based on concepts such as spectral analysis and kernel learning. However, it is apparent that many well-known techniques, which have been widely employed in document clustering, suffer from non-trivial limitations. This particularly relates to important issues, such as the production of accurate clusterings in the presence of overlapping groups or the ability to scale to large datasets. In the latter chapters of this thesis we empirically examine these issues and propose novel solutions that are suitable for use in the context of document clustering.

Chapter 3

Cluster Validation Methods for Text Data

3.1 Introduction

In the previous chapter, we discussed a variety of clustering algorithms suitable for analysing text corpora. We now consider the task of assessing the validity of the output of a clustering algorithm, which represents a fundamental problem in unsupervised learning. Unlike in classification tasks, cluster analysis procedures will generally be unable to refer to predefined class labels when employed in real-world applications. Consequently, there is no clear definition of what constitutes a correct clustering for a given dataset. As a result, it may be difficult to distinguish between a solution consisting of groups that accurately reflect the underlying patterns in the data and one that does not provide the user with any helpful insight. While it may be possible in some cases for a domain expert to manually evaluate a clustering solution, this will be unfeasible for larger datasets and may introduce an element of human bias. For instance, while some authors have suggested producing a hierarchical clustering of a corpus and identifying the "best" cut-off point on a dendrogram by inspection, it is likely that different users will suggest different cutting levels.

In contrast, *cluster validation* methods automatically produce a quantitative evaluation, which can be highly useful both in the exploratory analysis of data and in the design of new clustering algorithms. The validation problem can be viewed as comprising of several different tasks:

Examining cluster tendency: In certain applications, a crucial initial step in the cluster analysis process is to determine whether any significant structures exist in a dataset at all. However, in the document clustering literature it has been common to assume that text corpora will contain at least two identifiable topics.

Model selection: This task relates to the identification of an appropriate clustering algorithm and a corresponding set of parameter values. In the context of document clustering, a particularly important model selection problem is that of estimating the optimal number of clusters in a corpus, denoted by $\hat{k}$. For certain datasets, there may be several reasonable values for $\hat{k}$.

Relative comparison: It is often necessary to directly compare two or more candidate clusterings of the same dataset. This comparison may be performed as part of model selection, or may be used to evaluate the performance of a newly proposed clustering algorithm, relative to that afforded by existing algorithms.

Stability analysis: When a clustering solution is generated using an algorithm that contains a stochastic element or requires the selection of key parameter values, it is important to consider whether the solution represents a "definitive" solution that may easily be replicated. This can typically be determined by assessing the level of pairwise agreement between two or more clusterings of the same data.

A wide variety of validation methods have been proposed in the cluster analysis literature, which pertain to one or more of the above tasks. In the remainder of this chapter we review a range of methods, both classical and contemporary, many of which are relevant for document clustering. These methods are often organised into three distinct categories:

1. *Internal validation:* Compare clustering solutions based on the goodness of fit between each clustering and the raw data on which the solutions were generated.

2. *External validation:* Assess the agreement between the output of a clustering algorithm and a predefined reference partition that is unavailable during the clustering process.

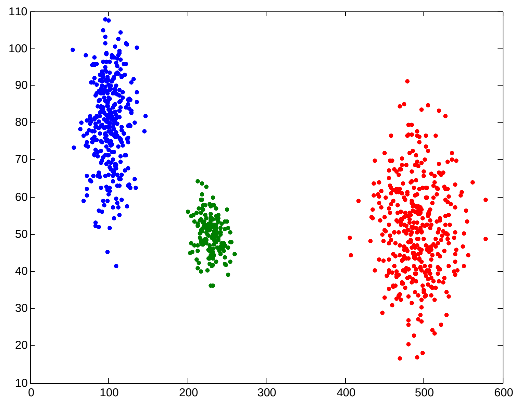
3. *Stability-based validation:* Evaluate the suitability of a given clustering model by examining the consistency of solutions generated by the model over multiple trials.

3.2 Internal Validation

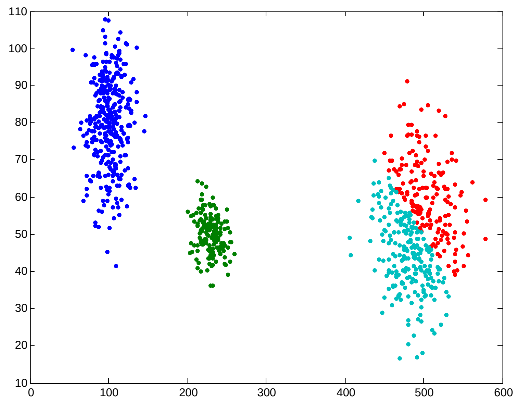
Internal validation methods are designed to provide a means of systematically assessing the quality of a clustering based on some evaluation function, which usually takes the form of an index measuring the goodness of fit between the clustering and the data on which it was generated. This evaluation is based solely on aspects of the features and metrics used during the clustering process, without considering any additional information or external supervision. In certain cases, these indices can also be used to provide an objective function for clustering, although many are intractable to optimise directly. Consequently, internal validation techniques are generally applied after the completion of the clustering procedure.

Model selection in areas such as bioinformatics has frequently been performed by using internal techniques (Bolshakova & Azuaje, 2002). Specifically, it is common to generate multiple clusterings of the data for a range of reasonable parameter values. As noted in Section 1.2.1, for knowledge discovery tasks it may be necessary to repeatedly adjust parameter values and reapply the clustering algorithm until a useful solution is obtained. To guide this process, one or more internal validation indices may be employed to assess the quality of different solutions. A set of suitable parameter values may be identified by locating a solution which optimises these indices.

It is common for internal validation indices to measure goodness-of-fit by examining aspects of a clustering solution such as intra-cluster compactness and inter-cluster separation. To illustrate this idea, we consider the simple two-dimensional data depicted in Figure 3.1, for which we wish to choose a suitable value for the number of clusters k. An



(a) Well-separated clusters ($k = 3$)



(b) Poorly-separated clusters ($k = 4$)

Figure 3.1: Two possible clusterings of a simple synthetic dataset containing three well-separated groups, which illustrates the importance of inter-cluster separation.

internal index is likely to favour the first clustering in Figure 3.1(a), which consists of three clusters that are relatively compact and well-separated. However, the partition shown in Figure 3.1(b) is clearly a poor fit for the data, as two of the clusters are not well-separated. Therefore, evaluating the second partition with the same index should result in a relatively poor score. From this, we can conclude that $k = 3$ is likely to represent a more suitable choice for the data.

3.2.1 Classical Internal Validation Techniques

We now discuss a number of internal indices that have been traditionally used to evaluate hard clusterings. Note that, while many of these indices were originally designed to make use of Euclidean distances, it may be more appropriate to use cosine distance (2.5) when working with text data. For the remainder of this section, we assume the use of an arbitrary distance metric $d(x, y)$, unless otherwise stated.

Calinski-Harabasz index

Motivated by the clustering objectives used in well-known partitional algorithms, a number of internal indices have been proposed which assess cluster quality by considering the squared distances between data objects and cluster representatives. Formally, the *within-cluster sum of squares* is the total of the squared distances between each object x_i and the centroid of the cluster C_c to which it has been assigned:

$$W(\mathcal{C}) = \sum_{c=1}^{k} \sum_{x_i \in C_c} d(x_i, \mu_c)^2$$

When employing Euclidean distance, this is equivalent to the SSE function (2.7) used in the standard k-means algorithm. The *between-cluster sum of squares* is the total of the squares of the distances between the each cluster centroid and the centroid of the entire dataset, denoted $\hat{\mu}$:

$$B(\mathcal{C}) = \sum_{c=1}^{k} |C_c| \; d(\mu_c, \hat{\mu})^2 \quad \text{where} \quad \hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

The statistics $W(\mathcal{C})$ and $B(\mathcal{C})$ have been combined in a number ways by different authors for the purposes of validation. A representative example, the *Calinski-Harabasz* (CH) index (Calinski & Harabasz, 1974), involves computing the normalised ratio of within-cluster relative to inter-cluster scatter:

$$CH(\mathcal{C}) = \frac{B(\mathcal{C})/(k-1)}{W(\mathcal{C})/(n-k)} \tag{3.1}$$

A larger value is indicative of greater internal cohesion and a large degree of separation between the clusters in $\mathcal{C}$. This index has been frequently used as a means of automatically selecting the number of cluster in data, particularly in conjunction with agglomerative hierarchical clustering methods. It has also recently been applied to the task of model selection in document clustering (Surdeanu *et al.*, 2005).

Generalised Dunn's index

Many popular cluster validation indices are based on the assumption that a correct clustering will minimise intra-cluster dissimilarity, while simultaneously maximising inter-cluster dissimilarity. A prototypical index is that proposed by Dunn (1974b), which was designed to reward "compact and well separated clusters". This index was generalised by Bezdek & Pal (1995) to support the use of arbitrary cluster evaluation criteria. Formally, for a disjoint k-way clustering, we let $\Delta : C \rightarrow \mathbb{R}$ denote a function that evaluates the intra-cluster dissimilarity or diameter of a cluster in $\mathcal{C}$, and let $\delta : C \times C \rightarrow \mathbb{R}$ denote a function that evaluates the inter-cluster dissimilarity between a pair of clusters. An overall evaluation for $\mathcal{C}$ is calculated using the expression:

$$D(\mathcal{C}) = \min_{1 \leq i \leq k} \left\{ \min_{1 \leq j \leq k, i \neq j} \left\{ \frac{\delta(C_i, C_j)}{\max_{1 \leq l \leq k} \{\Delta(C_l)\}} \right\} \right\} \tag{3.2}$$

A larger value for $D(\mathcal{C})$ indicates that the clustering $\mathcal{C}$ consists of compact clusters which are well-separated.

To evaluate clusters, the original formulation of Dunn's index made use of complete intra-cluster diameter and single-linkage inter-cluster distance, as defined by:

$$\Delta_1(C_i) = \max_{x \in C_i, y \in C_i} \{d(x,y)\} \qquad \delta_1(C_i, C_j) = \min_{x \in C_i, y \in C_j} \{d(x,y)\} \tag{3.3}$$

Since both functions only make use of a single distance value corresponding to the most extreme case, this formulation is highly sensitive to the presence of outliers. An alternative approach is to include the contribution of all objects in a cluster by considering object-centroid scatter and measuring inter-cluster dissimilarity in terms of the distance between centroids:

$$\Delta_2(C_i) = 2\left(\frac{\sum_{x \in C_i} d(x, \mu_i)}{|C_i|}\right) \qquad \delta_2(C_i, C_j) = d(\mu_i, \mu_j) \tag{3.4}$$

Bezdek & Pal (1995) suggested that, by considering average object-centroid distance together with average-linkage inter-cluster distance, more robust cluster evaluations can be produced:

$$\Delta_3(C_i) = 2\left(\frac{\sum_{x \in C_i} d(x, \mu_i)}{|C_i|}\right) \qquad \delta_3(C_i, C_j) = \frac{\sum_{x \in C_i, y \in C_j} d(x, y)}{|C_i|\,|C_j|} \tag{3.5}$$

It should be noted that evaluation criteria based on object-centroid distances will often exhibit an unfair bias toward spherical clusters in the same way as the standard k-means algorithm, leading to the production of misleading results on data where the underlying groups are elongated or non-convex in structure.

Davies-Bouldin index

Davies & Bouldin (1979) proposed a related internal validation technique that considers the ratio of intra-cluster scatter to inter-cluster separability across all k groups in a clustering. Formally, the *DB index* is defined as a function of the proximity between each cluster and its nearest neighbour:

$$DB(\mathcal{C}) = \frac{1}{k}\sum_{i=1}^{k} \max_{i \neq j} \left\{ \frac{\Delta(C_i) + \Delta(C_j)}{\delta(C_i, C_j)} \right\} \tag{3.6}$$

This value will decrease as clusters become more compact and more distinctly separated, making smaller values for this index desirable. As with Dunn's index, arbitrary cluster evaluation functions can potentially be used in Eqn. 3.6. However, typically the centroid-based metrics defined in Eqn. 3.4 are employed to assess scatter and inter-cluster dissimilarity. A significant disadvantage of the DB index is that it does not have a fixed range, with an output value only constrained to be non-negative, making interpretation problematic. In addition, empirical analysis has shown that, when attempting to select k, this index tends to underestimate the number of groups, particularly for weakly clustered data (Dubes, 1987).

C-index

Hubert & Levin (1976) proposed a cluster validation measure that evaluates the homogeneity of a set of clusters by comparing the weight of the intra-cluster distances induced to a similar proportion of inter-cluster distances. Formally, let d_w denote the sum of all l_w intra-cluster distances induced by a clustering $\mathcal{C}$. Furthermore, let d_{min} denote the sum of the l_w smallest and d_{max} denote the sum of the l_w largest distances across all pairs of objects in the dataset. Having examined all pairwise distances, the *C-index* for a clustering is calculated as the ratio:

$$HL(\mathcal{C}) = \frac{d_w - d_{min}}{d_{max} - d_{min}} \tag{3.7}$$

A small value for this ratio is generally indicative of a more cohesive clustering. While the C-index has not been frequently employed in unstructured text clustering problems, Dalamagas *et al.* (2004) did evaluate its application in determining a suitable cut-off point for hierarchical clustering of XML documents.

Silhouette index

Rousseeuw (1987) suggested computing a "silhouette value" for each object in a clustering, which measures the degree to which the object belongs to its current cluster relative to the other $k-1$ clusters. Formally, for each $x_i \in C_a$, let $a(i)$ denote the average distance between the object and all other objects in C_a, and let $b(i)$ denote the average distance between x_i and all objects in the nearest competing cluster C_b:

$$a(i) = \frac{1}{|C_a|} \sum_{j \in C_a, i \neq j} d(i,j) \qquad\qquad b(i) = \frac{1}{|C_b|} \sum_{j \in C_b} d(i,j)$$

The silhouette width for x_i is then computed using the expression:

$$sil(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{3.8}$$

This produces an evaluation in the range $[-1, 1]$, indicating how well the object fits in its own cluster when compared to how well it would fit if moved to another cluster. A value close to 1 indicates that x_i is likely to have been assigned to the appropriate cluster, a silhouette closer to 0 suggests that x_i could also have been assigned to the nearest alternative cluster, while a negative value suggests that x_i is likely to have been incorrectly assigned. The latter case can also be interpreted to mean that the object is an outlier. An overall evaluation for a k-way clustering, the *average silhouette width*, can be computed by taking the mean silhouette of all n participating objects:

$$ASW(\mathcal{C}) = \frac{1}{n} \sum_{i=1}^{n} sil(i) \tag{3.9}$$

While the use of the silhouette index in identifying a suitable cut-off point for hierarchical document clustering was considered by Yao & Choi (2005), it has not been widely employed to validate clusterings of text corpora. When working with this type of data, an alternative formulation, based on cosine similarity values, may be more appropriate. In this case, $ca(i)$ denotes the average similarity between a document $x_i \in C_a$ and the other documents in that cluster, and $cb(i)$ denotes the average similarity between x_i and the documents in the next nearest cluster C_b:

$$ca(i) = \frac{1}{|C_a|} \sum_{j \in C_a, i \neq j} cos(i,j) \qquad cb(i) = \frac{1}{|C_b|} \sum_{j \in C_b} cos(i,j)$$

A silhouette width for the document x_i is now calculated using the expression:

$$csil(i) = \frac{2(ca(i) - cb(i))}{ca(i) + cb(i)} - 1 \tag{3.10}$$

To produce an overall evaluation for a clustering, we take the average across all n documents in a manner analogous to Eqn. 3.9:

$$CASW(\mathcal{C}) = \frac{1}{n} \sum_{i=1}^{n} csil(i) \tag{3.11}$$

Once again, a higher values signifies a superior clustering of the data.

3.2.2 Recent Internal Validation Techniques

While the tasks of cluster evaluation and model selection remain largely unsolved problems in document clustering, there has been relatively little progress in this area over recent years. Therefore, we now describe several general-purpose internal validation indices that have been proposed in recent literature. While not all of these techniques are applicable to text data, they are nonetheless interesting for the broader task of cluster validation.

Hybrid clustering criterion

Zhao & Karypis (2001) suggested a criterion specifically for use in the context of document clustering, which rewards partitions that represent a good trade-off between the standard goals of maximising intra-cluster cohesion while minimising inter-cluster associations. The authors suggest that the former can be achieved by maximising the sum of the average pairwise cosine similarities induced by a clustering $\mathcal{C}$, as quantified by:

$$I(\mathcal{C}) = \frac{1}{n} \sum_{c=1}^{k} \left(\frac{\sum_{x_i, x_j \in C_c} cos(x_i, x_j)}{|C_c|} \right)$$

The second goal can be attained by ensuring that a clustering clearly separates the documents in a cluster from the rest of the collection, specifically by producing a low value for the quantity:

$$E(\mathcal{C}) = \frac{1}{\sum_{c=1}^{k} |C_c| \cdot (n - |C_c|)} \sum_{c=1}^{k} \left(\sum_{x_i \in C_c, x_j \notin C_c} cos(x_i, x_j) \right)$$

A complete evaluation for a clustering $\mathcal{C}$ is produced considering both objectives simultaneously, using the ratio:

$$H(\mathcal{C}) = \frac{I(\mathcal{C})}{E(\mathcal{C})} \tag{3.12}$$

It is clear that a good clustering should produce a high value for this index, indicating strong associations between documents assigned to the same cluster and low associations between documents assigned to different clusters. While this criterion may be useful for comparing clusterings with equal numbers of clusters, it exhibits a strong bias toward larger values of k, making it unsuitable for estimating the number of topics in a document collection.

Geometric indices

Frederix & Pauwels (2004) proposed a number of non-parametric validation techniques which do not exhibit the strong bias toward spherical clusters that is present in a number of traditional indices. In contrast, the proposed indices are primarily based on the assumption that a cluster may take the form of an arbitrary geometric shape consisting of a dense region that is well-separated from the other clusters.

Motivated by the observation that regions forming cluster structures will normally be locally homogeneous, such that the majority of the set of neighbours of each object are likely to belong to the same cluster as that object, the authors propose an index referred to as *cluster tension.* This index examines the diversity of cluster assignments among the p nearest neighbours of each data object, while also asserting that diversity in high-density regions is more indicative of an incorrect clustering than diversity in less dense regions. This assertion is incorporated into the measure by weighting diversity by local density computed using a suitable density function $\phi(x)$. Formally, a global score for a clustering $\mathcal{C}$ is computed by taking the mean local tension over all n data objects

$$T(\mathcal{C}) = \frac{1}{n} \sum_{i=1}^{n} \delta_p(x_i)\phi(x_i) \tag{3.13}$$

where $\delta_p(x_i)$ denotes the fraction of the p nearest neighbours of x_i which have been assigned to a different cluster in $\mathcal{C}$. In practice, the expected value for Eqn. 3.13 is calculated by applying the index to a collection of randomly generated clusterings of the data. An estimated p-value is then used to assess the significance of the clustering evaluation, where a small value relative for Eqn. 3.13 relative to the random evaluation scores is indicative of a good clustering.

While the index described above will assess cluster compactness, Frederix & Pauwels (2004) proposed a second measure, the *connectivity index*, that evaluates the degree to which a clustering $\mathcal{C}$ achieves the alternative objective of "connectedness". Specifically, for any pair of objects assigned to the same cluster in $\mathcal{C}$, the density of the data along the path connecting the pair should be consistently high. On the other hand, a poor clustering may contain clusters with weakly connected sub-regions. This objective has previously been employed in so-called path-based clustering algorithms (*e.g.* Fischer & Buhmann, 2003). In practice, connectivity is evaluated by randomly selecting r pairs (x_i, x_j), such that the objects in each pair belong to the same cluster in $\mathcal{C}$. The density of the data at the midpoint of the path connecting each pair (x_i, x_j) is then computed. A total evaluation for the clustering is obtained by calculating the mean density at all r midpoints:

$$C(\mathcal{C}) = \frac{1}{r}\sum_{i=1}^{r} \phi(m_i) \tag{3.14}$$

where m_i denotes the midpoint for the i-th pair of objects. The authors suggest that the robustness of this index can be improved by computing the density at multiple points along each path.

While density-based and path-based cluster analysis procedures have not been generally applied to text data, the dual objectives of connectedness and local density are still desirable in the context of document clustering. We examine a novel validation technique that makes use of the latter concept in Section 7.2.

Bayesian Information Criterion

When using model-based clustering techniques, a popular approach for estimating the number of clusters has been to apply the Bayesian Information Criterion (BIC) (Schwarz, 1978). This criterion estimates the optimal number of mixture components for a dataset $\mathcal{X}$ containing n objects by computing an evaluation for the current model based on

$$BIC = L(\mathcal{X}) - \frac{r}{2}\log n \tag{3.15}$$

where r denotes the number of parameters in the model and $L(\mathcal{X})$ refers to its log-likelihood. The first term indicates how well the model fits the data in each cluster, while the second term penalises model complexity. Consequently, a larger value for Eqn. 3.15 indicates that the clustering model is more appropriate for the data.

When using the BIC criterion, it is generally assumed that the data consists of a mixture of Gaussian components, corresponding to k clusters (Pelleg & Moore, 2000). In this model, the maximum log-likelihood for a cluster C_j of size n_j is computed as

$$L(C_c) = -\frac{n_j}{2}\log(2\pi) - \frac{m \cdot n_j}{2}\log(\hat{\sigma}^2) - \frac{n_j - k}{2} + n_j \log(n_j) - n_j \log n$$

where the variance estimate $\hat{\sigma}^2$ is calculated in terms of Euclidean object-centroid distances:

$$\hat{\sigma}^2 = \frac{1}{n-k}\sum_i ||x_i - \mu_j||^2$$

The overall log-likelihood for a dataset is found by taking the sum over all k clusters:

$$L(\mathcal{X}) = \sum_{c=1}^{k} L(C_c)$$

The BIC technique has been used by a number of authors when employing model-based document clustering methods, including spherical EM clustering (Kruengkrai *et al.*, 2004). However, it has been observed that the criterion exhibits a bias toward larger values of k when the cluster structures in the data are not perfectly spherical (Hamerly & Elkan, 2004). Other similar probabilistic model selection techniques include the Akaike Information Criterion (AIC) (Akaike, 1974) and Minimum Description Length (MDL). It has been observed that these techniques can often be computationally expensive for high-dimensional data (Torre & Black, 2003).

Gap statistic

Tibshirani *et al.* (2000) proposed an alternative approach for estimating the number of clusters in data, which involves comparing the observed intra-cluster cluster dispersion with its expectation as the value of k increases. Formally, let W_k denote the total of the average intra-cluster pairwise squared distances for a clustering containing k groups:

$$W_k = \sum_{c=1}^{k} \frac{1}{|C_c|} \sum_{x,y \in C_c} d(x,y)^2 \qquad (3.16)$$

The *Gap statistic* compares the observed value of $\log(W_k)$ with its expected value under an appropriate null reference distribution. In practice, this comparison is performed by firstly generating b reference datasets, typically from a uniform distribution. To evaluate the clustering model for a specific value of k, a clustering is generated on each reference set and the average value of $\log(W_k)$ over the b runs is computed. The difference between the actual value for Eqn. 3.16 and its approximated expectation is then examined:

$$GAP(k) = \frac{1}{b}\left[\sum_{h=1}^{b} \log W_{kb}\right] - \log W_k \tag{3.17}$$

A final estimation for $\hat{k}$ is obtained by selecting the smallest value k such that the quantity $GAP(k) - GAP(k+1)$ is above a given threshold value.

An advantage of the gap statistic derives from its ability to produce an evaluation for the case of $k = 1$, which is useful when examining the issue of cluster tendency. However, its reliance on the within-cluster squared distance criterion given in Eqn. 3.16 leads it to favour compact clusters. In addition, the problem of creating a suitable reference distribution is non-trivial, particularly for high-dimensional data (Ben-Hur *et al.*, 2002). We are not aware of any application of this technique that involves text data.

3.3 External Validation

A significant disadvantage of internal techniques is that useful comparisons may only be made between clusterings that are generated using the same data model and similarity metric (Ghosh, 2003). We have also seen that many well-known internal indices make assumptions about the structure of the clusters in data, so that they favour clusters with certain geometric properties.

An alternative approach to validation is to apply the algorithm to a dataset for which a reference partition or "ground truth" is available, typically in the form of predefined class labels. External validation indices make use of this information, unavailable to the clustering algorithm itself, to quantify the level of agreement between the algorithm's output and the set of k' natural classes $\mathcal{C}' = \{C'_1, \ldots, C'_{k'}\}$ in a reference partition. Since these indices generally only consider the final partition of the data, they are independent of the representation and metrics used during the clustering procedure. In this section we provide a comprehensive review of external validation indices that are suitable for evaluating disjoint clusterings. When describing these indices, we let n'_i denote the number of objects in class C'_i, let n_j denote the number of objects in cluster C_j, and let n_{ij} denote the number of objects common to both the class C'_i and cluster C_j

3.3.1 Set Matching Measures

A simple external validation approach is to identify a match between each cluster and a corresponding natural class in the reference partition. Once a mapping has been found, evaluations can be readily computed based on the $k' \times k$ *confusion matrix* $\mathbf{N}$, where the entry N_{ij} denotes the size of the intersection $|C'_i \cap C_j|$ between the class C'_i and cluster C_j. An example demonstrating the construction of a confusion matrix is shown in Figure 3.2. It is clear that, by examining the entries in the 3×3 matrix, we can find the optimal permutation π of the groups in $\mathcal{C}$ relative to those in $\mathcal{C}'$.

$$C' = \{\{x_1,x_2\},\{x_3,x_4,x_5\},\{x_6,x_7,x_8\}\}$$
$$C = \{\{x_1,x_2,x_3\},\{x_6,x_7\}\{x_4,x_5,x_8\}\}$$

$$\longrightarrow$$

Class	C_1	C_2	C_3
C'_1	2	0	0
C'_2	1	0	2
C'_3	0	2	1

$$\longrightarrow$$

$$\pi\{C_1,C_2,C_3\} = \{C'_1,C'_2,C'_3\}$$

Figure 3.2: Example of a simple matching procedure to align the groups in a correct classification $\mathcal{C}'$ of eight data objects with those in a clustering $\mathcal{C}$ of the same data.

Classification accuracy

Motivated by conventional evaluation techniques in supervised learning, several authors have suggested assessing the quality of a partition by assigning a unique dominant natural class to each cluster and counting the number of objects that have been assigned to the correct cluster (Meila, 2002). To do this, a heuristic correspondence procedure is applied, which first identifies the largest intersection N_{ij}, resulting in a match between C'_i and C_j. The next match is chosen based on the highest value N_{ij} from the remaining pairs, with the procedure continuing until $min(k', k)$ matches have been found. Note that no class may be matched to more than one cluster. The *classification accuracy* for the clustering $\mathcal{C}$ is then calculated using the expression

$$H(\mathcal{C}',\mathcal{C}) = \frac{1}{n} \sum_{j'=match(j)} N_{jj'} \tag{3.18}$$

where $match(j)$ denotes the index of the class selected as a match for the cluster C_j.

Purity

Zhao & Karypis (2001) suggested measuring the extent to which each cluster contains objects from a single dominant natural class. The *purity* of a cluster C_j is defined as the fraction of objects in the cluster that belong to the dominant class contained within that cluster:

$$P(C'_i, C_j) = \frac{1}{n_j} \max_i\{N_{ij}\} \tag{3.19}$$

Unlike the classification accuracy measure, purity allows multiple clusters to be matched to the same dominant class. The overall purity of a clustering is defined as the sum of the individual cluster purities, weighted by the size of each cluster:

$$P(\mathcal{C}', \mathcal{C}) = \sum_{j=1}^{k} \frac{n_j}{n} P(C', C_j) \tag{3.20}$$

This measure provides a naïve estimate of partition quality, where larger purity values are intended to indicate a better clustering. However, as noted by Strehl & Ghosh (2002a), the index favours small clusters, with the degenerate case of a singleton cluster resulting in a maximal cluster purity score.

F-Measure

The *F-Measure* (Larsen & Aone, 1999) is based on the *recall* and *precision* criteria that are commonly used in information retrieval tasks. Each cluster is viewed as the result of a query operation, and each natural class is viewed as the target set of documents for the query. In the ideal case, each cluster will directly correspond to a natural class. Using our notation, precision and recall for a class C'_i and cluster C_j are defined respectively as:

$$p(C_j, C'_i) = \frac{N_{ij}}{n_j} \qquad r(C_j, C'_i) = \frac{N_{ij}}{n'_i}$$

High precision implies that most objects in a given cluster belong to the same class, while high recall suggests that most objects from a single class were assigned to the same cluster. The F-measure for a pair (C'_i, C_j) is given by the harmonic mean of their precision and recall, calculated as:

$$F_{ij} = \frac{2 \cdot r_{ij} \cdot p_{ij}}{p_{ij} + r_{ij}} \tag{3.21}$$

For each class C'_i, a unique matching cluster C_j is selected so as to maximise the value F_{ij}. An overall score for a clustering $\mathcal{C}$ is obtained by taking the weighted average of the maximum F-values across all k' classes:

$$F(\mathcal{C}', \mathcal{C}) = \sum_{i=1}^{k'} \frac{n'_i}{n} \max_j \{F_{ij}\} \tag{3.22}$$

Ghosh (2003) observed that this index has a tendency to favour clusterings containing a small number of clusters.

Partition distance

Gusfield (2002) defined the *partition distance* between two clusterings, denoted $D(\mathcal{C}', \mathcal{C})$, as the minimum number of objects that must be reassigned in $\mathcal{C}$ to produce a solution identical to $\mathcal{C}'$. A maximum value for this measure is achieved in the trivial case where one clustering contains a single cluster and the other contains only singleton clusters. To address this, a normalised distance function producing values in the range $[0, 1]$ was proposed, which is given by:

$$ND(\mathcal{C}', \mathcal{C}) = \frac{D(\mathcal{C}', \mathcal{C})}{(n-1)} \tag{3.23}$$

To efficiently identify the optimal correspondence between groups in both classifications, the authors suggested using the well-known Hungarian method (Kuhn, 1955).

3.3.2 Pairwise Co-assignment Measures

An alternative approach to external validation is to count the pairs of objects for which the clusters and natural classes agree on their co-assignment. By considering all pairs, we can calculate statistics for each of four possible cases:

- a = number of pairs in the same class in $\mathcal{C}'$ and assigned to the same cluster in $\mathcal{C}$.
- b = number of pairs in the same class in $\mathcal{C}'$, but in different clusters in $\mathcal{C}$.
- c = number of pairs assigned to the same cluster in $\mathcal{C}$, but in different classes in $\mathcal{C}'$.
- d = number of pairs belonging to different classes in $\mathcal{C}'$ and assigned to different clusters in $\mathcal{C}$.

Note that $a+d$ corresponds to the number of agreements between $\mathcal{C}'$ and $\mathcal{C}$, $b+c$ corresponds to the disagreements, and $M = a+b+c+d = \frac{n(n-1)}{2}$ is the total number of unique pairs. For instance, in the example shown in Figure 3.2, examining the co-assignments for the 28 unique pairs results in the values $a = 3$, $b = 4$, $c = 4$, $d = 17$.

Jaccard index

The *Jaccard coefficient* (Jaccard, 1912) has been commonly applied to assess the similarity between binary sets. It is also possible for this measure to be used in the context of external validation, where the level of agreement of between the disjoint partitions $\mathcal{C}'$ and $\mathcal{C}$ is given by normalising the number of positive agreements:

$$J(\mathcal{C}', \mathcal{C}) = \frac{a}{a+b+c} \tag{3.24}$$

This index produces a result in the range $[0, 1]$, where a value of 1 indicates that $\mathcal{C}'$ and $\mathcal{C}$ are identical. Dubes (1987) observed that Eqn. 3.24 tends to produce high values for random clusterings and favours lower values of k.

Rand index

The *Rand index* (Rand, 1971) is similar to the above measure, but also considers cases where both partitions assign a pair of objects to different groups. This results in an evaluation in the range $[0, 1]$ based on the fraction of pairs for which there is an agreement:

$$R(\mathcal{C}', \mathcal{C}) = \frac{a+d}{a+b+c+d} \tag{3.25}$$

To eliminate biases related to different cluster size distributions and the number of clusters, Hubert & Arabie (1985) proposed the *corrected Rand index*, which is computed as follows:

$$CR(\mathcal{C}', \mathcal{C}) = \frac{2(ad-bc)}{(a+b)(b+d)+(a+c)(c+d)} \tag{3.26}$$

After applying this correction, a value of 1 indicates a perfect agreement between the two groupings, while a value of 0 indicates that a clustering is no better than a random partitioning of the data.

Fowlkes-Mallows index

A popular index for assessing the similarity between partitions was proposed by Fowlkes & Mallow (1983), which is based on the calculation of two probability scores: the probability that a pair of objects are assigned to the same cluster given that they belong to the same class, and the probability that a pair objects belong to the same class given that they were assigned to the same cluster. A value for the *Fowlkes-Mallows index* (FM) is found by taking the geometric mean of these probabilities:

$$FM(\mathcal{C}', \mathcal{C}) = \sqrt{\left(\frac{a}{a+b}\right)\left(\frac{a}{a+c}\right)} \tag{3.27}$$

A value close to 1 indicates that the clusters in $\mathcal{C}$ provide a good estimate for the reference partition.

3.3.3 Information Theoretic Measures

Recent research relating to cluster validation has focused on concepts from information theory, which consider the uncertainty of predicting a set of natural classes based on the information provided by a clustering of the same data. We now describe two indices, based on these concepts, which have frequently been applied to evaluate clusterings of text data.

Entropy index

Steinbach *et al.* (2000) suggested an entropy-based measure for assessing the agreement between two partitions. By considering the probability $\frac{N_{ij}}{n_j}$ that an object assigned to cluster C_j belongs to a class C'_i, we can compute the entropy for the assignments in C_j:

$$E(C_j) = -\sum_{i=1}^{k'} \frac{N_{ij}}{n_j} \log \frac{N_{ij}}{n_j} \tag{3.28}$$

An overall score for a clustering $\mathcal{C}$ is given by the sum of the entropy values for each cluster weighted by the fraction of objects assigned to that cluster:

$$E(\mathcal{C}', \mathcal{C}) = \sum_{j=1}^{k} \frac{n_j}{n} E(C_j) \tag{3.29}$$

Smaller values for this measure are desirable, with a value of 0 indicating that each cluster contains instances from a single class. To eliminate the strong bias of Eqn. 3.29 with respect to k, a variant of this index was proposed by Zhao & Karypis (2001), where the normalised entropy for a cluster C_j is calculated as:

$$NE(C_j) = -\frac{1}{\log k'} \sum_{i=1}^{k'} \frac{N_{ij}}{n_j} \log \frac{N_{ij}}{n_j} \tag{3.30}$$

Unlike purity and classification accuracy, entropy considers the distribution of all classes in a cluster, rather than a single dominant class. However, this index still exhibits a bias in favour of smaller clusters.

Normalised Mutual Information

Strehl & Ghosh (2002a) observed that external measures such as purity and entropy are biased with respect to the number of clusters k, since the probability of each cluster solely containing objects from a single natural class increases as k increases. To address this problem, an alternative index was proposed, based on *mutual information*, which quantifies the amount of information shared between the random variables describing a pair of disjoint partitions.

Formally, let $p'(i)$ and $p(j)$ denote the probabilities that an object belongs to class C'_i and cluster C_j respectively. Furthermore, let $p(i,j)$ denote the joint probability that an object belongs to both C'_i and C_j. For each data object assigned to a class in $\mathcal{C}'$, mutual information evaluates the degree to which knowledge of this assignment reduces the uncertainty regarding the assignment of the object in $\mathcal{C}$. The mean reduction in uncertainty across all objects can be expressed as:

$$I(\mathcal{C}',\mathcal{C}) = \sum_{i=1}^{k'} \sum_{j=1}^{k} p(i,j) \log \frac{p(i,j)}{p'(i)p(j)} \tag{3.31}$$

$I(\mathcal{C}',\mathcal{C})$ takes values between zero and $\min\left(E(\mathcal{C}'), E(\mathcal{C})\right)$, where the upper bound is the minimum of the entropy values for the two clusterings. To produce values in the range $[0,1]$, Strehl & Ghosh (2002a) defined *normalised mutual information* (NMI), where the mutual information between the two clusterings is normalised with respect to the geometric mean of their entropies:

$$NI(\mathcal{C}',\mathcal{C}) = \frac{I(\mathcal{C}',\mathcal{C})}{\sqrt{E(\mathcal{C}')E(\mathcal{C})}} \tag{3.32}$$

In practice, an approximation for this quantity, based on cluster assignments, can be calculated using:

$$NMI(\mathcal{C}',\mathcal{C}) = \frac{\sum_{i=1}^{k'} \sum_{j=1}^{k} n_{ij} \log \left(\frac{n \cdot n_{ij}}{n'_i n_j}\right)}{\sqrt{\left(\sum_{i=1}^{k'} n'_i \log \frac{n'_i}{n}\right) \left(\sum_{j=1}^{k} n_j \log \frac{n_j}{n}\right)}} \tag{3.33}$$

An accurate clustering should maximise this score, where a value of 1 indicates an exact correspondence between the assignment of objects in $\mathcal{C}'$ and $\mathcal{C}$, while a value of 0 indicates that knowledge of $\mathcal{C}$ provides no information about the true classes $\mathcal{C}'$. Eqn. 3.33 does have a slight tendency to favour clusterings for larger values of k, although it exhibits no bias against unbalanced cluster sizes.

3.4 Stability-Based Validation

Recently, a number of methods based on the concept of *stability analysis* have been proposed for the task of model selection. The *stability* of a clustering algorithm refers to its ability to consistently produce similar solutions on data originating from the same source (Lange *et al.*, 2004). Since only a single set of data objects will be generally available in unsupervised learning tasks, clusterings are generated on perturbations of the original dataset. A key advantage of stability analysis methods lies in their ability to evaluate a model independently of any specific clustering algorithm or similarity measure. Thus, they represent a robust approach for selecting key algorithm parameters (Law & Jain, 2003).

In this section, we focus on stability-based methods that are relevant when estimating the optimal number of clusters $\hat{k}$ in a dataset. These methods are motivated by the observation that, if the number of clusters in a model is too large, repeated clusterings will lead to arbitrary partitions of the data, resulting in unstable solutions. On the other hand, if the number of clusters is too small, the clustering algorithm will be constrained to merge subsets of objects which should remain separated, also leading to unstable solutions. In contrast, repeated clusterings generated using the optimal number of clusters $\hat{k}$ will generally be consistent, even when the data is perturbed or distorted.

3.4.1 Stability Analysis Based on Resampling

The most common approach to stability analysis involves perturbing the data by randomly sampling the original objects to produce a set of τ non-disjoint subsets. For each potential value of k in a reasonable range $[k_{min}, k_{max}]$, a corresponding set of τ clusterings are generated on the data subsets. The stability of the clustering model for each candidate value of k is evaluated using indices operating on pairs of hard clusterings, such as the external validation indices described in Section 3.3. A higher overall stability score suggests that k is a better estimate for the optimal value $\hat{k}$.

A representative example of this approach is the algorithm proposed by Levine & Domany (2001). For each value of k, an initial partition $\mathcal{C}_0$ is generated on the entire dataset using a partitional clustering algorithm, which represents a "gold standard" for analysing the stability afforded by using k clusters. Subsequently, τ samples of the data are constructed by randomly selecting a subset of βn data objects without replacement, where $0 \leq \beta \leq 1$ denotes the sampling ratio controlling the number of objects in each sample. A set of clusterings $\{\mathcal{C}_1, \ldots, \mathcal{C}_\tau\}$ is then generated by applying the clustering algorithm to each sample. For each clustering $\mathcal{C}_i$, the fraction of co-assignments preserved from $\mathcal{C}_0$ is calculated, which is equivalent to the Rand index (3.25). An overall evaluation for the stability afforded by k is found by averaging the agreement scores across all τ runs. This process is repeated for each potential $k \in [k_{min}, k_{max}]$. A final estimation for $\hat{k}$ is chosen by identifying the value k leading to the highest average agreement.

Law & Jain (2003) proposed an alternative stability analysis approach for model selection where the data is perturbed by bootstrapping. This involves generating τ samples of size n by randomly sampling with replacement. Rather than comparing each clustering to a single gold standard solution, stability is evaluated by considering the level of agreement between each pair of clusterings. A number of indices were considered for assessing agreement, including the Jaccard index (3.24) and the Fowlkes-Mallows index (3.27). The authors note that scores produced by these indices should be corrected for chance to eliminate biases toward smaller values of k. After computing the variance of the corrected agreement scores for each potential value k, the model resulting in the lowest variance is selected as the best estimate for $\hat{k}$.

Ben-Hur *et al.* (2002) described a similar approach based on pairwise stability analysis, where agglomerative hierarchical clustering is applied to each sample. By using different cut-off levels from the same hierarchy, the output of a single clustering procedure may be used in the evaluation of all potential values of k. Giurcaneanu & Tabus (2004) extended this approach further to encompass the problem of cluster tendency. This is achieved by setting a threshold θ value for the minimum average pairwise stability that is sufficient to indicate a consistent clustering model. If no stability evaluation exceeds this threshold for any candidate $k \in [k_{min}, k_{max}]$, the data is assumed to have no significant underlying structure. The choice of θ largely depends on the index used to measure the agreement between clusterings.

3.4.2 Prediction-Based Validation

In supervised learning problems, model selection is typically performed by identifying a learning model whose estimated prediction accuracy is highest. A number of authors have suggested that the concept of prediction accuracy can be adapted to the problem of evaluating models in clustering tasks. Recent work by Tibshirani *et al.* (2001) has provided a theoretical basis for *prediction-based validation* methods, which assess the stability of a clustering model by measuring the degree to which it allows us to consistently construct a classifier on a training set that will predict the assignment of objects in a clustering of a corresponding test set.

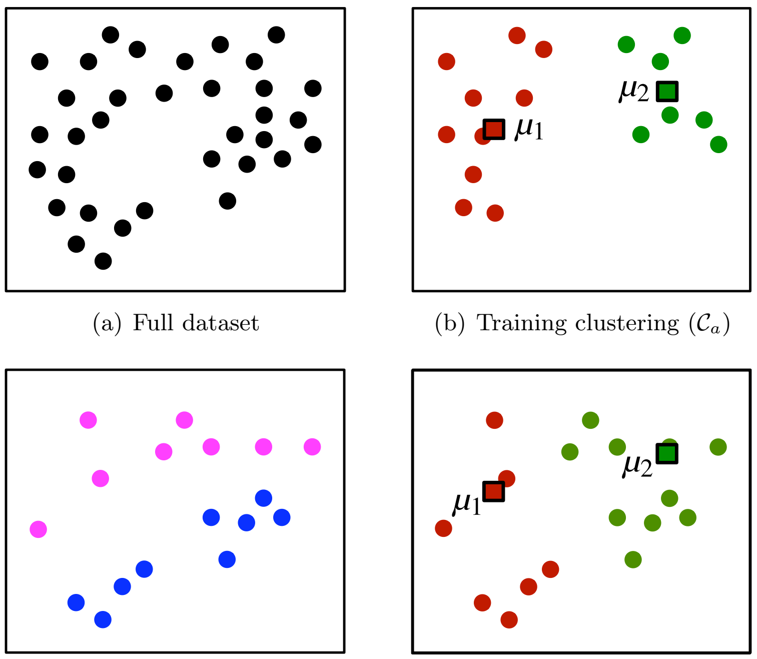
Formally, the validation process involves applying two-fold cross-validation to randomly split a dataset $\mathcal{X}$ into disjoint training and test sets, denoted by $\mathcal{X}_a$ and $\mathcal{X}_b$ respectively. Both sets are then clustered to produce partitions $\mathcal{C}_a$ and $\mathcal{C}_b$, typically using the standard k-means clustering algorithm. Subsequently, a prediction $\mathcal{P}_b$ for the assignment of objects in the test set is produced by assigning each $x_i \in \mathcal{X}_b$ to the nearest centroid in $\mathcal{C}_a$. Prediction accuracy is measured by evaluating the degree to which the class memberships in $\mathcal{P}_b$ correspond to the cluster assignments in $\mathcal{C}_b$.

To numerically evaluate prediction accuracy, Tibshirani *et al.* (2001) proposed a new pairwise measure for comparing partitions, referred to as *prediction strength*. For each cluster in the test clustering $\mathcal{C}_b = \{C_1, \ldots, C_k\}$, we identify the number of pairs of objects assigned to the same cluster that also belong to the same class in the prediction $\mathcal{P}_b$. These associations can be represented as a $\frac{n}{2} \times \frac{n}{2}$ binary matrix $\mathbf{M}$, where $M_{ij} = 1$ only if the pair (x_i,x_j) are co-assigned in both $\mathcal{C}_b$ and $\mathcal{P}_b$. From this matrix, an evaluation is computed based on the cluster containing the smallest fraction of correctly predicted pairs:

$$S(\mathcal{C}_b, \mathcal{P}_b) = \min_{1 \leq h \leq k} \left[\frac{1}{|C_h|\left(|C_h| - 1\right)} \sum_{x_i \neq x_j \in C_h} M_{ij} \right] \tag{3.34}$$

The cross-validation process is repeated over τ runs for each candidate value k in the range $[k_{min}, k_{max}]$. The authors suggest a heuristic approach to select the final number of clusters, which is chosen to be the largest k such that $ps(k)$ is above a user-defined threshold. This can be viewed as the selection of the largest number clusters that can be reliably predicted for a given dataset. They note that a threshold in the range $[0.8, 0.9]$ was appropriate for the datasets with which they evaluated the algorithm.

As a simple example, we consider a single cross-validation run for $k = 2$ applied to the set of 34 data objects shown in Figure 3.3(a). This dataset is randomly divided into two subsets containing 17 objects each. A training clustering $\mathcal{C}_a$ is generated on the first subset and a test clustering $\mathcal{C}_b$ is generated on the second, as shown in figures 3.3(b) and 3.3(c) respectively. The centroids μ_1 and μ_2 of $\mathcal{C}_a$ are subsequently used to build a nearest centroid classifier, which produces the predicted classification $\mathcal{P}_b$ for the set of test objects as illustrated in Figure 3.3(d). By constructing a 17×17 co-assignment matrix $\mathbf{M}$ from $\mathcal{C}_b$ and $\mathcal{P}_b$, and applying Eqn. 3.34, we can calculate that this run leads to a prediction



(a) Full dataset (b) Training clustering ($\mathcal{C}_a$)

(c) Test clustering ($\mathcal{C}_b$) (d) Predicted clustering ($\mathcal{P}_b$)

Figure 3.3: Example of applying prediction-based validation to examine the suitability of a clustering model with $k = 2$ for a synthetic dataset of 34 data objects.

strength of $S(\mathcal{C}_b, \mathcal{P}_b) = 0.43$, indicating that the clustering model is relatively unstable. In practice, multiple cross-validation runs would be applied to produce a result that is robust to the effects of unbiased random sampling.

A similar approach for selecting the number of clusters was described by Roth *et al.* (2002), which also makes use of two-fold cross validation. However, prediction accuracy was measured by finding the optimal correspondence of the clusters in the test clustering with the predicted clusters using the Hungarian method (Kuhn, 1955) and computing the number of incorrect assignment predictions. An evaluation for the instability of a clustering model is found by averaging the number of incorrect assignments over τ iterations. Another variation of this approach was discussed by Giurcaneanu & Tabus (2004), who suggested evaluating prediction accuracy using a measure related to the partition distance index defined in Eqn. 3.23. Specifically, the *partition similarity* between a test clustering $\mathcal{C}_b$ and a corresponding prediction $\mathcal{P}_b$ is equivalent to the inverse of the normalised partition distance between the two groupings:

$$PSIM(\mathcal{C}_b, \mathcal{P}_b) = 1 - \frac{D(\mathcal{C}_b, \mathcal{P}_b)}{(n-1)} \tag{3.35}$$

A higher mean value for this index across τ runs indicates that the clustering model under consideration affords a greater degree of stability.

3.5 Summary

In this chapter, we surveyed a range of approaches for assessing the validity of clustering solutions, and the models used to produce them. External validation methods have been widely used in the document clustering literature for evaluating the ability of novel clustering methods to uncover the natural classes in pre-classified corpora. Hence, in the remainder of this thesis we make extensive use of external indices when comparing the relative performance of clustering algorithms. It is important to note that *a priori* class information will generally be inaccessible in real unsupervised tasks, making external validation methods inappropriate for problems such as model selection. As an alternative, internal validation indices have frequently been employed for this task in the past. Unfortunately, internal indices often exhibit biases towards clusters with certain characteristics. Recent work has shown model selection methods based on stability analysis to be superior in domains such as bioinformatics (Dudoit & Fridlyand, 2002). However, the computational cost of generating and analysing many clusterings can be prohibitive for large, high-dimensional datasets. Consequently, these methods have rarely been applied by researchers working with text data. We examine this issue in detail in Section 7.2 and propose a new strategy for greatly enhancing the scalability of stability analysis methods.

Chapter 4

Text Clustering Toolkit

4.1 Introduction

For researchers interested in the exploration of unstructured text datasets, the absence of a comprehensive toolkit providing access to state-of-the-art algorithms represents a significant obstacle. Motivated by this problem, we have developed the *Text Clustering Toolkit* (TCT)[1], a new Java-based framework for document clustering, which provides researchers with the ability to compare and extend many popular cluster analysis procedures. In this chapter, we describe the architectural design and functionality of TCT, and provide practical details regarding its implementation and usage.

4.1.1 Design Considerations

Unlike existing machine learning toolkits, many of which are limited to the demonstration of specific types of learning algorithm, we have aimed to create a framework for unsupervised learning that is modular and extensible. In particular, our goal has been to provide researchers with a test-bed to facilitate both the evaluation of existing techniques and the development of novel analysis procedures. Many clustering techniques share common characteristics, such as the iterative reassignment process common to partitional clustering algorithms. We have tried to ensure that implementations of frequently employed routines are readily available to serve as "building blocks" for new algorithms. By abstracting away these lower-level procedures, TCT allows researchers to concentrate on more sophisticated aspects of their algorithms.

[1]Available from `http://mlg.ucd.ie/tct`

With TCT, we aimed to produce a set of loosely coupled libraries suitable for a variety of tasks, without significantly compromising on efficiency or scalability. While our current focus is on the analysis of text data, we envisage that TCT will provide a basis for unsupervised learning applications in other areas, such as bioinformatics and image processing.

4.2 Toolkit Structure

In this section, we describe the overall architecture of TCT, and highlight important aspects of the functionality of the toolkit.

4.2.1 Architecture

The architecture of the toolkit reflects the considerations described in the previous section. To support the development of a wide range of machine learning applications, TCT is designed around a layer-based architecture, containing generic components that can be easily extended and customised for problem-specific tasks. With regard to the actual structure of the toolkit, it may be divided into three distinct layers as shown in Figure 4.1. These layers correspond to three separate libraries: the matrix library, the core learning library, and the document clustering library.

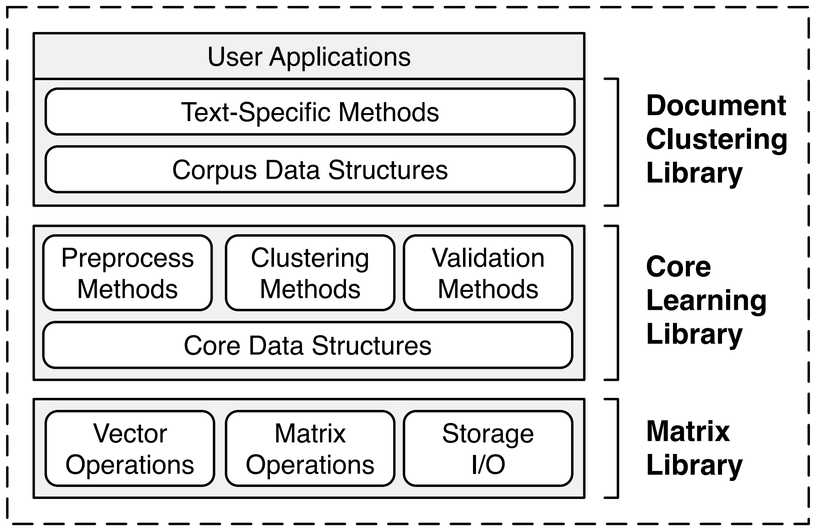


Figure 4.1: Design of the layer-based architecture for the Text Clustering Toolkit (TCT).

Matrix Library

Many unsupervised learning algorithms can be reduced to a series of matrix and vector operations. Thus, the foundation layer of the toolkit stack consists of a library providing essential linear algebra operations. Feature spaces, graphs, and other data models may be represented as matrices and manipulated by this library. TCT includes support for both dense and sparse matrices, which are useful for representing different types of data. In these cases, the matrix library supports functionality that is comparable to that provided by commercial offerings such as the MATLAB environment[2]. This includes operations for EVD and SVD computation, sparse and dense matrix multiplication routines, and functions for generating a variety of statistics describing a given vector or matrix.

Core Learning Library

The central layer of the stack structure consists of two parts. Firstly, we provide a range of key data structures relevant to various clustering tasks. This includes generic structures for representing data in feature spaces, and models for hard, soft and co-clusterings. The second part of this layer consists of a large set of modular components, providing implementations for many of the algorithms discussed in this thesis. Conceptually, the set of component implementations in the learning library may be sub-divided into groups of methods corresponding to the three phases of the clustering workflow originally shown in Figure 1.2: preprocessing, clustering and validation.

Document Clustering Library

At the top of layer stack, we have built a library specific to our area of interest, comprising of data structures and algorithms relevant to the task of document clustering. The former covers implementations for models commonly used to represent sparse text corpora, while the latter includes algorithms specific to text mining tasks, such as term weighting, term selection and cluster labelling methods. Custom applications may subsequently be built upon the document clustering library to provide users with access to toolkit functionality. It is worth noting that this library could be interchanged for an alternative implementation when working in other domains.

[2]`http://www.mathworks.com/products/matlab/`

4.2.2 Functionality

Data Formats

For matrix storage, we support dense, whitespace-separated matrix storage with row labels, coordinate-based storage compatible with the standard Matrix Market exchange format[3] for sparse matrices, and a variety of character delimited formats compatible with MATLAB and Microsoft Excel. We build upon these basic formats to provide storage for both generic, low-dimensional datasets and sparse, high-dimensional document collections. In the latter case, the original term strings are available for use in the generation of cluster labels.

Repository

While TCT can make use of stand-alone files, to provide ready access to datasets when performing experimental evaluations, the document clustering layer of the toolkit includes support for storing corpora in a local repository. In addition to providing access to various aspects of the data (such as raw documents or terms), we also provide the facility to save newly generated clusterings or reduced representations to the repository, which can be retrieved for use at a later time.

Data Preprocessing Components

For preprocessing document collections, we provide a standard "bag of words" parser, which supports both word and n-gram tokenisation. Extremely rare or frequent terms may be removed at this stage to reduce the size of the corpus vocabulary. After an initial representation of the data model is produced, a variety of term weighting functions may be applied corresponding to those originally described by Salton & Buckley (1987). In addition, many state-of-the art methods are available to tackle issues resulting from high dimensionality, including many of the feature selection and extraction strategies described in Chapter 2. For the construction of affinity and kernel matrices, we provide implementations for a range of similarity metrics and kernel functions. A full list of currently supported preprocessing components is given in Table 4.3.

[3]See `http://math.nist.gov/MatrixMarket` for format specifications.

Category	**Components**
Parsing	• Word-based bag of words parser • N-gram bag of words parser
Normalisation	• *Tf-idf* normalisation (and variants) • Document length normalisation
Feature selection	• Mean *tf-idf* • Term variance quality • Term contribution
Feature extraction	• Principal Component Analysis • K-dimensional spectral embedding • Bipartite spectral embedding
Similarity measures	• Euclidean distance • Minkowski distance • Cosine similarity/distance • Jaccard similarity
Kernel functions	• Normalised linear kernel • Gaussian RBF kernel • Polynomial kernel

Table 4.1: A list of standard preprocessing methods supported by TCT.

Clustering Components

TCT contains implementations for a wide variety of classical algorithms, such as k-means and agglomerative hierarchical clustering, together with recent techniques based on spectral decomposition, non-negative matrix factorisation and ensemble clustering. While we have primarily used these implementations in conjunction with text data, the majority of these algorithms are also relevant to other types of data. Table 4.2 provides a complete list of supported algorithms. Note that, in addition to those listed here, implementations of the novel methods proposed later in chapters 6 and 7 are also included in the toolkit.

Validation Components

Since one of the primary aims of this toolkit is to provide a test-bed for the design and development of new clustering methods, we have paid particular attention to providing a range of components for assessing the output of clustering algorithms and for guiding parameter selection procedures. For benchmark comparisons, a number of external validation indices are provided, including all of those discussed in Section 3.3. For model selection, both internal indices and stability-based validation methods are available. A

Category	**Components**
Hierarchical algorithms	• Agglomerative (single, complete, average linkage) • Min-max agglomerative clustering • Bisecting k-means • Principal Direction Divisive Partitioning
Partitional algorithms	• Standard/generalised k-means • Spherical k-means • Fuzzy c-means
Matrix decomposition	• NJW spectral clustering • Bipartite spectral co-clustering • Euclidean NMF clustering • KL divergence NMF clustering • Symmetric NMF
Kernel clustering	• Kernel k-means • Weighted kernel k-means • Kernel bisecting k-means
Ensemble clustering	• Subsampling-based generation • Random parameter value generation • Initialisation-based generation • Co-association integration • Hypergraph integration • Bipartite integration • Correspondence-based integration

Table 4.2: A list of standard clustering methods supported by TCT.

full list of supported validation methods is given in Table 4.3.

4.3 Implementation and Use

In this section we provide details regarding the practical implementation of the TCT framework, and describe several ways in which the toolkit may be used by researchers.

4.3.1 Implementation Details

To provide cross-platform compatibility, TCT was developed using Sun Java (version 1.5) on Ubuntu Linux 6 (Intel x86). In addition, it has been tested on Windows XP and Apple OS 10.4. The key data structures in the matrix and core learning layers take advantage of new features in Java 1.5, including generics and greater type safety. When implementing TCT, we found that the object-oriented nature of the language lends itself naturally to the modular, component-based architecture of the toolkit.

As discussed previously, the issue of scalability is highly important when performing

Category	**Components**
Internal indices	• CH-index • C-index • Generalised Dunn's index • Generalised DB index • Silhouette index • Bayesian information criterion
External indices	• Purity index • Class accuracy • Rand index • Corrected Rand index • Jaccard index • Fowlkes-Mallows index • F-measure • Partition similarity/distance • Normalised entropy • Normalised mutual information
Stability analysis	• Pairwise stability analysis • Prediction-based analysis • Correspondence-based analysis

Table 4.3: A list of standard validation methods supported by TCT.

text mining tasks. To improve efficiency, many of the clustering and validation methods in TCT have been implemented in a way that takes advantage of the sparse matrix representation used for corpus storage. For instance, calculating similarity values or computing cluster centroids can be made significantly more efficient by only iterating over the non-zero entries in a sparse matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, rather than examining all $m \cdot n$ entries. Similar improvements may also be made when applying spectral dimension reduction methods. To this end, we include a Java Native Interface (JNI) wrapper for ARPACK (Lehoucq *et al.*, 1997), a standard library of FORTRAN routines for computing a small number of eigenvectors or singular vectors for a matrix. These routines are particularly efficient when employed to sparse matrices, making them well suited to the task of decomposing sparse corpus representations.

4.3.2 User Tools

Our primary focus has been on the production of an underlying framework that may be extended by developers and researchers alike. However, to provide direct access to the functionality of the toolkit, the TCT distribution comes with a set of command-line applications covering all aspects of the cluster analysis process. The applications included are as follows:

`parse-vsm`: Employs a standard word-based bag of words parser to produce a vector space model from a raw collection of unstructured text documents.

`parse-ngram`: Employs an n-gram bag of words parser to process a raw collection of unstructured text documents.

`cluster-hier`: A tool for comparing agglomerative and divisive document clustering algorithms, including support for a variety of linkage strategies for the former and several representative algorithms for the latter.

`cluster-part`: A tool for experimenting with popular partitional clustering algorithms that produce hard and soft clusterings.

`cluster-decomp`: Supports the comparison of a range of previously proposed and novel clustering methods based on matrix decomposition, which are suitable for text data.

`cluster-kernel`: A tool for comparing various kernel clustering methods, and examining the effects of applying different kernel functions on text data. In addition, this application provides access to a range of diagonal dominance reductions strategies (these strategies are described in detail in Section 6.3).

`cluster-ensemble`: Provides access to ensemble techniques for document clustering. As noted in Section 2.9, there are a number of important design decision that must be made when employing these techniques. Therefore, this tool allows users to experiment with different ensemble generation strategies, integration methods and parameter values.

`cluster-label`: A tool for comparing different approaches for generating interpretable labels for clusterings of text data, based on the techniques described later in Section 6.2.

A sample of the terminal output produced by the `cluster-label` tool when applied to the *bbcsport* dataset is shown in Figure 4.2. These command-line applications were used to produce the experimental results presented throughout this thesis.

```
=== cluster-label : 'bbcsport' ========================
Loaded corpus (Documents=737 Features=4613)
Loaded natural classes (K=5)
>> Pre-processing
Applying TfIdfNormalizer (tf=logarithmic idf=logarithmic)
Applying UnitLengthNormalizer
>> Clustering
Constructing linear kernel matrix (CosineSimilarity)
Selected clustering algorithm: KSSC (k=5)
Selected label generator: IGAIN (top=9)
Applying clustering algorithm...
Generated 5 clusters in 1.8 seconds
>> Post-processing
Validation: NMI=0.87
Applying label generator...
Generated 5x9 cluster labels
rugby    : rugby,wales,ireland,nation,scotland,robinson,england,france,six
tennis   : seed,open,6-3,6-4,australian,7-6,tennis,7-5,roddick
athletics: olympics,athlete,indoor,race,athens,drug,european,gold,iaaf
football : chelsea,club,arsenal,league,united,football,boss,manchester,manager
cricket  : cricket,one-day,pakistan,wicket,series,india,test,bowl,bowler
```

Figure 4.2: Output for the `cluster-label` tool when applied to the *bbcsport* dataset.

4.3.3 Sample Usage

The components provided in the core and document clustering libraries of TCT are designed to be used in a modular fashion, allowing developers to chain together multiple analysis procedures to produce custom applications. To demonstrate how this works in practice, we consider a sample configuration that combines several procedures as defined by the workflow shown in Figure 4.3. Specifically, given a previously parsed document collection, we wish to chain five individual toolkit components pertaining to different phases of the cluster analysis process: preprocessing (TF-IDF normalisation, k-way spectral decomposition), clustering (ensemble clustering based on resampling and hierarchical co-association integration) and validation (the external NMI measure).

Figure 4.4 provides a listing for the Java source code for an application that implements the configuration given in Figure 4.3. By utilising the TCT libraries, this relatively complex configuration can be rendered using few lines of code. It is clear from these figures that there is a direct mapping between the conceptual cluster analysis workflow and the corresponding application implementation. It is also apparent from this example that the toolkit allows us to experiment with novel algorithm combinations.

4.4 Summary

In this chapter we provided an overview of the design and implementation of the *Text Clustering Toolkit* (TCT), a new framework for the development of unsupervised data analysis applications. While this toolkit is specifically aimed at researchers working with document collections, its modular, extensible nature makes it suitable for use in the context of a variety of alternative machine learning problems. This chapter also introduced specific functions and example usage. Further details regarding technical aspects of the implementation, including developer documentation and user tool manual pages, are available at the toolkit homepage[4].

[4]TCT homepage: `http://mlg.ucd.ie/tct`

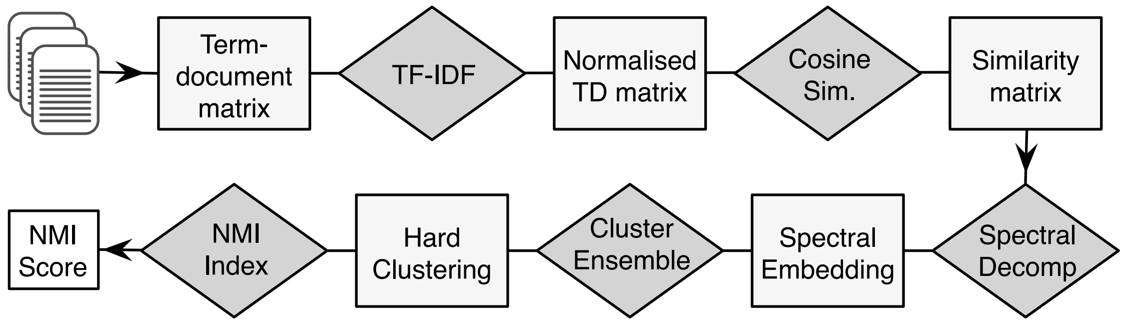


Figure 4.3: Sample cluster analysis workflow that chains multiple toolkit components.

```
public class TestApplication extends TCTApplication
{
  public double runTest( String[] args ) throws Exception
  {
    // Initialise toolkit
    init(args);

    // Load dataset from the repository
    Corpus corpus = repository.getCorpus( "bbcsport" );
    int k = corpus.getNaturalClasses().size();
    // Apply TF-IDF pre-processing
    new TfIdfNormalizer().apply(corpus);

    // Build a cosine similarity matrix
    SymMatrix S = new CosineSimilarity().buildSimilarityMatrix( corpus );
    // Apply spectral decomposition to the matrix
    SpectralDecomposition decomp = new NCutAffinityDecomposition( S, k );
    Embedding spectralEmbedding = decomp.apply(corpus);

    // Apply ensemble clustering to the embedding
    Generator generator = new SamplingGenerator( new KMeans( k ) );
    Integrator integrator = new HierCoAssociationIntegrator( k );
    EnsembleClusterer ensemble = new EnsembleClusterer( generator, integrator );
    Clustering clustering = ensemble.findClusters( spectralEmbedding );

    // Validate the clustering using NMI external validation
    return new NMIMeasure().validate( clustering );
  }
}
```

Figure 4.4: Source code for an application implementing the cluster analysis workflow defined in Figure 4.3, which involves chaining multiple toolkit components.

Chapter 5

Baseline Analysis

5.1 Introduction

When exploring an unstructured document collection in a real-life learning scenario, a user is faced with the dilemma of choosing from among the many existing approaches that have been proposed for document clustering, such as those reviewed in Chapter 2. A similar problem arises when selecting a suitable validation method from those surveyed in Chapter 3. Although several authors have performed comparative evaluations on text data involving a limited number of algorithms (*e.g.* Steinbach *et al.*, 2000), we use TCT to provide a comprehensive evaluation of classical cluster analysis methods, allowing us to examine their respective advantages and limitations. This evaluation also serves as a baseline for the assessment of the novel approaches described in chapters 6 and 7.

5.2 Experimental Setup

Before detailing our experimental results, we begin by providing a full description of the datasets and experimental methodologies which are common to all experiments discussed in the remainder of this thesis.

5.2.1 Real-World Datasets

When assembling a set of real-world document collections for experimental evaluation, we aimed to select corpora that differ significantly in their theme, dimensions, complexity and underlying structure. Of those chosen, the smallest contains 505 documents, while the largest contains 8580 documents. Some of these corpora have been widely used as

Dataset	Description	n	m	$\hat{k}$	$\bar{n}_c$	Balance
bbc	News articles from BBC	2225	9635	5	445	0.755
bbcsport	Sports news articles from BBC	737	4613	5	147	0.377
classic	CISI/CRAN/MED sets	7097	8276	4	1774	0.322
classic3	CACM/CISI/CRAN/MED sets	3893	6733	3	1297	0.708
cstr	Computer science technical abstracts	505	2117	4	126	0.398
ng17-19	Overlapping newsgroups	2625	12020	3	875	0.824
ng3	Well-separated newsgroups	2928	12357	3	976	0.943
reuters5	Top 5 categories from *Reuters-21578*	2317	4627	5	463	0.136
reviews	Entertainment articles from TREC	4069	18152	5	813	0.099
sports	Sports news articles from TREC	8580	14615	7	1225	0.036

Table 5.1: Details of the real-world text datasets used in experiments.

benchmarks for document clustering, while others represent novel collections that have proved useful during the course of our research. In all cases, external knowledge in the form of a manual classification of documents was available for comparison purposes. A full summary of these datasets is provided in Table 5.1, where n denotes the number of documents, m denotes the number of terms, $\hat{k}$ indicates the number of natural classes in the data, and $\bar{n}_c$ denotes the mean number of documents in each class. In addition, the "balance" of each dataset refers to the ratio of the size of the smallest natural class in the data relative to the size of the largest.

News Articles

Articles from news sources provide a rich source of topics for document clustering due to their high quality and tendency to cover a wide range of interesting themes. For the purposes of our evaluation, we constructed two new text corpora[1] from articles published by the BBC news service[2]. Specifically, the *bbc* corpus consists of 2225 complete news articles published during 2004-2005, which cover five topical areas: 'business', 'entertainment', 'politics', 'sport', and 'technology'. The *bbcsport* corpus consists of a smaller set of 737 sports news articles from the same source and time period, also containing five categories: 'athletics', 'cricket', 'football', 'rugby', and 'tennis'. An advantage of producing corpora from these sources is that the data and any resulting clusterings may be interpreted by researchers without the need for any specialist knowledge regarding the subject matter.

Reuters-21578 is a collection of documents published by Reuters newswire in 1987,

[1]Available from `http://mlg.ucd.ie/datasets/`

[2]See `http://news.bbc.co.uk`

which has become the most widely used benchmark for text classification. The entire collection consists of 21578 documents relating to 135 topics. In our evaluations we consider a subset of documents from this collection, referred to as *reuters5*, which is constructed as described by Lafferty & Lebanon (2004). Specifically, all documents not assigned to a single category are removed, and documents assigned to the five largest categories are then selected: 'earn', 'acq', 'crude', 'money-fx' and 'grain'.

The *reviews* and *sports* datasets[3] have been used by a number of authors in the evaluation of document clustering algorithms. They consist of subsets of news articles that were selected from the TREC collection based on their theme. The *reviews* dataset contains articles, originally published by the San Jose Mercury newspaper, which pertain to food, movies, music, radio and restaurants. The *sports* dataset contains sports news articles from the same source covering American football, baseball, basketball, boxing, cycling, golf and ice hockey. Both of these datasets contain largely overlapping natural classes that differ significantly in size, which is reflected by their low balance scores in Table 5.1.

Technical Abstracts

The *classic3* and *classic* datasets are formed from sets of technical abstracts contained in Cornell's SMART repository[4], which have been widely used to evaluate supervised text mining procedures. The former is constructed from sets of abstracts from information retrieval papers (CISI), medical journals (*Medline*) and aeronautical systems papers (*Cranfield*). The *classic* dataset also includes a fourth set, containing abstracts from computer systems research papers (CACM).

The *cstr* dataset[5] represents a smaller collection of abstracts that have also frequently been used for benchmark evaluation. The abstracts are taken from technical reports published by members of the Department of Computer Science at the University of Rochester, and relate to four fields of research: AI, robotics/vision, systems, and theory.

Newsgroup Messages

The *20 Newsgroups* (20NG) collection[6] contains 20,000 Usenet postings, consisting of 1000 documents from each of 20 different newsgroups covering a wide range of topics. The

[3]Available from `http://www.cs.umn.edu/~karypis/cluto`
[4]Available from `ftp://ftp.cs.cornell.edu/pub/smart`
[5]Original abstracts available from `http://www.cs.rochester.edu/trs`
[6]Available from `http://people.csail.mit.edu/jrennie/20Newsgroups/`

categorisation scheme for the collection is derived from these newsgroups. Various subsets of this collection have recently been used to evaluate clustering algorithms. Specifically, we consider the *ng3* subset, which is composed of three relatively well-separated groups pertaining to religion, politics and cryptography. We also make use of the *ng17-19* subset, which contains messages from three newsgroups that exhibit considerable overlap.

5.2.2 Artificial Datasets

While many authors evaluating clustering techniques have made use of synthetic datasets, the generation of data that realistically models the distribution of term frequency values in natural language text is difficult. As an alternative, we used the 20NG collection as a source for artificially composed datasets because it contains a range of topics that overlap to varying degrees. These datasets are specifically designed to evaluate the ability of cluster analysis methods to work on data containing a variety of different cluster structures, such as unbalanced cluster sizes or overlapping clusters. From the full collection we derived a large number of smaller datasets for which the correct value of $\hat{k}$ is known. Specifically, we constructed 84 sets in total[7], 12 for each value of $\hat{k} \in [2, 8]$. Half of these datasets consist of newsgroups that are reasonably compact and well-separated (*e.g.* 'graphics', 'hockey', 'mideast'). The remaining datasets are formed from newsgroups that overlap considerably (*e.g.* 'mac', 'windows'). These two halves are further divided into subsets of datasets containing clusters of different proportions, in a manner similar to that suggested by Giurcaneanu & Tabus (2004) for producing artificial data: balanced clusters containing 500 documents each, unbalanced clusters where one cluster contains 10% of the documents in the dataset, and unbalanced clusters where one cluster contains 60% of the documents. In all cases the documents were randomly drawn from each newsgroup. The resulting datasets range in size from 1000 to 4000 documents, with the corresponding dimensionality ranging from 4674 to 16282 unique terms.

5.2.3 Data Preprocessing

The collections of raw documents were parsed and processed according to the standard text mining practises described in Section 2.2.1. For all datasets, we employed a stop-list containing 300 entries[8] to remove common functional words. We then applied the

[7]Available from `http://mlg.ucd.ie/datasets/`

[8]Available from `http://mlg.ucd.ie/files/tct/stopwords.txt`

standard Porter suffix stripping algorithm (Porter, 1980) to stem words to their roots. We subsequently excluded terms occurring in less than three documents, which is widely performed in text mining tasks to reduce dimensionality (Drucker *et al.*, 1999). To weight term frequency values, we use a variant of log-based *tf-idf*, where the inverse document frequency component is defined as $idf(i,j) = \log \frac{m}{df_i}$. Unless otherwise specified, all experiments in this thesis involve clustering using the cosine measure (2.4) to assess the similarity between document vectors.

5.2.4 Evaluation Criteria

Accuracy

In Section 3.3 we described a variety of existing approaches that may be used to measure the agreement between two partitions of the same dataset, which is frequently done when comparing a newly generated clustering solution to a set of manually labelled natural classes. However, while a number of these indices have been specifically used in conjunction with text data, many of them exhibit biases with respect to the number of clusters in a partition or their structure.

To demonstrate this, we consider the mean scores computed on a large number of pairs of clusterings for four popular indices: NMI, purity, normalised entropy (here we consider the inverse $1 - NE(\mathcal{C})$) and the F-measure. In Figure 5.1(a), where the clusterings have

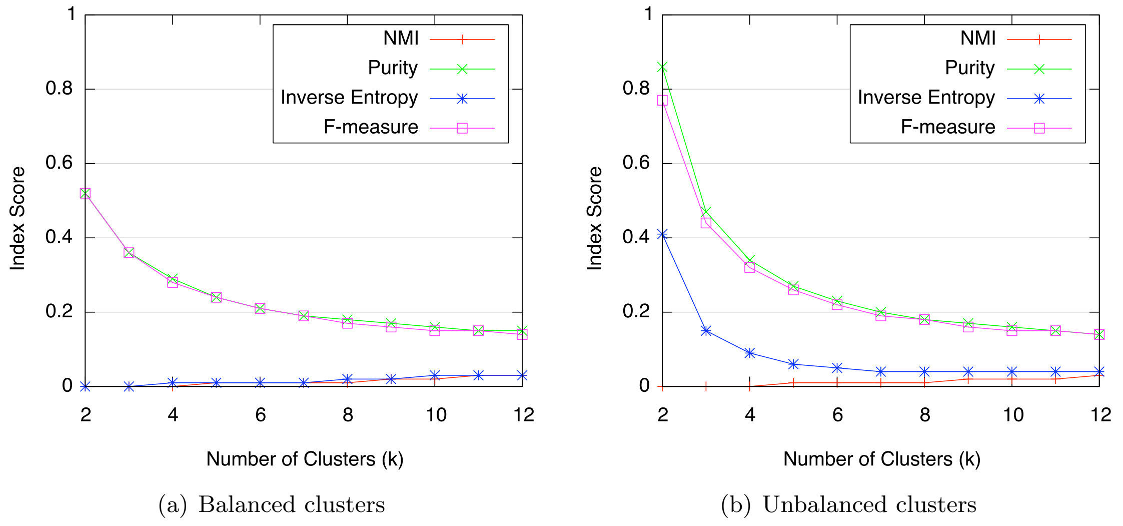


Figure 5.1: Mean external validation index scores based on the pairwise agreement between random partitions of 1000 data objects, illustrating the biases of popular external indices with respect to the number of clusters k and cluster balance.

been generated by random assignment, we see that the two measures based on set matching award relatively high scores to trivial solutions, particularly for smaller values of k. In the experiments summarised by Figure 5.1(b), the solutions were also randomly generated, but in each case one cluster was constrained to be approximately 25% the size of the others, leading to unbalanced clusterings. As a consequence of this, the purity, entropy and F-measure techniques all exhibit a significant bias in relation to the relative proportions of the clusters. In contrast, the NMI measure exhibits little bias toward the number of clusters, particularly for smaller values of k, and the quality of its evaluations are not influenced by cluster balance. We have observed similar behaviour when examining other configurations involving skewed cluster sizes.

Our empirical observations corroborate the theoretical work of Strehl & Ghosh (2002a), who proposed NMI as a means of providing a robust assessment of external accuracy. For the remainder of this thesis, we employ NMI as calculated using Eqn. 3.33 to compare the relative accuracy of document clustering algorithms. It is worth noting that, in recent literature, NMI has largely become the standard technique for performing this task.

Stability

When considering stochastic clustering algorithms, not only is it important to consider the ability of an algorithm to produce accurate clusters, but also the frequency with which such solutions are produced. While it is possible to consider the variance of the NMI scores resulting from several runs of the algorithm, this can sometimes be deceptive as it is possible for an algorithm to produce solutions of approximately equal accuracy but that differ significantly in terms of cluster content. Therefore, another aspect for evaluation is related to the notion of stability introduced in Section 3.4. In cases where we wish to assess the stability of stochastic techniques, we employ an extension of the *average normalised mutual information* (ANMI) measure (Strehl & Ghosh, 2002a). Formally, given a collection of r hard clusterings $\mathbb{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_r\}$ generated on the same dataset, we measure the average NMI score between each unique pair of solutions:

$$ANMI(\mathbb{C}) = \frac{1}{r(r-1)} \sum_{\mathcal{C}_i, \mathcal{C}_j \in \mathbb{C}} NMI(\mathcal{C}_i, \mathcal{C}_j) \tag{5.1}$$

A larger value is indicative of a higher level of agreement between the clusterings in $\mathbb{C}$, suggesting that the algorithm producing the solutions affords a greater degree of stability.

5.3 Comparison of Benchmark Clustering Algorithms

To provide a baseline for comparing clustering algorithms in the remainder of this thesis, we now present an evaluation of the standard partitional and hierarchical clustering algorithms described in sections 2.3 and 2.4 respectively.

5.3.1 Partitional Algorithms

We performed a comparison of three classical partitional clustering methods that have been widely employed to many types of data including document corpora: standard k-means with cosine similarity (KM), spherical k-means (SKM), and EM clustering. Each experiment was run for 200 executions on the real datasets listed in Table 5.1. In all cases, the parameter k was set to correspond to the number of natural classes $\hat{k}$, and the algorithms were initialised with random cluster assignments.

Table 5.2 presents the mean and standard deviation of the NMI scores for the three partitional algorithms. We observe that k-means can potentially produce reasonably accurate clusterings of text data. However, its performance is highly dependent on the choice of initial clusters. The other algorithms demonstrated similar sensitivity, resulting in a high level of variance in terms of accuracy on all datasets. The SKM algorithm is approximately equivalent to k-means, with the sole difference that document and centroid vectors are normalised to unit length. Therefore, it is unsurprising that the performance of the two algorithms is often highly similar. Of the three approaches, the EM algorithm consistently resulted in the lowest clustering accuracy.

Dataset	KM	SKM	EM
bbc	0.81 ± 0.08	**0.82 ± 0.08**	0.81 ± 0.08
bbcsport	0.73 ± 0.10	**0.77 ± 0.10**	0.73 ± 0.09
classic	0.70 ± 0.04	**0.71 ± 0.03**	0.68 ± 0.04
classic3	**0.93 ± 0.08**	0.90 ± 0.12	0.86 ± 0.11
cstr	**0.69 ± 0.05**	0.65 ± 0.03	0.66 ± 0.05
ng17-19	**0.41 ± 0.12**	0.39 ± 0.12	0.35 ± 0.10
ng3	**0.83 ± 0.10**	**0.83 ± 0.10**	0.81 ± 0.12
reuters5	**0.55 ± 0.07**	**0.55 ± 0.05**	0.50 ± 0.07
reviews	**0.56 ± 0.08**	**0.56 ± 0.08**	0.52 ± 0.07
sports	**0.62 ± 0.05**	0.60 ± 0.06	0.53 ± 0.05

Table 5.2: Summary of NMI accuracy results (mean and standard deviation) for partitional clustering methods, when applied to real-world text datasets.

5.3.2 Hierarchical Algorithms

Agglomerative Clustering

We now turn our attention to evaluating methods based on hierarchical agglomeration, which has previously been regarded as the prototypical strategy for clustering document collections. As discussed in Section 2.4, a number of strategies exist for determining which pair of clusters should be merged at each stage of the agglomeration process. We evaluate four linkage strategies based on a precomputed cosine similarity matrix: average linkage (AHC-AL), single linkage (AHC-SL), complete linkage (AHC-CL), and clustering based on the min-max graph partitioning criterion (AHC-MM). In all experiments, the merging process is terminated when $\hat{k}$ leaf clusters remain. As these techniques are deterministic, instability is not an issue and a single execution on each dataset is sufficient to produce a definitive solution.

The NMI scores listed in Table 5.3 for the traditional linkage strategies (AL/SL/CL) clearly illustrate the significant drawbacks of agglomerative clustering techniques when working with noisy, real-world data. For the AHC-AL and AHC-CL methods, the presence of outlying documents leads to poor merging decisions being made in the early stages of the clustering process, which cannot be subsequently rectified. In the case of AHC-SL, the tendency of the single linkage strategy to suffer from the effect of "chaining" is highly evident. For all three strategies, these problems are often reflected in the generation of a single, dominant cluster and one or more singleton clusters. As a result, the final clustering is often little better than random, as documents are coerced into being co-

Dataset	AHC-AL	AHC-SL	AHC-CL	AHC-MM
bbc	0.03	0.02	0.17	**0.74**
bbcsport	0.48	0.03	0.46	**0.90**
classic	0.01	0.01	0.22	**0.65**
classic3	**0.01**	**0.01**	**0.01**	**0.01**
cstr	0.51	0.04	0.40	**0.70**
ng17-19	0.04	0.01	0.13	**0.39**
ng3	0.01	0.01	0.20	**0.82**
reuters5	**0.61**	0.02	0.28	0.50
reviews	0.06	0.01	0.14	**0.57**
sports	0.20	0.01	0.11	**0.60**

Table 5.3: Summary of NMI accuracy results for agglomerative hierarchical clustering methods, when applied to real-world text datasets.

assigned even though they do not share a common topic or set of terms. It is interesting to note that, when evaluating hierarchical methods, some authors tend to make use of external validation measures such as purity (3.19) or entropy (3.29), which can produce reasonably high scores for such unbalanced clusterings, obscuring the true degree of error in the co-assignment of documents.

In contrast to the other linkage strategies, the min-max cut technique (AHC-MM) proposed by Ding & He (2002) lead to significantly better clusterings. Since values for this criterion are scaled by self-similarity (see Eqn. 2.20), it is possible that small clusters can result in large linkage values, allowing them to be merged with larger clusters. This leads to an inherent bias toward balanced clusters, which minimises the influence of outliers and reduces the tendency to produce singletons. However, as evidenced by the poor solution produced for the *classic3* dataset, inappropriate merging decisions can still be made when using this criterion. This is surprising given the well-separated nature of the underlying classes in this dataset, which are identified with relative ease by the partitional clustering algorithms discussed previously.

Refined Agglomerative Clustering

Several authors have considered the application of iterative relocation to improve the output of agglomerative clustering. Ding & He (2002) proposed a greedy process where documents are moved between clusters based on a given objective function, in a manner similar to that used by the "first variation" algorithm described by Dhillon *et al.* (2002a). However, the computational cost of this process can vary significantly, depending on the size of the dataset and the number of refinement iterations. As a simpler, less time consuming alternative, we suggest the application of the standard k-means algorithm with cosine similarity to the output of an agglomerative algorithm, so that erroneously assigned documents can be moved to more appropriate clusters. The refinement process may also be viewed as using hierarchical clustering to provide k-means with a deterministic and hopefully accurate set of initial clusters. This idea was originally proposed by Jain & Dubes (1988) and developed by Cutting *et al.* (1992) in the 'buckshot' algorithm, although it has not been commonly considered in recent clustering literature. It should be noted that the application of all such refinement techniques, which effectively involve moving documents between leaf nodes of the hierarchy, has the effect that the clustering can no longer be interpreted using a dendrogram, which may affect the interpretability of the

Dataset	AHC-ALR	AHC-SLR	AHC-CLR	AHC-MMR
bbc	0.38	0.02	0.65	**0.88**
bbcsport	0.48	0.45	0.78	**0.95**
classic	0.38	0.54	**0.67**	0.65
classic3	0.00	0.00	**0.95**	0.00
cstr	0.63	0.41	0.44	**0.75**
ng17-19	0.05	0.01	0.52	**0.56**
ng3	0.62	0.22	0.51	**0.90**
reuters5	**0.68**	0.66	0.59	0.58
reviews	0.44	0.01	0.55	**0.67**
sports	**0.70**	0.56	0.58	0.62

Table 5.4: Summary of NMI accuracy results for agglomerative hierarchical clustering methods with refinement, when applied to real-world text datasets.

solution. However, the production of a more accurate, flat partition of the data may be more useful than a trivial, inaccurate tree of clusters.

Table 5.4 provides a comparison of the NMI scores achieved by the agglomerative methods with k-means refinement. We observe that the application of refinement can often lead to significant increases in accuracy when compared to the results listed in Table 5.3. In particular, the application of refinement subsequent to min-max linkage clustering (AHC-MMR) produced highly accurate clusterings on several datasets, such as the *bbcsport* and *reviews* corpora. However, the quality of the output of the hierarchical algorithm still plays a significant role. For instance, AHC-MMR shows little improvement on the *classic3* dataset, where the singleton leaf clusters in the hierarchy lead k-means to quickly converge to an almost identical final solution.

Divisive Clustering

We now investigate the performance of divisive clustering algorithms, which have frequently been applied for document clustering. Specifically, we considered the Principal Direction Divisive Partitioning (PDDP) and bisecting k-means algorithms. In the latter case, we examined cluster splitting based on both cluster size (BKM-S) and minimum average intra-cluster similarity (2.13) (BKM-A). For each split, we produced 20 randomly-initialised bisections, from which the best was selected as determined by the mean document-centroid similarity criterion (2.12). Due to the stochastic element in the bisecting k-means algorithm, we ran the entire process 100 times. In all experiments, we

Dataset	PDDP	BKM-S	BKM-A
bbc	0.77	0.82 $\pm$ 0.04	**0.83 $\pm$ 0.03**
bbcsport	0.60	**0.74 $\pm$ 0.03**	**0.74 $\pm$ 0.03**
classic	0.65	0.71 $\pm$ 0.01	**0.80 $\pm$ 0.09**
classic3	0.85	**0.94 $\pm$ 0.00**	**0.94 $\pm$ 0.00**
cstr	0.64	**0.71 $\pm$ 0.02**	**0.71 $\pm$ 0.02**
ng17	0.29	0.46 $\pm$ 0.08	**0.47 $\pm$ 0.07**
ng3	**0.81**	0.71 $\pm$ 0.12	0.73 $\pm$ 0.12
reuters5	0.50	0.60 $\pm$ 0.02	**0.62 $\pm$ 0.04**
reviews	0.50	0.52 $\pm$ 0.04	**0.53 $\pm$ 0.04**
sports	0.67	0.61 $\pm$ 0.04	**0.70 $\pm$ 0.05**

Table 5.5: Summary of NMI accuracy results for divisive hierarchical clustering methods, when applied to real-world text datasets.

stop the divisive process when the number of leaf clusters reaches $\hat{k}$.

From the results listed in Table 5.5, we firstly observe that divisive techniques perform better on average than their agglomerative counterparts. This corresponds to the experimental findings of Steinbach *et al.* (2000), who suggested that the improvement in accuracy is due to the availability of a global view of the entire dataset at the beginning of the clustering process. While bisecting k-means does exhibit some variance over multiple runs, its output tends to be far more consistent than the randomly-initialised k-means algorithm. The choice of cluster splitting criterion for bisecting k-means did not appear to influence performance as greatly as one might expect, which may be due to the relatively balanced nature of the group structures in many of the datasets under consideration. The BKM-A splitting strategy lead to slightly higher mean NMI scores on several datasets, with a particularly noticeable improvement on the *sports* corpus, which contains highly unbalanced cluster sizes. This confirms our belief that the size-based BKM-S strategy is generally unsuitable unless it is known in advance that a corpus will contain clusters of approximately equal size.

While the PDDP algorithm has been widely used in the document clustering literature, our results indicate that it often performs worse than other hierarchical or partitional methods when applied to data with significantly overlapping structures. Notably, it frequently produced less accurate clusterings than those generated by bisecting k-means. On the other hand, a significant benefit of PDDP is the deterministic nature of the algorithm so that, like the agglomerative algorithms evaluated previously, a single execution is sufficient to produce a definitive result. From a computational perspective, PDDP also performs favourably when compared to the other hierarchical algorithms considered here.

Refined Divisive Clustering

We also considered the possibility of applying k-means to refine the output of hierarchical divisive clustering algorithms. The PDDP-R approach is similar to that employed by Kruengkrai *et al.* (2004), who used Gaussian EM clustering to improve the output of the PDDP algorithm. However, the results given in the evaluation of the partitional algorithms in Section 5.3.1 suggest that k-means with cosine similarity is better suited for this task when working with text data. The combination of refinement with the size-based bisecting k-means variant, denoted BKM-SR, is equivalent to the strategy proposed by Steinbach *et al.* (2000). BKM-AR represents an analogous approach which makes splitting decisions based on average intra-cluster similarity.

The application of refinement in conjunction with bisecting k-means lead to increases in accuracy on certain datasets, although in the case of the *classic* and *reuters5* corpora we actually observed a decrease in the resulting NMI scores. The marginally better initial partitions available to the BKM-AR method lead refined solutions that were better than those produced by BKM-SR or PDDP-R. The gains in accuracy resulting from iteratively relocating documents in clusterings produced by PDDP were quite significant. However, these techniques do not frequently achieve scores comparable to those afforded by the best agglomerative method (AHC-MMR). Once again, the application of refinement transforms a nested hierarchy of clusters into a flat partition, which may impact upon interpretability.

Dataset	PDDP-R	BKM-SR	BKM-AR
bbc	**0.87**	**0.87 ± 0.03**	**0.87 ± 0.03**
bbcsport	0.69	**0.77 ± 0.04**	0.76 ± 0.03
classic	0.70	0.69 ± 0.01	**0.71 ± 0.03**
classic3	**0.95**	**0.95 ± 0.00**	**0.95 ± 0.00**
cstr	0.70	**0.72 ± 0.02**	**0.72 ± 0.03**
ng17	**0.56**	0.47 ± 0.07	0.47 ± 0.08
ng3	**0.88**	0.73 ± 0.13	0.73 ± 0.12
reuters5	0.59	0.59 ± 0.02	**0.60 ± 0.04**
reviews	0.52	**0.55 ± 0.05**	0.54 ± 0.05
sports	0.65	0.61 ± 0.04	**0.66 ± 0.05**

Table 5.6: Summary of NMI accuracy results for divisive hierarchical clustering methods with refinement, when applied to real-world text datasets.

5.3.3 Summary

While partitional clustering methods such as k-means are commonly employed in document clustering, we observe that the success of these algorithms is largely dependent on the choice of initial clusters. When using standard stochastic initialisation strategies, the accuracy of solutions generated on the same data across multiple trials can vary significantly. To address this problem, in Section 7.3 we propose an ensemble clustering approach that combines multiple clusterings generated by k-means to produce a more robust solution.

From our comparison of hierarchical clustering algorithms, it is apparent that both agglomerative and divisive techniques have their respective merits and drawbacks. Neither approach produced highly accurate clusterings on all the datasets considered in our experiments. The agglomerative techniques, which have been widely used for document clustering, can often produce meaningless clusterings in the presence of outliers. The use of the min-max cut criterion (AHC-MM) does reduce this problem somewhat, although it may still produce poor results due to erroneous merging decisions.

Of the divisive hierarchical approaches we considered, the bisecting k-means algorithm appears to be the most promising. However, the sensitivity of the k-means algorithm to the choice of initial clusters remains a factor in the production of stable, accurate bisections. The execution of multiple iterations per bisection alleviates this somewhat. Unfortunately, this largely negates the efficiency benefits of the algorithms, and the choice of a suitable candidate bisection essentially introduces a cluster validation problem. Experimental results show that our suggestion of applying k-means with cosine similarity to refine the leaf nodes of a hierarchy of clusters can frequently lead to more accurate solutions, although a meaningful tree structure will no longer be available for user inspection.

5.4 Comparison of Benchmark Validation Methods

We now turn to the problem of cluster validation in document clustering applications. In this evaluation, we examine the performance of the internal validation techniques described in Section 3.2. Note that we only consider those that are practical for use on text data, which are denoted as follows:

CH:	Calinski-Harabasz index (3.1)
DI1:	Original formulation of Dunn's index (3.3)
DI2:	Centroid-based formulation of Dunn's index (3.4)
DI3:	Formulation of Dunn's index recommended by Bezdek & Pal (1995) (3.5)
DB:	Centroid-based formulation of Davies-Bouldin index (3.6)
CI:	C-index (3.7)
SIL:	Cosine-based silhouette index (3.11)
BIC:	Bayesian Information Criterion (3.15)

Internal validation indices are generally based on the assumption that the metrics used during validation will correspond to those that were originally employed in the generation of the clustering under consideration. Therefore, when employing these indices, we use cosine distance (2.5) as a measure of dissimilarity in place of Euclidean distance. While we did examine the possibility of using the latter metric in this context, our results reflected the findings of Strehl (2002), indicating that its emphasis on absent values makes Euclidean-based indices unsuitable in this context. However, for the BIC technique, which is based on the assumption that the clusters have Gaussian distributions where variance is modelled in terms of object-centroid Euclidean distances, the use of an alternative dissimilarity metric is not appropriate. We therefore use the original formulation of the criterion as described in Section 3.2.2.

5.4.1 Cluster Evaluation

Internal indices are commonly applied in real-world unsupervised learning tasks, where they provide a means of comparing multiple clusterings of a given dataset. As mentioned previously, we regard the external NMI measure as providing a robust measure of clustering accuracy in cases where class information is available. This external knowledge is assumed to provide a "gold standard" on which cluster analysis techniques may be evaluated. Therefore, we now examine the degree of correlation between NMI and the classical internal indices described in Section 3.2.1. Ideally, a high correlation should exist between the evaluations produced by NMI and those produced with a chosen internal index, indicating that the latter is successful at identifying clusterings that reflect the underlying classes in the data. Note that BIC is not included here as it is generally used for the task of model selection, rather than for cluster evaluation.

To evaluate the correlations, we required a large number of datasets to emphasise the differences between the various indices. Therefore, we made use of the artificial datasets derived from the 20NG collection as described in Section 5.2.2. We generated 100 clusterings on each of the 84 datasets, applying the NMI and internal validation indices to the resulting clusterings. For each dataset, we then computed the correlation between external and internal accuracy scores as given by the Spearman rank coefficient.

A summary of the absolute values of the mean correlations for different categories of artificial dataset is given in Figure 5.2. The standard formulation of the Dunn's index (DI1) and that recommended by Bezdek & Pal (1995) (DI3) frequently performed poorly, particularly in the presence of overlapping clusters. The technique involving centroid-based metrics (DI2) fared better, although the Dunn's index generally appears to represent a poor choice of validation of text data. The DB-index and C-index techniques achieved approximately equivalent validation performance, with the former performing better on balanced clusters, while the latter proving more successful on unbalanced clusters. The CH-index was noticeably better than either, while the cosine-based silhouette measure (SIL) proposed in Section 3.2.1 consistently provided the highest level of correlation with the external NMI index. This suggests that, from the indices considered here, SIL provides the most useful measure of clustering quality when class information is unavailable. Furthermore, the consistency of its performance across the various categories suggests that

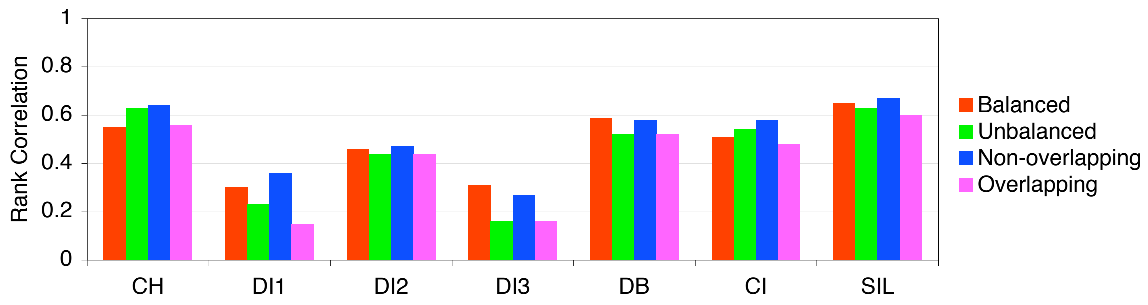


Figure 5.2: Summary of the absolute mean rank correlations between common internal validation indices and external NMI accuracy, measured across all artificial text datasets.

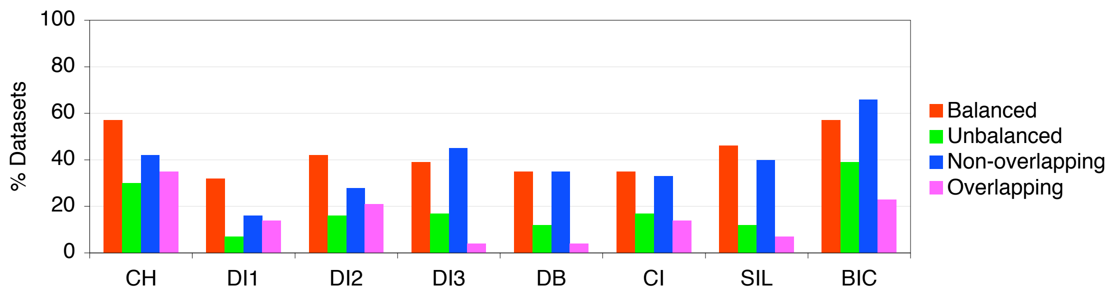
it does not suffer dramatically due to the presence of unbalanced or overlapping classes. However, it should be noted that the level of correlation for all the indices examined here is relatively low, indicating that internal indices can frequently fail to recognise clusterings which successfully uncover the underlying structures in text corpora.

5.4.2 Model Selection

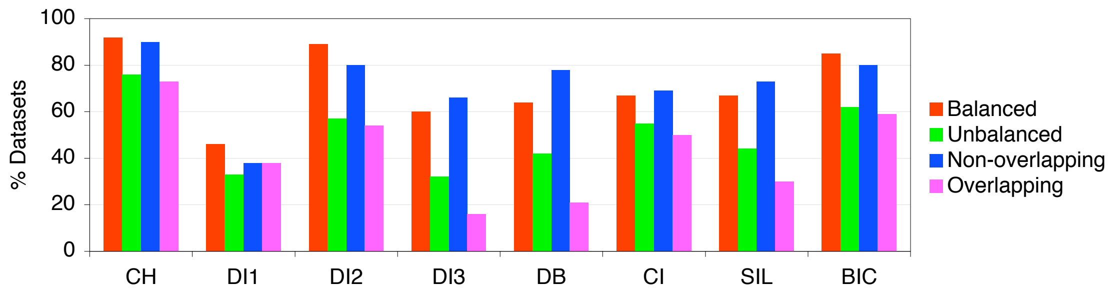
For many document clustering procedures, the most important consideration during parameter selection is the choice of the number of clusters k. Therefore, we now evaluate the suitability of popular internal validation techniques for the purpose of estimating the true number of clusters $\hat{k}$ in a document collection.

Evaluation on Artificial Data

For our initial set of experiments, we again used the set of artificial datasets constructed from the 20NG collection. We ran 100 executions of the k-means algorithm with cosine similarity for each value of k in the range $[2, 10]$, and applied eight internal indices to each clustering. We subsequently averaged the scores over all runs and compared the rankings



(a) Correct estimations



(b) Top-3 estimations

Figure 5.3: Summary of the overall percentage of correct and top-3 estimations for $\hat{k}$ as selected by common internal validation indices, measured across all artificial text datasets.

produced by the indices with the known number of clusters $\hat{k}$.

To summarise the results of our experiments, Figure 5.3(a) shows the percentage of datasets on which each validation index correctly identified $\hat{k}$, while Figure 5.3(b) shows the percentage of datasets for which $\hat{k}$ was selected among the top three choices. These results indicate that, although the internal indices under consideration have proved popular and effective for model selection in other domains, the majority of these techniques perform poorly on text data. Even with the use of an appropriate cosine-based dissimilarity metric in place of Euclidean distance, techniques such as variants of the classical Dunn and DB indices fail to correctly estimate $\hat{k}$ on more than 25% of the 84 datasets. There are two exceptions: the Calinski-Harabasz (CH) index and the Bayesian Information Criterion (BIC). Both afforded significantly better estimates for the number of clusters, particularly on datasets containing overlapping groups. Surprisingly, BIC performed well when using Euclidean distance, while we have generally observed that the other internal indices perform very poorly when using that metric. It is interesting to note that the cosine-based silhouette measure (SIL), which exhibited a high level of correlation with external accuracy in the previous set of experiments, performs poorly in the context of parameter selection. This suggests that, while certain indices can provide acceptable performance when choosing between partitions with the same number of clusters, this success may not translate to the problem of choosing a suitable clustering model. In the case of SIL, this is due to an inherent bias in the index toward larger values of k, which is also present in the original formulation proposed by Rousseeuw (1987).

Evaluation on Real Data

For our second evaluation, we compared the eight internal validation schemes on the real-world corpora described in Section 5.2.1. The experimental setup was identical to that used for the artificial data. Table 5.7 summarises the top three estimations produced by the indices on each corpus, where highlighted values indicate cases where $\hat{k}$ was correctly identified. Once again, we observe that the classical CH index generally proved most successful in correctly estimating the number of clusters. Notably, it was the only technique to provide useful recommendations for the *sports* dataset, which contains highly unbalanced clusters. In contrast to the results produced on artificial data, in this set of experiments the BIC technique generally lead to the poorest results of all those techniques considered. This was particularly evident on the larger datasets, suggesting that the performance of

Dataset	$\hat{k}$	CH	DI1	DI2	DI3	DB	CI	SIL	BIC
bbc	5	**5**,6,4	**5**,2,4	**5**,4,6	**5**,6,7	**5**,6,4	6,7,**5**	**5**,6,7	6,7,**5**
bbcsport	5	6,**5**,7	4,3,**5**	4,6,**5**	6,**5**,4	6,7,**5**	8,7,6	6,7,8	**5**,3,4
classic	4	3,**4**,5	3,2,**4**	2,3,**4**	3,**4**,2	3,2,**4**	2,3,**4**	2,3,**4**	10,9,8
classic3	3	**3**,4,2	2,**3**,4	**3**,2,4	**3**,2,9	**3**,2,4	**3**,2,4	**3**,2,4	8,9,7
cstr	4	3,**4**,5	3,2,**4**	2,3,**4**	10,9,8	3,2,10	3,**4**,2	3,2,**4**	3,2,**4**
ng17-19	3	5,6,4	2,**3**,4	4,**3**,5	9,5,7	10,9,8	2,**3**,4	9,8,10	7,6,5
ng3	3	**3**,4,5	**3**,2,4	**3**,2,4	**3**,4,5	**3**,4,2	**3**,2,4	**3**,4,2	6,5,4
reuters5	5	2,3,4	2,3,4	2,3,4	3,2,4	2,3,4	2,3,4	2,3,4	10,9,8
reviews	5	2,3,4	2,3,4	2,3,6	2,3,4	2,7,8	2,9,7	2,3,4	10,9,8
sports	7	6,**7**,5	2,3,4	2,3,6	2,3,4	10,8,9	2,3,6	2,3,6	10,9,8

Table 5.7: Summary of the top-3 estimations for $\hat{k}$ as selected by common internal validation indices when applied to clusterings of real-world text datasets.

the BIC formulation proposed by Pelleg & Moore (2000) is biased with respect to the number of documents n. For larger values of n, BIC frequently over-estimates the number of clusters, which corroborates the experimental observations made by Hamerly & Elkan (2004).

5.4.3 Summary

Our experiments show that the related tasks of evaluating clustering solutions and choosing suitable models for clustering can be difficult to perform successfully when working with text corpora due to the complex nature of the data. It is apparent that many internal validation technique, such as the well-known indices proposed by Dunn (1974b) and Davies & Bouldin (1979), are ill-suited to assessing the validity of clusterings of text data, particularly in cases where the cluster structures are non-spherical or overlapping. For the task of estimating the optimal number of clusters $\hat{k}$, the popular model-based BIC technique performed well on artificial data, although it proved less successful on larger real-world corpora. The index described by Calinski & Harabasz (1974), adapted to use squared cosine distance, was best suited for this task, when considering both artificial and real datasets. Therefore, we use this index as a baseline for validation performance in the remainder of this thesis.

The experimental results presented here indicate that there is significant scope for improving the techniques available for model selection in document clustering. It should also be noted that, while our validation experiments are based on averaging index scores across a large number of runs, it is common practice to estimate $\hat{k}$ based on a single clustering for each potential value of k (Bolshakova & Azuaje, 2002). This can lead to estimations that are far less robust and highly sensitive to any instabilities introduced by the underlying clustering algorithm. To address these shortcomings, in Section 7.2 we introduce an alternative approach for performing model selection, based on the aggregation of multiple clusterings, which is robust to the presence of outlying documents and overlapping clusters that differ in structure.

Chapter 6

Improving Accuracy and Interpretability

6.1 Introduction

Experimental evaluations have shown that the accuracy afforded by many popular document clustering algorithms can vary significantly, depending on the complexity of the cluster structures in the data to which they are applied. In Section 6.2, we introduce novel approaches designed to improve the accuracy of recently proposed clustering algorithms when working with text data, specifically those based on spectral analysis and non-negative matrix factorisation. Many existing document clustering techniques do not address the problem of presenting a newly generated clustering to a user in an interpretable manner. To facilitate the extraction of knowledge from clusterings of text corpora, strategies are proposed for generating readily interpretable summary information, in the form of *descriptive* and *discriminative* cluster labels.

In Section 6.3, we shift our focus to improving the performance of recently-proposed kernel learning methods. We examine the phenomenon of *diagonal dominance*, which can significantly impact upon the accuracy and stability of popular centroid-based algorithms. This phenomenon especially problematic when kernel functions are applied to sparse high-dimensional data, such as text corpora. We explore the implications of diagonal dominance for kernel document clustering tasks, and propose a selection of strategies for reducing these effects, thereby leading to increased clustering accuracy and stability.

6.2 Producing Accurate Interpretable Clusters

A fundamental goal of document clustering is the identification of a set of groups that accurately reflects the topics present in a corpus. A second objective that is often overlooked is the provision of information to facilitate the human interpretation of the clustering solution. The application of dimension reduction techniques in document clustering has largely focused on improving algorithm accuracy and scalability. However, from a user's perspective, the production of concise, unambiguous descriptions of cluster content is also highly important. A simple but effective means of achieving this goal is to generate weights signifying the relevance of the terms in the corpus vocabulary to each cluster, from which a set of cluster labels can subsequently be derived. The provision of document membership weights can also help a user to gain an insight into a given clustering solution. For instance, when a document is assigned to a cluster, it may be useful to quantify the confidence of this assignment. These weights also allow us to represent cases where a given document relates to more than one topic.

In this section, we introduce a family of co-clustering algorithms which are motivated by the spectral analysis techniques described in Section 2.6. These algorithms provide membership weights for both terms and documents in the form of a soft co-clustering of the data. Furthermore, by applying an iterative matrix factorisation scheme, we can produce a refined clustering that affords improved accuracy and interpretability. We also examine a kernel-based approach, which can produce co-clusterings based on the use of an arbitrary affinity metric. The proposed algorithms are compared to existing matrix decomposition methods on a range of real-world datasets. Strategies for generating useful cluster descriptions in conjunction with these algorithms are also presented.

6.2.1 Soft Spectral Clustering

Spectral clustering algorithms have focused on the production of hard clusterings, where it is assumed that the matrix representation of a dataset can be organised in a block-diagonal fashion, such that the blocks correspond to a set of well-defined disjoint classes. In contrast, for text corpora it is not unusual for a single document to relate to more than one topic, resulting in overlapping classes. Given a term-document matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, we now examine the problem of inducing membership weights from a hard clustering, and propose an intuitive method to produce soft clusters based on the spectral co-clustering model introduced by Dhillon (2001).

Related Techniques

A common approach for the task of generating feature weights from a hard clustering is to examine the cluster centroid vectors, which can be viewed as providing a summary of the content of each cluster (Karypis & Han, 2000). In the *spherical k-means* algorithm (Dhillon & Modha, 2001), term weights are extracted from the unit normalised centroid of each cluster. Unfortunately, an analogous technique for spectral clustering is not feasible due to the presence of negative values in centroid vectors, which is a consequence of using eigenvectors to form the dimensions of the spectral embedding. Another possible approach is to consider the membership weights of a given document as being a function of the similarity between the document and each cluster centroid (Zhao & Karypis, 2004). Documents that are highly similar to a particular cluster centroid will be assigned a high membership weight for that cluster, whereas documents that bear little similarity to the centroid will be assigned a low weight. In the case of a co-clustering, we can also derive term membership weights using an analogous approach.

The success of spectral clustering methods has been attributed to the construction of an embedding from a truncated set of eigenvectors. When applied to text data, this has the effect of amplifying the association between documents that are highly similar, while simultaneously attenuating the association between documents that are dissimilar (Brand & Huang, 2003). However, while this process has been shown to improve the ability of a post-processing algorithm to identify cohesive clusters, the truncation of the decomposition of $\mathbf{A}$ to $k \ll m$ singular vectors introduces a distortion that makes the extraction of natural membership weights problematic. As a consequence, we observe that directly employing embedded term-centroid similarity values as membership weights will not provide intuitive cluster labels.

As an alternative to inducing soft weights from a hard partition, one may advocate the application of existing fuzzy clustering techniques. An analogous approach to spectral k-means clustering would involve the application of the fuzzy c-means algorithm (Bezdek, 1981). However, its effectiveness as a post-processing method for spectral document clustering is limited due to its reliance on a squared-norm metric to measure similarity and its inability to deal with outliers. The membership values produced by a fuzzy co-clustering of a spectral embedding will also be subject to the effects of the distortion described above.

Motivated by these factors, we focus on the problem of deriving soft membership weights from a hard clustering.

Inducing Soft Clusters

As a starting point, we construct a spectral embedding using the bipartite co-clustering approach described in Section 2.6.4. However, we form an embedding from k leading singular vectors rather than $\log_2 k$, as recent work has shown that truncating the eigenbasis to a smaller number of dimensions may negatively affect clustering accuracy and stability (Ng *et al.*, 2001). Once we have constructed the embedding $\mathbf{Z} \in \mathbb{R}^{(m+n)\times k}$, we produce a disjoint k-way clustering of the row vectors by applying the k-means algorithm with cosine similarity. This clustering can be represented as the $(m+n) \times k$ partition matrix $\mathbf{P}$, where the i-th column is a binary membership indicator for the i-th cluster.

Since the spectral co-clustering strategy is based on the principle of the duality of clustering documents and terms (Dhillon, 2001), we argue that we can induce a soft clustering of terms from the partition of documents in $\mathbf{Z}$, while a soft clustering of documents from the partition of terms. Observe that the matrix $\mathbf{P}$ has the following structure:

$$\mathbf{P} = \begin{bmatrix} \mathbf{P_1} \\ \mathbf{P_2} \end{bmatrix}$$

The $m \times k$ sub-matrix $\mathbf{P_1}$ indicates the assignments of terms to clusters, while the $n \times k$ sub-matrix $\mathbf{P_2}$ indicates the assignment of documents. An intuitive approach to producing term weights is to apply the transformation $\mathbf{A}\hat{\mathbf{P}}_\mathbf{2}$, where $\hat{\mathbf{P}}_\mathbf{2}$ denotes the matrix $\mathbf{P}_2$ with columns normalised to unit length. This effectively projects the centroids of the partition of documents in $\mathbf{Z}$ to the original feature space. Similarly, we can compute document weights by applying the transformation $\mathbf{A}^\mathsf{T}\hat{\mathbf{P}}_\mathbf{1}$, thereby projecting the embedded term cluster centroids to the original data.

We observe that, due to the "winner takes all" nature of the k-means algorithm, membership weights derived using the above approach will not reflect the existence of boundary objects lying between clusters or outliers that may be equally distant from all centroids. To overcome this problem, we propose projecting the centroid-similarity values from the embedded clustering to the original data. Due to the presence of negative values in $\mathbf{Z}$, these values will lie in the range $[-1, 1]$. We rescale the values to the interval $[0, 1]$ and normalise the k columns to unit length, representing them by the matrix $\mathbf{S} \in \mathbb{R}^{(m+n)\times k}$

as defined by:

$$S_{ij} = \frac{1 + cos(z_i, c_j)}{2} \qquad S_{ij} \leftarrow \frac{S_{ij}}{\sum_l S_{lj}} \tag{6.1}$$

As with the partition matrix $\mathbf{P}$ of the embedded clustering, the centroid-similarity matrix $\mathbf{S}$ can be divided into two sub-matrices

$$\mathbf{S} = \begin{bmatrix} \mathbf{S_1} \\ \mathbf{S_2} \end{bmatrix}$$

where $\mathbf{S_1} \in \mathrm{I\!R}^{m \times k}$ corresponds to the term-centroid similarity matrix and $\mathbf{S_2} \in \mathrm{I\!R}^{n \times k}$ is the document-centroid similarity matrix. By applying the projections $\mathbf{AS_2}$ and $\mathbf{A}^{\mathsf{T}}\mathbf{S_1}$ respectively, we can generate membership weights that consider both the affinity between vectors in the embedded space and the term frequency values from the original term space.

Soft Spectral Co-clustering (SSC) Algorithm

Motivated by the duality of the co-clustering model, we now present a spectral clustering algorithm with soft assignment of terms and documents that employs a combination of the transformation methods described in the previous section. We formulate the output of the algorithm as a pair of matrices $(\mathbf{U}, \mathbf{V})$, where $\mathbf{U}$ represents the term-cluster membership function and $\mathbf{V}$ represents the document-cluster membership function.

As an appropriate document membership function, we select the projection $\mathbf{A}^{\mathsf{T}}\mathbf{S_1}$ on the basis that the use of similarity values extracts more information from the embedded clustering than purely considering the binary values in $\mathbf{P}$. We observe that this generally leads to a more accurate clustering, particularly on datasets consisting of overlapping classes.

The requirements for a term membership function differ considerably from those of a document membership function, where accuracy is the primary consideration. As the production of useful cluster descriptions is a central objective of our work, we seek to generate a set of weights that results in the assignment of high values to relevant features and low values to irrelevant features. Consequently, we select the projection $\mathbf{A\hat{P}_2}$ as previous work has shown that centroid vectors can provide a summarisation of the important concepts present in a cluster (Dhillon & Modha, 2001). Our choice is also motivated by the observation that the binary indicators in $\mathbf{\hat{P}_2}$ result in sparse discriminative weight vectors, whereas the projection based on $\mathbf{S_2}$ leads to term weights such that the highest ranking words tend to be highly similar across all clusters.

1. Construct the normalised term-document matrix:

$$\mathbf{A_n} = \mathbf{D_1}^{-\frac{1}{2}}\mathbf{A}\mathbf{D_2}^{-\frac{1}{2}}$$

2. Compute the k leading singular vectors of $\mathbf{A_n}$ to produce the truncated factors $\mathbf{U_k} = (u_1, \ldots, u_k)$ and $\mathbf{V_k} = (v_1, \ldots, v_k)$.

3. Construct the embedding $\mathbf{Z}$ by scaling and stacking $\mathbf{U_k}$ and $\mathbf{V_k}$:

$$\mathbf{Z} = \begin{bmatrix} \mathbf{D_1}^{-1/2}\mathbf{U_k} \\ \mathbf{D_2}^{-1/2}\mathbf{V_k} \end{bmatrix}$$

4. Apply k-means with scaled cosine similarity and orthogonal initialisation to the embedding $\mathbf{Z}$ to generate a disjoint co-clustering.

5. Construct matrices $\mathbf{S_1}$ and $\mathbf{\hat{P}_1}$ from the co-clustering.

6. Form soft clusters by applying the projections $\mathbf{U} = \mathbf{A}\mathbf{\hat{P}_2}$ and $\mathbf{V} = \mathbf{A}^{\mathsf{T}}\mathbf{S_1}$.

Figure 6.1: Soft Spectral Co-clustering (SSC) algorithm.

The traditional approach for initialising k-means in the post-processing phase of spectral clustering is to randomly divide the embedded data into k groups. While clustering based on an appropriate number of singular vectors should produce robust results, we have observed that, for larger datasets, stochastic initialisation can still lead to inconsistent solutions. To avoid this behaviour, we generate initial clusters in a manner modelled on the furthest-first scheme described in Section 2.3.3. Specifically, the first cluster centroid is nominated as the most centrally located point in the embedded space. Each successive centroid is chosen to be as close as possible to 90^o from those that have been previously selected. After k centroids have been selected, the remaining objects are assigned to the nearest centroid in the embedding. In this way a fully deterministic solution may be produced. The complete procedure, designated as the *Soft Spectral Co-clustering* (SSC) algorithm, is summarised in Figure 6.1.

6.2.2 Refined Soft Spectral Clustering

We now present an approach to document clustering that builds upon the co-clustering techniques described previously to produce a refined clustering that affords improved accuracy, while retaining the interpretability of the clusters. As discussed in Section 2.6, the dimensions of the embedded space produced by spectral decomposition are constrained to be orthogonal. However, as text corpora will typically contain documents that pertain to multiple topics, the underlying semantic variables in the data will rarely be orthogonal. The limitations of spectral techniques to effectively identify overlapping clusters has motivated the introduction of other dimension reduction techniques such as NMF, where each document may be represented as the additive combination of the topics. Unfortunately, the standard NMF approach, which involves initialising a pair of factors with random positive values, can lead to convergence to a range of solutions of varying quality.

We argue that initial factors, produced using the soft cluster induction techniques discussed previously, can provide a good set of well-separated "core clusters". By subsequently applying matrix factorisation with non-negativity constraints to the corresponding membership matrices, we can effectively uncover overlaps between clusters. The combination of the global information available to spectral techniques with the local nature of iterative matrix factorisation methods can yield accuracy superior to that achieved by either of the individual approaches. In addition, the relative sparsity of the factors produced by NMF improves our ability to identify outlying documents and eliminate irrelevant terms.

Refined Soft Spectral Co-clustering (RSSC) Algorithm

We now describe a procedure to refine the output of methods based on the soft spectral co-clustering model. In the SSC algorithm described previously, our choice of projection for the construction of the term membership matrix was motivated by the desire to produce weights for subsequent use in producing cluster labels. However, the projection $\mathbf{AS_2}$ retains additional information from the embedded clustering, in the form of the set of $n \times k$ normalised similarity values, while simultaneously considering the actual term frequencies in $\mathbf{A}$. Consequently, we apply soft spectral co-clustering as described previously, but select $\mathbf{V} = \mathbf{A}^{\mathsf{T}}\mathbf{S_1}$ and $\mathbf{U} = \mathbf{AS_2}$ as our initial pair of factors.

We refine the weights in $\mathbf{U}$ and $\mathbf{V}$ by iteratively updating these factors in order to minimise the divergence between the original term-document matrix $\mathbf{A}$ and the approximation $\mathbf{UV}^{\mathsf{T}}$, as quantified by:

$$D(\mathbf{A}||\mathbf{UV}^{\mathsf{T}}) = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(A_{ij}\log\frac{A_{ij}}{[\mathbf{UV}^{\mathsf{T}}]_{ij}} - A_{ij} + [\mathbf{UV}^{\mathsf{T}}]_{ij}\right) \tag{6.2}$$

This is equivalent to the Kullback-Leibler loss function given in 2.27, with a change of notation such that $\mathbf{W} \equiv \mathbf{U}$ and $\mathbf{H} \equiv \mathbf{V}^{\mathsf{T}}$. To compute the factors, we apply a pair of

1. Compute decomposition of $\mathbf{A_n}$, construct k-dimensional embedding $\mathbf{Z}$ and apply k-means as described in SSC algorithm.
2. Construct matrices $\mathbf{S_1}$ and $\mathbf{S_2}$ from the resulting co-clustering.
3. Generate initial factors $\mathbf{U} = \mathbf{AS_2}$ and $\mathbf{V} = \mathbf{A}^{\mathsf{T}}\mathbf{S_1}$.
4. Update $\mathbf{V}$ using the rule

$$v_{ij} \leftarrow v_{ij} \left[\left(\frac{A_{ij}}{[\mathbf{UV}^{\mathsf{T}}]_{ij}} \right)^{\mathsf{T}} \mathbf{U} \right]_{ij} \tag{6.3}$$

5. Update $\mathbf{U}$ using the rule

$$u_{ij} \leftarrow u_{ij} \left[\frac{A_{ij}}{[\mathbf{UV}^{\mathsf{T}}]_{ij}} \mathbf{V} \right]_{ij} \qquad u_{ij} \leftarrow u_{ij} \frac{U_{ij}}{\sum_{l=1}^{m} U_{lj}} \tag{6.4}$$

6. Repeat from step 4 until convergence.

Figure 6.2: Refined Soft Spectral Co-clustering (RSSC) algorithm.

multiplicative update rules as described by Lee & Seung (1999). The complete *Refined Soft Spectral Co-clustering* (RSSC) algorithm is summarised in Figure 6.2.

6.2.3 Kernel-Based Soft Spectral Clustering

The techniques previously described in this section involve modelling a corpus using a bipartite graph. However, recent spectral clustering methods that make use of the eigen-decomposition of a symmetric affinity matrix have often been shown to produce accurate clusterings (Yu & Shi, 2003). Other advantages of these methods, when compared with those working directly on a term-document matrix $\mathbf{A}$, include a decrease in storage overhead and the ability to make use of an arbitrary kernel function κ to construct a matrix. However, these methods have been designed to produce a hard clustering of documents without the provision of soft membership weights or explicit cluster descriptions. We now introduce a kernel-based technique, analogous to the SSC algorithm, which produces a highly accurate soft co-clustering and meaningful cluster labels.

As with the bipartite approaches, we attempt to minimise the normalised cut of a graph representation of the data. Rather than working on the original vector space model represented by $\mathbf{A}$, our aim is to derive a k-way partition of the weighted unipartite graph encoded in the form of a kernel matrix $\mathbf{K}$. To produce an approximation to the optimal cut of this graph, we begin by constructing the degree-normalised matrix:

$$\mathbf{K_n} = \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}} \tag{6.5}$$

We subsequently set the diagonal values of the matrix to zero as suggested by Ng *et al.* (2001), since we also observe that reducing self-similarity prior to decomposition often leads to an increase in clustering accuracy. A low-dimensional embedding $\hat{\mathbf{X}} \in \mathbb{R}^{n \times k}$ is formed from the eigenvectors corresponding to the k leading eigenvalues, which are normalised to L2 unit length. The k-means algorithm with cosine similarity is then applied to produce a clustering of the rows in $\hat{\mathbf{X}}$. As with SSC, we observe that the use of orthogonal initialisation provides a deterministic means of producing a good set of initial clusters. This embedded clustering, which consists of a grouping of documents only, may be represented by a $n \times k$ binary partition matrix $\mathbf{P}$. The application of L1 column normalisation to this matrix provides equal weighting to each cluster, resulting in a matrix $\hat{\mathbf{P}}$. As with the techniques described previously in this section, we produce a soft-clustering by applying two projections to map the embedded partition to the original space. In this case, the mapping of documents is based on the use of the degree-normalised kernel matrix $\mathbf{K_n}$, which leads to empirical results that are superior to those achieved when using the original kernel matrix $\mathbf{K}$. Specifically, we compute a document membership weight matrix $\mathbf{V} \in \mathbb{R}^{n \times k}$ by computing the projection $\mathbf{V} = \mathbf{K_n}\hat{\mathbf{P}}$. The entry V_{ij} represents the mean affinity between x_i and the documents assigned to C_j:

$$V_{ij} = \frac{\sum_{x_l \in C_j} \mathbf{K_n}_{il}}{|C_j|} \tag{6.6}$$

1. Construct the normalised kernel matrix $\mathbf{K_n} = \mathbf{D}^{-\frac{1}{2}}\mathbf{K}\mathbf{D}^{-\frac{1}{2}}$, and set the diagonal entries $[\mathbf{K_n}]_{ii} = 0$.
2. Compute the k leading eigenvectors of $\mathbf{K_n}$ to form the k-dimensional embedding $\mathbf{X}$.
3. Apply L2-normalisation to the rows of $\mathbf{X}$ to produce $\hat{\mathbf{X}}$.
4. Apply k-means with cosine similarity and orthogonal initialisation to the embedding $\hat{\mathbf{X}}$ to generate a hard clustering $\mathcal{C}_H$.
5. Construct the normalised partition matrix $\hat{\mathbf{P}}$ from the clustering $\mathcal{C}_H$.
6. Form soft clusters by applying the projections $\mathbf{V} = \mathbf{K_n}\hat{\mathbf{P}}$ and $\mathbf{U} = \mathbf{A}\hat{\mathbf{P}}$.

Figure 6.3: Kernel-based Soft Spectral Co-clustering (KSSC) algorithm.

To produce term membership weights, it is necessary refer back to the frequency values in the original term-document matrix $\mathbf{A}$. We apply a projection similar to that used in the SSC algorithm, where feature membership weights are induced from the partition matrix of the clustering of documents. In this case, we construct the matrix $\mathbf{U} \in \mathbb{R}^{m \times k}$, such that $\mathbf{U} = \mathbf{A}\hat{\mathbf{P}}$. The pair $(\mathbf{U}, \mathbf{V})$ may now be interpreted as a soft co-clustering of the data, where highest weighted terms in the columns of $\mathbf{V}$ provides us with labels describing the content of the k clusters. The full procedure is summarised in Figure 6.3.

6.2.4 Experimental Evaluation

In this section, we compare the accuracy afforded by the newly proposed spectral clustering algorithms with that achieved by three popular benchmark algorithms, which are also based on matrix decomposition methods:

BCC:	Bipartite spectral co-clustering with orthogonal initialisation (Dhillon, 2001), using k singular vectors.
NJW:	Ng-Jordan-Weiss k-way spectral clustering (Ng *et al.*, 2001).
NMF:	NMF with the KL-divergence objective function and random initialisation (Lee & Seung, 1999)

While the NJW and KSSC algorithms support the use of an arbitrary kernel function, for document clustering we use a normalised linear kernel as defined in Eqn. 2.36. In our experiments, we select a value for the number of clusters k corresponding to the number of natural classes $\hat{k}$. Clustering accuracy is measured using the external NMI index. When evaluating the non-deterministic NMF and NJW algorithms, we calculate the average accuracy over 100 runs of the clustering algorithm. Since NMI is applied to hard clusterings, for the algorithms generating soft clusterings we produce hard clusters from the membership matrix $\mathbf{V}$ by assigning each document x_i to the cluster C_j such that $j = \arg\max_j V_{ij}$.

Comparison of Clustering Accuracy

Table 6.1 summarises the NMI scores for the six clustering algorithm under consideration, with standard deviations supplied for those that are non-deterministic. The accuracy of the clusterings produced by the SSC algorithm was roughly comparable to that afforded by the bipartite spectral co-clustering method (BCC). By virtue of their ability to work with non-

Dataset	NMF	NJW	BCC	SSC	RSSC	KSSC
bbc	0.77 ± 0.09	0.84 ± 0.04	0.69	0.65	0.84	**0.89**
bbcsport	0.61 ± 0.09	0.79 ± 0.04	0.64	0.64	0.75	**0.87**
classic	0.73 ± 0.06	0.73 ± 0.02	0.69	0.73	**0.84**	0.76
classic3	0.93 ± 0.06	0.93 ± 0.01	0.65	0.62	0.93	**0.95**
cstr	0.67 ± 0.06	0.74 ± 0.03	**0.80**	0.79	0.76	0.79
ng17-19	0.30 ± 0.11	0.29 ± 0.01	0.14	0.20	0.40	**0.44**
ng3	0.79 ± 0.13	0.56 ± 0.05	0.14	0.37	**0.89**	0.61
reuters5	0.55 ± 0.03	0.63 ± 0.03	0.59	0.62	0.57	**0.65**
reviews	0.49 ± 0.04	**0.57 ± 0.02**	0.40	0.41	0.54	0.53
sports	0.52 ± 0.06	0.60 ± 0.03	0.48	0.52	0.56	**0.70**

Table 6.1: Summary of NMI accuracy results for clustering methods based on matrix decomposition, when applied to real-world text datasets.

orthogonal basis vectors, both the NMF and RSSC methods produced clusterings that were often superior to those generated using bipartite methods. However, the RSSC algorithm's use of spectral information to seed well-separated "core clusters" for subsequent refinement leads to a higher level of accuracy on most datasets. We did observe that the NMF and NJW methods exhibit considerable instability, with randomly-initialised NMF particularly resulting in solutions that varied greatly in terms of the clustering accuracy. In contrast, the deterministic nature of the orthogonal initialisation strategy employed by the newly proposed algorithms leads to a single definitive solution. The techniques based on the use of a normalised affinity matrix were frequently superior to those operating on the original term-document matrix, with the KSSC algorithm producing the most accurate solutions on the majority of the datasets. These results are generally significantly better than those afforded by the classical algorithms evaluated in Section 5.3. Only our proposed variation of min-max agglomerative hierarchical clustering with k-means refinement lead to higher accuracy on certain datasets, although KSSC was more consistent when considering all the corpora.

Comparison of Algorithm Efficiency

For the spectral clustering methods described in this chapter, we employ the implementation of the Implicitly Restarted Lanczos Method (Lehoucq *et al.*, 1997) provided by TCT. This makes the computation of a small number of leading eigenvectors or singular vectors far more efficient. For instance, the time required to compute the truncated SVD for a term-document matrix $\mathbf{A}$ of size $m \times n$ is reduced from $O(m^2n)$ to $O(zmn)$, where z is the average number of non-zero entries in each sparse document vector.

In our experiments we observed that, when applying k-means to a spectral embedding, very few iterations are typically required before convergence is achieved. In contrast, due to the slow convergence and computational cost resulting from the repeated application of multiplicative update rules, the NMF and RSSC algorithms exhibited much longer running times. While efficiency may be improved somewhat by using sparse matrix multiplication techniques, we suggest that, for larger text datasets, the KSSC method provides a more pragmatic choice for producing accurate clusterings.

6.2.5 Generating Cluster Labels

Once term membership weights have been produced by a soft co-clustering algorithm, an intuitive approach for generating human-readable cluster labels is to select the set of h terms which have the highest values from each column of the matrix $\mathbf{U}$ (*i.e.* top-rank selection). This results in labels that are *descriptive*, in the sense that they provide a summary of the content of each cluster. However, it may occur that certain terms will be selected as labels for multiple clusters, or that the set of chosen terms will be relatively generic in nature, consisting of words that are not necessarily topic-specific. In such cases, it may be more appropriate to generate cluster labels based on the most *discriminative* terms, which are less ambiguous and serve to highlight the distinctions between clusters.

To select discriminative cluster labels, we propose the application of techniques such as those use in classification tasks. We observe that the selection of discriminative terms based on a fixed set of clusters is closely related to the task of feature selection in classification tasks, where the goal is to identify a subset of features that will improve the ability of a classifier to distinguish between several predefined classes. Thus, many well-known feature selection methods, which have previously been applied in text classification, are relevant in this context. While a variety of options exist for this task, we propose labelling based on two criteria that have been particularly prevalent in the literature:

Information gain (IGAIN): To directly produce discriminative labels from a soft co-clustering such as that produced by KSSC, we suggest the application of a weighting scheme based on *information gain* (Yang & Pedersen, 1997) to the term membership matrix $\mathbf{U}$, which measures the number of bits of information obtained to distinguish between the clusters based on a low or high value in **U** for a given term. Formally, the weight for the i-th term in the cluster C_j is calculated as

$$W_{ij} = E_{ij} - \frac{1}{k} \sum_{l=1}^{k} E_{il} \tag{6.7}$$

where the entropy E_{ij} is given by

$$E_{ij} = -U_{ij} \log_2 U_{ij} - (1 - U_{ij}) \log_2 (1 - U_{ij}) \tag{6.8}$$

A labelling for the cluster C_j may subsequently be found by selecting the h terms with the highest weights in the j-th column of **W**.

χ^2 **measure (CHI):** The Chi-square test has been frequently applied in text classification to measure the level of association between a term and a set of categories. We suggest that, given a hard clustering, the measure may also be used to identify terms that occur frequently in one cluster but seldom in other clusters. Formally, given the i-th term t_i and the cluster C_j, let a_{ij} denote the number of documents in the cluster that contain t_i and let b_{ij} denote the number of documents assigned to other clusters that also contain t_i. In addition, let c_{ij} denote the number of documents in C_j that do not contain t_i, and let d_{ij} denote the number of documents in other clusters that do not contain t_i. A matrix $\mathbf{W} \in \mathbb{R}^{m \times k}$ of term weights is computed based on the standard χ^2 formula:

$$W_{ij} = \frac{n \cdot (a_{ij} d_{ij} - c_{ij} b_{ij})^2}{(a_{ij} + c_{ij}) \cdot (b_{ij} + d_{ij}) \cdot (a_{ij} + b_{ij}) \cdot (c_{ij} + d_{ij})} \tag{6.9}$$

Once again, after computing **W**, cluster labels are generated by choosing the highest ranked terms in each column of the matrix.

Note that this technique is suitable for use in conjunction with any hard clustering algorithm. For techniques such as KSSC, disjoint membership values are found by temporarily assigning each object to the cluster for which it has the highest membership weight in **V**.

To illustrate the effect of applying these techniques, tables 6.2 and 6.3 respectively provide a list of labels selected for clusters produced by the KSSC algorithm on the *bbc* and *bbcsport* corpora. We see that all three labelling methods provide meaningful cluster descriptions. It is interesting to note that these annotations clearly reflect some of the major news events from the period in which the datasets where constructed, including the

	Original		Information Gain		Chi Square	
C_1	company	share	growth	market	growth	oil
	year	bank	bank	oil	analyst	company
	market	economy	economy	price	market	profit
	firm	sale	profit	company	bank	investor
	growth	price	share	analyst	economy	firm
C_2	Labour	Blair	Labour	minister	Labour	party
	elect	tory	elect	government	tory	government
	government	plan	party	MP	elect	secretary
	party	people	tory	prime	minister	MP
	minister	MP	Blair	Conservative	Blair	Conservative
C_3	film	year	film	oscar	star	nomination
	award	show	award	singer	film	actress
	star	include	star	album	award	comedy
	best	actor	actor	band	actor	oscar
	music	nomination	nomination	best	singer	album
C_4	game	against	match	injury	coach	win
	play	champion	cup	game	game	injury
	win	team	win	player	match	player
	player	cup	champion	play	champion	championship
	match	England	coach	team	cup	season
C_5	people	mobile	user	digital	user	online
	technology	phone	technology	phone	technology	download
	user	software	computer	net	computer	internet
	computer	digital	software	online	digital	device
	service	firm	mobile	internet	software	PC

Table 6.2: Labels for a clustering of the *bbc* corpus. Terms were selected using three different strategies: top-rank selection, information gain selection, and χ^2 selection.

2004 Athens Olympics, the 2005 British general election and the 2005 Academy Awards ceremony. However, we observe that using the original term weights leads to more generic labels. For instance, consider the generation of a label for the cluster C_1 on the *bbcsport* corpus. While this cluster primarily consists of articles relating to 'rugby', the terms 'against', 'play' and 'game' are chosen, which could also pertain to any of the other topics in the corpus. In contrast, the IGAIN and CHI methods choose more specific terms such as 'lions' or 'fly-half'. These trends are reflected to varying degrees across all datasets. The CHI method exhibits a particular tendency to choose highly discriminative labels. In general, we observed that the production of intelligible cluster labels was closely related to clustering accuracy. Notably, the labels produced in conjunction with KSSC were generally more meaningful than those produced by the other algorithms due to the greater accuracy afforded by that algorithm.

	Original		Information Gain		Chi Square	
C_1	England	game	rugby	Robinson	rugby	fly-half
	rugby	six	Wales	England	nation	Robinson
	Ireland	against	Ireland	France	Ireland	six
	nation	coach	nation	six	Wales	France
	Wales	play	Scotland	lions	Scotland	flanker
C_2	Olympics	Athens	Olympics	drug	Olympics	gold
	athlete	European	athlete	European	athlete	trial
	race	year	indoor	gold	Athens	200m
	indoor	champion	race	IAAF	indoor	drug
	world	drug	Athens	medal	sprinter	race
C_3	open	Australian	seed	7-6	seed	7-5
	seed	set	open	tennis	6-3	7-6
	play	first	6-3	7-5	tennis	Wimbledon
	win	final	6-4	Roddick	open	6-1
	match	6-3	Australian	6-1	6-4	Australian
C_4	club	play	Chelsea	football	Chelsea	Manchester
	Chelsea	game	club	boss	Arsenal	football
	player	united	Arsenal	Manchester	league	FA
	league	football	league	manager	boss	manager
	Arsenal	manager	united	Liverpool	club	united
C_5	test	Pakistan	cricket	India	cricket	series
	cricket	wicket	one-day	test	wicket	batsman
	series	India	Pakistan	bowl	one-day	bat
	one-day	Australia	wicket	bowler	bowler	Pakistan
	play	first	series	Australia	bowl	over

Table 6.3: Labels for a clustering of the *bbcsport* corpus. Terms were selected using three different strategies: top-rank selection, information gain selection, and χ^2 selection.

6.2.6 Summary

In this section, we described methods based on spectral analysis that can yield accurate, interpretable co-clusterings on high-dimensional text datasets. We examined the idea of using spectral clustering to provide a good initial start point for non-negative matrix factorisation, which can often produce poor, unstable results when standard stochastic initialisation techniques are employed. Section 6.2.3 described a kernel-based approach, the KSSC algorithm, which frequently out-performs popular clustering methods based on matrix decomposition. Evaluations conducted on real-world text corpora demonstrate that these methods can lead to the improved identification of overlapping clusters, while simultaneously producing document and term weights that are amenable to human interpretation. Furthermore, we proposed that highly discriminative cluster labels can be generated by applying supervised feature selection measures to the output of document clustering algorithms. While we have specifically considered the information gain and χ^2 measures, we suggest that other similar selection methods may also prove useful.

6.3 Practical Solutions for Diagonal Dominance Reduction

We now shift our focus to the state-of-the-art kernel learning methods introduced in Section 2.8. These methods are composed of two key components: a generic learning algorithm, and a kernel function which assesses the similarity between objects in the kernel space. In many domains, it will often be the case that the average similarity of one object to another will be small when compared to the "self-similarity" of the object to itself. This characteristic of many popular kernel functions has proved to be problematic for supervised kernel methods (Schölkopf *et al.*, 2002). If, for a given kernel function, self-similarity values are large relative to between-object similarities, the Gram matrix of this kernel will exhibit *diagonal dominance*, which can significantly impair the accuracy of a classifier.

In the remainder of this chapter, we examine the implications of this phenomenon for document clustering when employing centroid-based kernel methods. We then propose a number of practical strategies for addressing this problem, and evaluate their ability to generate more accurate and stable clustering solutions on text data.

6.3.1 Dominant Diagonals in Supervised Learning

It has been observed that the performance of the SVM classifier can be poor in cases where the diagonal values of the Gram matrix are large relative to the off-diagonal values (Cancedda *et al.*, 2003; Schölkopf *et al.*, 2002). This problem, sometimes referred to as diagonal dominance in machine learning literature, frequently occurs when certain kernel functions are applied to data that is sparse and high-dimensional in its explicit representation. This phenomenon is particularly evident in text mining tasks, where linear or string kernels can often produce diagonally dominated Gram matrices. However, it can also arise with other kernel functions, such as when employing the Gaussian kernel with a small smoothing parameter, or when using domain-specific kernels for learning tasks in image retrieval (Tao *et al.*, 2004) and bioinformatics (Saigo *et al.*, 2004). These cases are all characterised by the tendency of the mean of the diagonal entries of the kernel matrix $\mathbf{K}$ to be significantly larger than the mean of the off-diagonal entries, resulting in a *dominance ratio*:

$$\frac{\frac{1}{n}\sum_i K_{ii}}{\frac{1}{n(n-1)}\sum_{i,j,i\neq j} K_{ij}} \gg 1 \tag{6.10}$$

We can interpret this to mean that the set of data objects are approximately orthogonal to one another in the kernel space. In many cases, a classifier applied to such a matrix will effectively memorise the training data, resulting in severe overfitting (Schölkopf *et al.*, 2002).

An unfortunate characteristic of this problem is that matrices which are strongly diagonally dominated will be positive semi-definite and measures to reduce this dominance run the risk of rendering a matrix indefinite, so that it no longer represents a valid Mercer kernel. Consequently, there is a tension between diagonal dominance on the one hand and the requirement that the matrix be positive semi-definite on the other.

6.3.2 Dominant Diagonals in Kernel Clustering

The phenomenon of diagonal dominance also has implications for centroid-based kernel clustering methods, such as the kernel k-means algorithm described in Section 2.8.1. Observe that, when calculating the square distance between a centroid μ_a and a document $x_i \in C_a$, Eqn. 2.32 can be separated as follows:

$$||\phi(x_i) - \mu_c||^2 = K_{ii} + \frac{\sum_{x_j, x_l \in C_a} K_{jl}}{|C_a|^2} - \frac{2\sum_{x_j \in C_a - \{x_i\}} K_{ij}}{|C_a|} - \frac{2K_{ii}}{|C_a|} \tag{6.11}$$

If $\mathbf{K}$ is diagonally dominated, the last term in the above expression will be large. This has the effect that x_i will appear to be close to the centroid of C_a and distant from the remaining clusters, regardless of the affinity between x_i and the other documents assigned to C_a. Consequently, even with random cluster initialisation, few subsequent reassignments will be made and the algorithm will converge to a poor local solution.

The problem of dominant self-similarity has previously been shown to adversely affect centroid-based clustering algorithms in high-dimensional feature spaces (Dhillon *et al.*, 2002a). Therefore, it is unsurprising that similar problems should arise when applying their kernel-based counterparts using kernel functions that preserve this sparsity. Dhillon *et al.* (2004a) observed that the effectiveness of kernel k-means can be significantly impaired when document-cluster distances are dominated by self-similarity values.

A normalised linear kernel (2.36) represents an intuitive choice when working with text data, due to its equivalence to the cosine metric (2.4). However, a matrix constructed using this function will typically suffer from diagonal dominance due to the sparsity of the vector space representation for real-world text datasets. Thus, while we will always have $S_{ii} = 1\, \forall i$, it will often be the case for sparse text data that $S_{ij} \ll 1$ for $i \neq j$. When applying kernel k-means using such a matrix, the large diagonal entries may prevent the identification of coherent clusters. Often incorrect assignments in an initial clustering

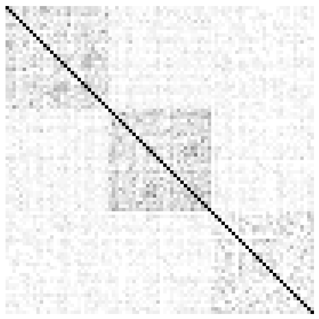


Figure 6.4: A normalised linear kernel matrix constructed on a subset of 90 documents from the *classic3* dataset, which exhibits diagonal dominance.

solution will fail to be subsequently rectified, since large self-similarity values may obscure similarities between pairs of documents belonging to the same natural class.

To illustrate this problem, we consider a small subset of the frequently used *classic3* dataset, which contains three relatively well-separated natural classes. From each class, we randomly sample 30 documents, resulting in a 1203×30 term-document matrix, such that 7% of the entries are non-zero. When constructing a normalised linear kernel on this data, the affinity values in the matrix reflect the reasonably well-separated nature of the original classes: the mean affinity between documents is 0.06, the mean affinity between documents in the same class is 0.11, while the mean affinity between documents belonging to different classes is 0.03. As in all cases where this kernel is employed, the mean self-similarity value is 1.0. Due to the separation between classes and small size of the dataset, we would expect to identify the three groups in the data with relative ease. However, the sparsity of the original high-dimensional space leads to pairwise affinity values that are small relative to very low relative to the self-similarity values. As a result, the matrix exhibits significant diagonal dominance, which is evident from Figure 6.4. When applying kernel k-means using this kernel over 1000 runs, we observe that the algorithm frequently terminates prematurely. In fact, 55% of the algorithm executions ended after four or less iterations. Consequently, the accuracy of the partitions generated was poor, with a mean NMI score of 0.11 indicating that they were usually little better than random. In addition, the average agreement between each pair of clusterings, measured using ANMI, was only 0.04, indicating that the output of the algorithm was highly unstable. This simple example clearly shows that, when employing kernel k-means on sparse data, dominant self-similarity values can often prevent the identification of the natural classes, even for small datasets containing well-separated structures. While this example represents a pathological case, the experimental evaluation presented in Section 6.3.5 indicates that diagonal dominance can similarly affect larger, more complex datasets.

6.3.3 Reducing Diagonal Dominance

We now present four practical strategies for addressing the issues raised by diagonal dominance in centroid-based kernel clustering.

Diagonal Shift (DS)

To reduce the influence of large diagonal values, Dhillon *et al.* (2004a) proposed the application of a negative shift to the diagonal of the Gram matrix. Specifically, a multiple σ of the identity matrix is added to produce

$$\mathbf{K_{DS}} = \sigma \mathbf{I} + \mathbf{S} \tag{6.12}$$

The parameter σ is a negative constant, typically selected so that the trace of the kernel matrix is approximately zero. For a normalised linear kernel matrix with trace equal to n, this will be equivalent to subtracting 1 from each diagonal value, thereby eliminating the first and last terms from the document-centroid distance calculation (6.11).

The diagonal shift technique is equivalent to the addition of a negative constant to the eigenvalues of $\mathbf{S}$. As a result, $\mathbf{K_{DS}}$ will no longer be positive semi-definite and the kernel k-means algorithm will not be guaranteed to converge when applied to this matrix. Figure 6.5 compares the trailing eigenvalues for the matrix shown in Figure 6.4, before and after applying a shift of $\sigma = -1$. Notice that the modification of the diagonal entries has the effect of shifting each eigenvalue down by 1. Consequently, a large number of eigenvalues are below zero, signifying that the modified matrix is indefinite.

The application of diagonal shifts to Gram matrices has previously proved useful in supervised kernel methods. However, rather than seeking to reduce diagonal dominance, authors have most frequently used the technique to ensure that a kernel matrix is positive semi-definite. Several authors have previously proposed the addition of a non-negative constant to transform indefinite symmetric matrices into valid kernels. Unfortunately, this will have the side effect of increasing the dominance ratio. Specifically, Saigo *et al.* (2004) suggested adding a shift $\sigma = |\lambda_n|$, where λ_n is the negative eigenvalue of the kernel matrix with largest absolute value. It is evident from Figure 6.5 that a positive shift will

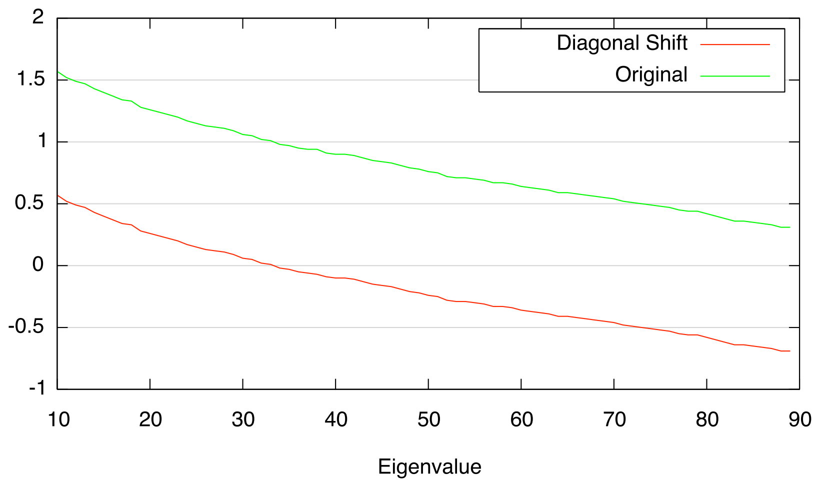


Figure 6.5: Set of leading eigenvalues in the range $[10, 90]$ for a normalised linear kernel matrix constructed on a subset of 90 documents from the *classic3* corpus.

often result in a matrix that is once again diagonally dominated. In addition, computing a full spectral decomposition for a large term-document matrix will often be impractical, although Wu *et al.* (2005) did suggest an estimation approach for approximating λ_n.

Subpolynomial Kernel With Empirical Kernel Map (SPM)

To address the problems introduced by large diagonals in SVMs, Schölkopf *et al.* (2002) suggested the reduction of the dynamic range of the kernel matrix by transforming its entries using a *subpolynomial kernel* function. The application of such a function has the range of "flattening" the gap between high affinity values (which will predominately correspond to self-similarities for certain kernels when applied to text data) and low affinity values. Formally, a subpolynomial kernel function is defined as

$$\kappa_{SP}(x_i, x_j) = \langle x_i, x_j \rangle^p \tag{6.13}$$

where $0 < p < 1$ is a user-defined *degree* parameter. For a corresponding kernel matrix $\mathbf{K}_{SP}$, decreasing the value of the degree p will have the effect of reducing the ratio of diagonal entries to off-diagonal entries. Unlike the diagonal shift technique, the non-linear transformation introduced by Eqn. 6.13 directly modifies the pair-wise affinities between the off-diagonal entries in $\mathbf{S}$, which may potentially distort the underlying cluster structure.

An important issue that must be addressed when using a subpolynomial kernel is the selection of the parameter p. If the value is too large the Gram matrix will remain diagonally dominated. On the other hand, if the value of p is too small, all documents will tend to become equally similar, thereby obscuring the true structures in the data. Schölkopf *et al.* (2002) suggested the use of standard cross-validation techniques for selecting p. However, this may not be feasible in cases where other key parameters such as the number of clusters k must also be determined by repeatedly clustering the data.

Since the kernel matrix κ_{SP} may no longer be a valid kernel, it may be desirable to apply some technique to render the matrix positive definite. Once such approach is the *empirical kernel map* method (Schölkopf & Smola, 2001). This involves mapping each document x_i to an n-dimensional feature vector

$$\phi_m(x_i) = (\kappa(x_i, x_1), \ldots, \kappa(x_i, x_n))^{\mathsf{T}} \tag{6.14}$$

By using this feature representation, we can derive a positive definite kernel matrix by simply computing the dot products

$$\mathbf{K_{SPM}} = \mathbf{K_{SP}}\mathbf{K_{SP}}^{\mathsf{T}} \tag{6.15}$$

In practice, normalising all rows of $\mathbf{K_{SP}}$ to unit Euclidean length prior to computing the dot product leads to significantly better results.

Diagonal Shift With Empirical Kernel Map (DSM)

While the empirical map technique was used by Schölkopf *et al.* (2002) to produce a valid kernel from the matrix of a subpolynomial kernel, this approach can be applied in combination with other reduction methods. Thus, even if we alter the diagonal of the kernel matrix in an arbitrary manner so that it becomes indefinite, we may still recover a positive definite matrix that will guarantee convergence for the kernel k-means algorithm.

Here we consider the possibility of applying a negative shift to minimise the trace of the kernel matrix as described previously. This is followed by the construction of the empirical map $\mathbf{K_{DSM}} = \mathbf{K_{DS}}\mathbf{K_{DS}}^{\mathsf{T}}$, after normalising the rows of $\mathbf{K_{DS}}$ to unit length. While this approach does reduce the dominance ratio (6.10), it should be noted that the application of the dot product will produce a kernel matrix with trace greater than zero. Thus, it can be viewed as providing a balance between reducing dominance and ensuring that the kernel matrix remains positive semi-definite.

Algorithm Adjustment (AA)

When attempting to apply supervised kernel methods to indefinite kernel matrices, Wu *et al.* (2005) distinguished between two fundamental strategies: *spectrum transformation* approaches that perturb the original matrix to produce a valid Gram matrix, and *algorithmic* approaches that involve altering the formulation of the learning algorithm. A similar distinction may be made between diagonal dominance reduction techniques. We now propose a new algorithmic approach that involves adjusting the kernel k-means algorithm described by Schölkopf *et al.* (1998) to eliminate the influence of self-similarity values.

If one considers the distance between a document x_i and the cluster C_a to which it has been initially assigned, a dominant diagonal will lead to a large value in the third term of Eqn. 2.32. This will often cause x_i to remain in C_a during the reassignment phase, regardless of the affinity between x_i and the other documents in C_a. A potential method for alleviating this problem is to reformulate the reassignment step as a "split-and-merge" process, where self-similarity values are not considered. Rather, we assign each document to the nearest centroid, where the document itself is excluded during centroid calculation.

Formally, each document x_i is initially removed from its cluster C_a, leaving a cluster $C_a - \{x_i\}$ with centroid denoted $\mu_{a'}$. For each alternative candidate cluster C_b, $b \neq a$, we consider the gain achieved by reassigning x_i to C_b rather than returning it back to C_a. This gain is quantified by the expression:

$$\Delta_{ab}(x_i) = ||\phi(x_i) - \mu_{a'}||^2 - ||\phi(x_i) - \mu_b||^2 \tag{6.16}$$

By considering the definition of squared object-centroid distance in a kernel space (2.32), it is apparent that the diagonal value K_{ii} is not considered in the computation of Δ_{ab}. If $\max_b \Delta_{ab}(x_i) > 0$, then x_i is reassigned to the cluster C_b which results in the maximal gain. Otherwise, x_i remains in cluster C_a. As with the standard batch formulation of kernel k-means, centroids are only updated after all n documents have been examined. A summary of the procedure is given in Figure 6.6.

This strategy could potentially be applied to improve the performance of the standard k-means algorithm in sparse spaces where self-similarity values have undue influence. However, the repeated adjustment of centroids in a high-dimensional space is likely to be impractical. Fortunately, for kernel k-means, we can efficiently compute Δ_{ab} by caching the contribution of each document to the common term in Eqn. 2.32, making it unnecessary to recalculate the term in its entirety when evaluating each document for reassignment.

1. Select k arbitrary initial clusters $\{C_1, \ldots, C_k\}$.
2. For each object x_i and cluster C_b such that $x_i \in C_a$ and $C_a \neq C_b$, compute the potential gain for moving x_i to C_b:

$$\Delta_{ab}(x_i) = ||\phi(x_i) - \mu_{a'}||^2 - ||\phi(x_i) - \mu_b||^2$$

3. If $\max_b \Delta_{ab}(x_i) > 0$, reassign x_i to $C_b = \arg\max_b \Delta_{ab}(x_i)$. Otherwise leave x_i in C_a.
4. Repeat from Step 2 until some termination criterion is satisfied.

Figure 6.6: Kernel k-means with algorithm adjustment (AA).

6.3.4 Comparison of Reassignment Behaviour

Dhillon *et al.* (2002a) observed that spherical k-means often becomes trapped at an initial clustering, where the similarity of any document to its own centroid is much greater than its similarity to any other centroid. As discussed previously, a diagonally dominated kernel matrix frequently elicits similar behaviour from the kernel k-means algorithm. Consequently, the algorithm will converge after relatively few reassignments have been made to a local solution that is close to the initial partition. If the initial clusters are randomly selected, it is possible that the final clustering will be little better than random. In addition, multiple runs may produce significantly different partitions of the same data.

To gain a clearer insight into this problem, we examine the reassignment behaviour resulting from the application of each of the reduction strategies proposed in the last section. Figure 6.7 illustrates the mean percentage of accepted reassignments occurring during the first 10 iterations of the kernel k-means algorithm when applied to the data represented by the matrix shown in Figure 6.4. It is evident that applying the algorithm to the original dominated matrix results in significantly fewer reassignments, which can be viewed as a cursory search of the solution space. It is interesting to note that these reassignment patterns are replicated to varying degrees across all the datasets described in Chapter 5. In this example, the increased number of reassignments coincides with higher accuracy and stability. For instance, the application of the DS strategy leads to mean NMI and ANMI scores of 0.84 and 0.93 respectively, indicating that kernel k-means is frequently successful in the identification of the three underlying classes.

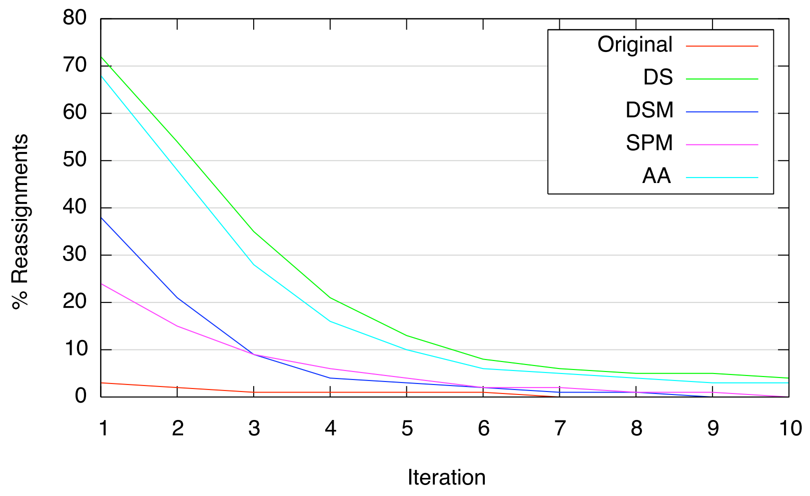


Figure 6.7: Average percentage of accepted reassignments made during the first 10 iterations of the kernel k-means algorithm, when applied to a subset of 90 documents from the *classic3* corpus.

6.3.5 Experimental Evaluation

We now evaluate the degree to which the diagonal dominance reductions strategies described previously can improve our ability to accurately and consistently group text documents when working with real-world corpora. Table 6.4 provides details of the normalised linear kernel matrices for the datasets used in our experiments, including mean affinity values for three cases: all pairs of documents, pairs of documents associated with the same natural class, and pairs belonging to different classes. The corresponding dominance ratios indicate that the kernel matrices generally exhibit significant diagonal dominance.

Dataset	Mean Affinity			Ratio
	Overall	Intra-Class	Inter-Class	
bbc	0.04	0.07	0.03	24.18
bbcsport	0.06	0.09	0.05	16.19
classic	0.02	0.04	0.01	50.73
classic3	0.03	0.05	0.01	40.53
cstr	0.06	0.10	0.04	17.10
ng17-19	0.03	0.04	0.03	29.44
ng3	0.03	0.05	0.02	30.93
reuters5	0.04	0.09	0.02	23.83
reviews	0.04	0.06	0.03	27.18
sports	0.04	0.06	0.03	27.75

Table 6.4: Details of normalised linear kernel matrices constructed on real-world datasets.

For each approach, we performed 200 trials of the clustering procedure, which was randomly initialised in each case. We set the number of clusters k to correspond to the number of natural classes in the data. For experiments using a subpolynomial kernel, we tested values for the degree parameter from the range $[0.4, 0.9]$. We observed that values for $p < 0.4$ invariably resulted in excessive flattening of the range of values in the kernel matrix, producing partitions that were significantly inferior to those generated on the original matrix.

To compare the accuracy of the clustering techniques, we again employ the NMI measure. As discussed in Section 2.3, many clustering algorithms such as k-means and its variations are particularly sensitive to initial starting conditions. This makes them prone to converging to different local minima when using a stochastic initialisation strategy. Therefore, when selecting a diagonal reduction method, we seek to identify a robust approach that will allow us to consistently produce accurate, reproducible clusterings. In our experiments, we assessed the stability of each candidate method using the average normalised mutual information (ANMI) between the set of all partitions generated on each dataset, calculated using Eqn. 5.1.

Analysis of Results

Our experiments indicate that all of the reduction approaches under consideration have merit. In particular, Table 6.5 shows that the AA and DS methods yield improved clustering accuracy in all cases. Generally, we observed that diagonal dominance reduction has

Dataset	Orig.	DS	DSM	AA	SPM
bbc	0.82 ± 0.07	0.84 ± 0.07	0.81 ± 0.08	**0.85 ± 0.06**	0.84 ± 0.06
bbcsport	0.71 ± 0.11	**0.82 ± 0.08**	0.78 ± 0.07	0.80 ± 0.08	0.75 ± 0.09
classic	0.73 ± 0.03	0.73 ± 0.03	0.71 ± 0.02	**0.74 ± 0.02**	0.72 ± 0.02
classic3	0.93 ± 0.06	0.93 ± 0.06	0.92 ± 0.08	**0.94 ± 0.06**	0.91 ± 0.06
cstr	0.70 ± 0.05	**0.75 ± 0.04**	0.70 ± 0.02	0.74 ± 0.04	0.71 ± 0.03
ng17-19	0.38 ± 0.13	0.41 ± 0.14	0.39 ± 0.10	0.42 ± 0.13	**0.46 ± 0.11**
ng3	0.81 ± 0.11	0.83 ± 0.11	0.68 ± 0.08	**0.84 ± 0.10**	0.79 ± 0.11
reuters5	0.58 ± 0.05	0.59 ± 0.04	0.62 ± 0.05	0.59 ± 0.04	**0.60 ± 0.06**
reviews	0.57 ± 0.07	**0.59 ± 0.05**	0.32 ± 0.06	**0.59 ± 0.05**	0.46 ± 0.08
sports	0.66 ± 0.06	0.66 ± 0.05	0.53 ± 0.07	**0.67 ± 0.05**	0.53 ± 0.06

Table 6.5: Summary of NMI accuracy results (mean and standard deviation) for kernel clustering with dominance reduction, when applied to real-world text datasets.

Dataset	Orig.	DS	DSM	AA	SPM
bbc	0.83	0.88	0.86	**0.90**	**0.90**
bbcsport	0.63	0.82	**0.83**	0.78	0.74
classic	0.89	0.89	0.81	**0.90**	0.79
classic3	0.96	0.97	0.95	**0.97**	0.96
cstr	0.71	**0.81**	0.79	**0.81**	0.77
ng17-19	0.45	0.52	0.60	0.52	**0.63**
ng3	0.81	0.84	0.68	**0.86**	0.82
reuters5	0.74	0.78	**0.92**	0.78	0.87
reviews	0.76	0.82	**0.86**	0.81	0.70
sports	0.72	0.74	**0.78**	0.74	0.72

Table 6.6: Summary of ANMI stability results for kernel clustering with dominance reduction, when applied to real-world text datasets.

a greater effect on some datasets than on others. While the difference in reassignment behaviour after reduction is less pronounced on datasets such as *classic3*, there is no strong correlation between the distribution of the affinity values in the kernel matrix and the increase in accuracy. However, it is apparent from Table 6.6 that applying kernel k-means to a dominated kernel matrix consistently results in poor stability. It is clear that the restriction placed on the reassignment behaviour in these cases frequently results in less deviation from the initial random partition, thereby increasing the overall disagreement between solutions. We now discuss the performance of each of the approaches individually.

Diagonal Shift (DS). Table 6.5 shows that the negative diagonal shift approach frequently produced clusterings that were more accurate than those generated on the original dominated kernel matrices. As noted in Section 6.3.3, this method provides no guarantee of convergence. However, our results support the assertion made by Dhillon *et al.* (2004a) that, in practice, lack of convergence may not always be a problem. Frequently we observed that a comparatively stable partition is identified after a relatively few number of iterations. At this stage the algorithm proceeds to oscillate indefinitely between two nearly identical solutions without ever attaining convergence. To address this issue, we chose to terminate the reassignment process after five consecutive oscillations were detected and select the solution with the lowest distortion. This had no noticeable adverse effect on clustering accuracy.

Diagonal Shift With Empirical Kernel Map (DSM). While the application of the empirical kernel map technique subsequent to a diagonal shift does guarantee con-

Dataset	$p = 0.4$	$p = 0.5$	$p = 0.6$	$p = 0.7$	$p = 0.8$	$p = 0.9$
bbc	0.83	**0.84**	0.83	**0.84**	**0.84**	0.83
bbcsport	0.67	0.70	0.73	0.75	**0.76**	**0.76**
classic	0.71	**0.73**	**0.73**	0.72	0.72	0.71
classic3	0.89	0.90	0.89	0.91	0.91	**0.92**
cstr	0.66	0.67	0.69	0.71	0.72	**0.72**
ng17-19	0.44	0.45	0.45	**0.46**	0.43	0.41
ng3	**0.85**	**0.85**	0.82	0.79	0.79	0.74
reuters5	0.56	0.56	0.59	0.60	0.61	**0.62**
reviews	**0.55**	0.52	0.48	0.46	0.44	0.42
sports	0.51	0.52	0.53	0.53	0.54	**0.55**

Table 6.7: Comparison of mean NMI accuracy results for kernel clustering of real world text-datasets, when using a subpolynomial kernel with degree parameter $p \in [0.4, 0.9]$.

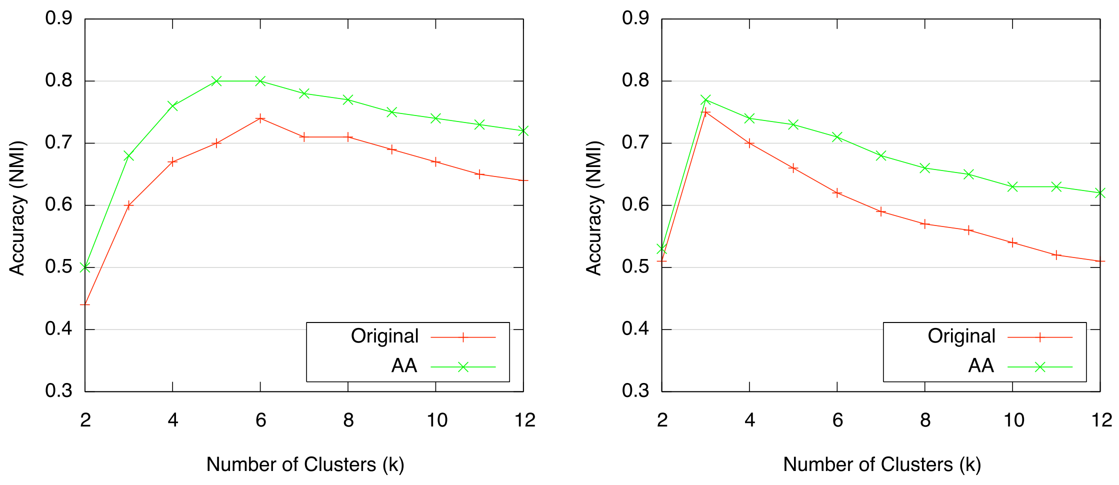
vergence after relatively few iterations, the map also has the effect of increasing the dominance ratio, resulting in accuracy gains that are not as significant as those achieved by the other strategies. The higher level of consistency between solutions generated using this method does suggest that it represents a reasonable trade-off between accuracy and stability. However, there remains the additional computational expense of constructing the matrix $\mathbf{K}_{DSM}$, which requires $O(n^3)$ time.

Subpolynomial Kernel With Empirical Kernel Map (SPM). For the subpolynomial kernel reduction method, our experimental findings underline the difficulty of setting the degree parameter p. The gains in accuracy resulting from this approach were significant, though less consistent than those achieved by the other methods. On certain datasets, such as the *ng3* and *reviews* collections, specific values of p lead to a large improvement in both accuracy and stability, while in other cases there was little or no improvement. This suggests that the alteration of cluster structure induced by the subpolynomial function may prove beneficial in some cases, but not in others. Therefore, while a value of $p = 0.7$ lead to the highest mean accuracy when considering all datasets, from the results shown in Table 6.7 we conclude that the selection of a value for p is largely data dependent. Once again, the expense of calculating the empirical map must be taken into consideration when making use of this reduction method. As with the DSM approach, the application of the empirical map resulted in a marked increase in cluster stability.

Algorithm Adjustment (AA). The adjusted kernel k-means algorithm yielded improvements in accuracy that were marginally better than those produced by the diagonal shift method (DS), while also achieving slightly higher ANMI stability scores. The correlation between the two methods is understandable given their similar reassignment behaviour. This stems from the fact that applying a negative diagonal shift of $\sigma = -1$ to a matrix with trace equal to n effectively eliminates the dominant last term in Eqn. 6.11, leading to document-centroid distances that are approximately the same as those achieved using the "split-and-merge" adjustment. It should be noted that, while the AA reduction method frequently failed to achieve complete convergence, the oscillation detection technique described previously resolved this problem satisfactorily on all datasets.

From our experiments, it is apparent that the DS and AA techniques both represent efficient strategies for reducing diagonal dominance and thereby improving the output of centroid-based kernel clustering algorithms. Of these, the latter leads to slightly greater improvement in clustering accuracy.

While the results shown previously relate to experiments that involve using a value k corresponding to the number of natural classes in the data, we have also noted that comparable improvements in accuracy and stability are apparent on all datasets when using alternative values for the number of clusters. As an example, Figure 6.8 shows a comparison of the mean NMI scores for the original kernel k-means algorithm and the new AA technique, covering a range of values of k. In several cases, such as for the *cstr* dataset



(a) *bbcsport* dataset (b) *cstr* dataset

Figure 6.8: Plot of mean NMI accuracy scores for clusterings generated by kernel k-means and adjusted kernel k-means, for values of $k \in [2, 10]$.

shown in Figure 6.8(b), the gain in accuracy becomes more pronounced as the number of clusters increases. This is largely due to the frequent presence of smaller clusters in the initial clustering solutions, from which incorrectly assigned documents fail to escape due to the strong influence of self-similarity.

6.3.6 Summary

We have considered a range of practical solutions to the issues introduced by diagonally dominated kernel matrices in unsupervised kernel methods. Furthermore, we have demonstrated the effectiveness of the solutions when performing the task of document clustering. From our evaluation, it is apparent that the presence of disproportionately large self-similarity values precipitates a reduction in the number of reassignments made by the kernel k-means algorithm. This may limit the extent to which the solution space is explored, causing the algorithm to become stuck close to its initial state. In cases where the initialisation strategy is stochastic or unsuitable, this can result in an appreciable decrease in accuracy and cluster stability. To remedy this, in Section 6.3.3 we proposed a selection of strategies for reducing the effects of dominance. The diagonal shift and adjusted kernel k-means techniques represent particularly appealing options for this task, particularly in light of the negligible increase in running time resulting from their application.

An interesting point arising from our experiments is that techniques such as the diagonal shift strategy, which work on indefinite matrices, can still produce good clusterings and can be terminated after a tractable number of iterations without any significant decrease in clustering accuracy. This indicates that kernel k-means may not be as sensitive to the requirement for a positive semi-definite kernel matrix as supervised algorithms, such as the SVM classifier.

We suggest that the diagonal dominance strategies described here may also be useful in improving other centroid-based kernel clustering methods, such as the weighted kernel k-means algorithm (Dhillon *et al.*, 2004b) or a kernel-based formulation of the bisecting k-means algorithm given in Section 2.4.2. In addition, we believe that these strategies will have merit when applying kernel clustering in other domains, such as bioinformatics and image retrieval, where the ratio of diagonal to off-diagonal entries in the kernel matrix will often be significantly higher.

Chapter 7

Aggregating Information from Multiple Clusterings

7.1 Introduction

A compelling notion in machine learning research is the idea that, by combining the output of a group of algorithms, a superior solution may be found for a given learning problem. Ensemble classification techniques, such as bagging and boosting, are the most prominent examples of this idea (Breiman, 1996). Similar techniques can be employed in unsupervised learning problems, where analysing information obtained from multiple clustering solutions can provide us with a better overall view of the structure of a dataset. In recent cluster analysis research, it has been shown that this type of information can be used in a number of ways. For instance, the ensemble clustering algorithms described in Section 2.9 can produce more accurate, stable clustering solutions by combining a collection of base clusterings. Another previously explored area is that of stability analysis, which is discussed in Section 3.4. Techniques based on this concept use the information aggregated over many runs of a clustering algorithm to produce a robust assessment regarding the validity of a given clustering model.

Even with the availability of significant computing resources, the scalability issues arising from the dimensionality and size of the data models used in text mining tasks remain a constant concern. A serious drawback, common to both ensemble clustering and stability analysis, is the significant computational cost incurred when generating and analysing multiple clusterings. In both cases, a larger collection of base clusterings is desirable to provide more conclusive evidence regarding the cluster structures. A user employing these methods on large corpora must therefore deal with the classic trade-off between speed and precision. Many users will be reluctant to compromise on either algorithm scalability or accuracy. As a consequence, neither aggregation method has been widely adopted for use in document clustering tasks.

To address these concerns, we now propose efficient strategies for aggregating information from multiple clusterings, which maintain algorithm performance while significantly reducing running time. In Section 7.2, we consider a prediction-based approach for stability analysis suitable for estimating the number of clusters in a document collection. In Section 7.3, we shift our focus to the related problem of rendering ensemble clustering techniques more practical for use on text datasets, without negatively impacting upon algorithm accuracy or stability.

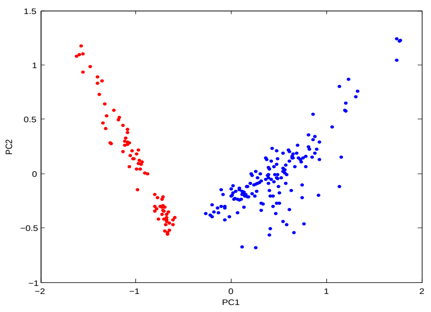
7.2 Efficient Prediction-Based Cluster Validation

Validation techniques based on stability analysis have been shown to provide an effective means of determining the optimal number of clusters $\hat{k}$ in a dataset (Lange *et al.*, 2004). For small datasets, stability-based validation techniques offer a highly attractive option for inferring a value for $\hat{k}$. However, as the number of dimensions grows, the time required to repeatedly apply an algorithm such as k-means will greatly increase. The number of objects n will also be a limiting factor, since a larger value for n will substantially increase the computational cost of the clustering and the stability assessment procedures, which typically run in $O(n^2)$ time or slower. Given these scalability issues, it is unsurprising that stability analysis methods have rarely been applied to large-scale datasets such as text corpora. In this section, we tackle these issues by proposing an efficient prediction-based validation scheme based on the idea of *prototype reduction*, which involves producing a smaller set of objects or prototypes to represent a given dataset.

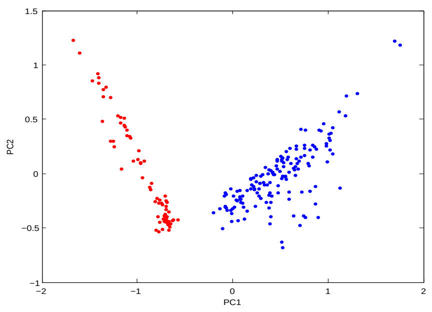
7.2.1 Background

Prediction-Based Validation for Text Data

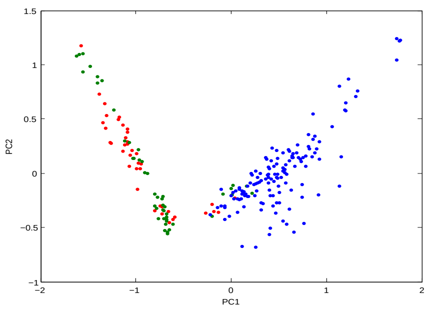
Prediction-based methods for stability analysis are motivated by the concept of prediction accuracy in supervised learning (Tibshirani *et al.*, 2001). To illustrate the relevance of this idea in the context of model selection for document clustering, we consider a small subset



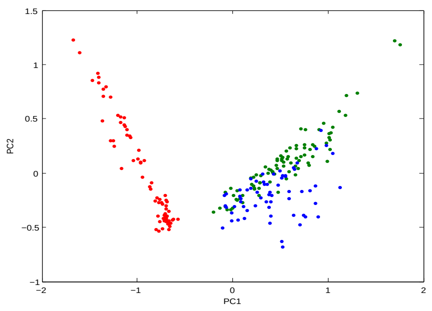
(a) Training ($k = 2$)



(b) Testing ($k = 2$)



(c) Training ($k = 3$)



(d) Testing ($k = 3$)

Figure 7.1: Plot of the first two Principal Components of a subset of 450 documents from the 20NG collection, showing partitions generated when applying two iterations of prediction-based validation.

of the 20NG collection, consisting of 300 documents from the 'cryptography' newsgroup and 150 documents from the 'hockey' newsgroup. Figures 7.1(a) and 7.1(b) show training and testing partitions of this data generated for $k = 2$. Clearly, a suitable classifier built upon the groups in the former is likely to be successful at predicting the assignment of documents in the latter partition. In contrast, the partitions of the same sets for $k = 3$, as shown in figures 7.1(c) and 7.1(d), are significantly different. This makes it unlikely that a classifier constructed on the former will accurately predict the latter. If these patterns are frequently replicated over many different splits of the data, it is reasonable to conclude that $k = 2$ represents a more appropriate choice for the number of clusters than $k = 3$.

Prototype Reduction

Prototype reduction techniques have been extensively used in supervised learning for tasks involving large datasets, typically in conjunction with a nearest-neighbour classifier. These techniques are concerned with producing a minimal set of objects or prototypes to represent the data, while ensuring that a classifier applied to this set will perform approximately as well as on the original dataset. In the literature, these techniques are generally divided into two categories: *prototype selection* techniques seek to identify a subset of representative objects from the original data, while *prototype extraction* techniques involve the creation of an entirely new set of objects. A comprehensive overview of supervised reduction schemes has been provided by Bezdek & Kuncheva (2001).

Many reduction techniques are computationally intensive, often involving clustering-like procedures to identify relevant prototypes. In contrast, Hamamoto *et al.* (1997) proposed a simple stochastic technique (BTS), which is based on bootstrap editing. Initially, a random sample of n' seed objects is drawn from the dataset. Each seed object is then replaced by a new prototype constructed from the mean of its p-nearest neighbours and the seed itself. A 1-NN classifier is subsequently applied to the new set of n' prototypes. The entire process may be repeated multiple times to give more robust results. Kim & Oommen (2005) described a novel framework that involves using a chosen reduction scheme, such as BTS, to produce a reduced set of prototypes, from which a smaller kernel matrix is constructed. Ensemble classifier methods are then employed on this matrix to compensate for any loss in accuracy resulting from the reduction in dataset size.

While most work in prototype reduction has focused on supervised learning tasks, the concept has been used implicitly as part of many clustering algorithms. Notably, Cutting *et al.* (1992) proposed a technique, referred to as *fractionation*, to improve the efficiency of agglomerative hierarchical clustering methods. This technique can be viewed as performing a type of prototype extraction. Initially, a corpus is divided into a predefined number of fractions. The documents in each fraction are then clustered separately so that, by treating each cluster as a single "meta-document", the number of data objects is subsequently reduced. The *buckshot* method, also proposed by Cutting *et al.* (1992), employs a stochastic prototype selection scheme and combines elements of classical hierarchical and partitional algorithms to reduce the time required to cluster large sets of documents. It is interesting to note that the application of prototype selection in clustering is closely related to both the problem of outlier removal (Kollios *et al.*, 2003) and the choice of seeds in cluster initialisation (Juan & Vidal, 2000).

7.2.2 Kernel Prediction-Based Validation

To avoid having to work in the original high-dimensional feature space, we make use of the recently proposed kernel clustering methods as described in Section 2.8. The advantage of these methods in the context of stability analysis stems from the fact that, having constructed a single kernel matrix $\mathbf{K}$, multiple clusterings can subsequently be generated without referring back to the original data. Since the k-means algorithm has commonly been used in both stability analysis and document clustering, we make use of the corresponding kernelised version of the algorithm.

To form the basis for our validation scheme, we choose the prediction-based method proposed by Tibshirani *et al.* (2001) due to its empirical success and computational advantage over other stability analysis methods. The latter benefit derives from the fact that each run of the clustering algorithm only considers a sample of $\frac{n}{2}$ objects. Formally, the validation process involves applying two-fold cross-validation to randomly split the dataset $\mathcal{X} = \{x_1, \ldots, x_n\}$ into disjoint training and test sets, denoted $\mathcal{X}_a$ and $\mathcal{X}_b$ respectively. Both sets are subsequently clustered using kernel k-means with random initialisation. A prediction for the assignment of objects in the test set is then produced by assigning each $x_i \in \mathcal{X}_b$ to the nearest centroid in training clustering. The accuracy of this prediction is assessed by measuring the degree to which it agrees with the original clustering of $\mathcal{X}_b$. To measure prediction accuracy, we propose using an adjusted version of the *prediction strength* index (3.34), because of its strong theoretical foundation and superior empirical performance. Rather than manually choosing from among the potential values, we automatically select the value k that leads to the maximum average score over τ runs. However, prediction strength does exhibit a strong bias toward smaller values of k as demonstrated by the

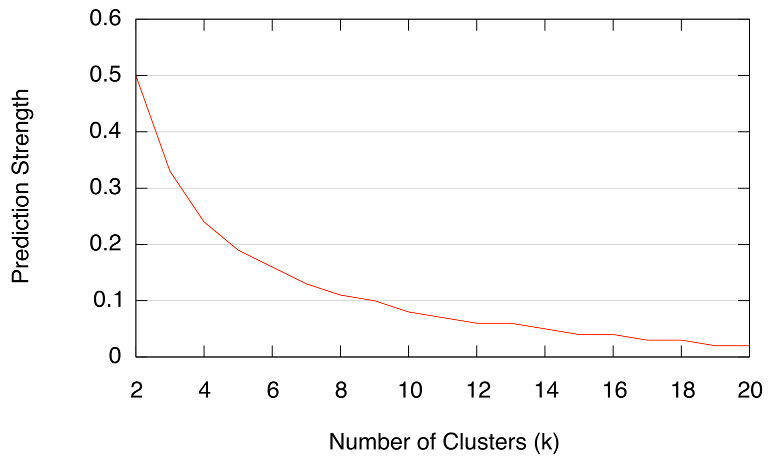


Figure 7.2: Plot of the expected values for the *prediction strength* index when applied to randomly generated pairs of partitions for $k \in [2, 20]$, illustrating the bias of the index with respect to smaller values of k.

```
1. Construct a $n \times n$ kernel matrix $\mathbf{K}$.
2. Randomly generate $\tau$ splits of $\mathcal{X}$ into training and test sets.
3. For each value of $k \in [k_{min}, k_{max}]$ :
    (i) For each split $(\mathcal{X}_a, \mathcal{X}_b)$:
        a. Apply kernel $k$-means to the training set $\mathcal{X}_a$ using kernel $\mathbf{K}$.
        b. Predict the assignment of documents in $\mathcal{X}_b$ based on the centroids from the
           clustering of $\mathcal{X}_a$.
        c. Apply kernel $k$-means to the test set $\mathcal{X}_b$ using kernel $\mathbf{K}$.
        d. Evaluate prediction strength and correct for chance.
    (ii) Compute the mean corrected prediction strength for $k$ over all $\tau$ splits.
4. Estimate $\hat{k}$ by selecting the candidate value $k$ leading to the highest mean corrected
   prediction strength.
```

Figure 7.3: Kernel prediction-based validation scheme.

plot of expected values given in Figure 7.2. We can readily address this by employing the widely-used adjustment technique described in Hubert & Arabie (1985) to correct for chance agreement:

$$S'(\mathcal{C}_b, \mathcal{P}_b) = \frac{S(\mathcal{C}_b, \mathcal{P}_b) - \bar{S}_k(\mathcal{C}_b, \mathcal{P}_b)}{1.0 - \bar{S}_k(\mathcal{C}_b, \mathcal{P}_b)} \tag{7.1}$$

Note that $\bar{S}_k(\mathcal{C}_b, \mathcal{P}_b)$ is the expected prediction strength on the split $(\mathcal{X}_a, \mathcal{X}_b)$, where each partition contains k clusters. This value may be approximated by calculating the mean value of Eqn. 3.34 over a large number of pairs of randomly generated partitions.

As discussed by Lange *et al.* (2004), the choice of classifier used to make predictions should complement the clustering algorithm. To “mimic” the assignment behaviour of the kernel k-means algorithm, we employ a kernel nearest centroid classifier, such that each object in $\mathcal{X}_b$ is classified as being a member of the class represented by the nearest kernel pseudo-centroid in the training clustering. Subsequently, we use corrected prediction strength to evaluate the degree to which the predicted classification agrees with the clustering of $\mathcal{X}_b$ as produced by kernel k-means. The full validation scheme is summarised in Figure 7.3.

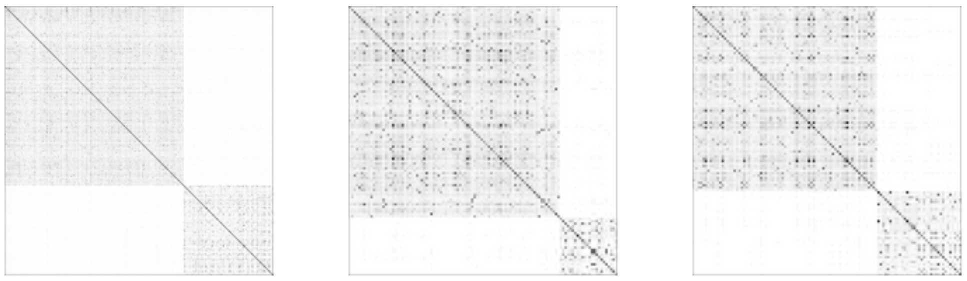
7.2.3 Kernel Prototype Reduction

In the previous section, we described a stability-based validation method suitable for use on high-dimensional data. However, the validation process still requires τ runs consisting of clustering and prediction assessment phases, which both run in $O((\frac{n}{2})^2)$ time. Clearly, decreasing n will make the process significantly less computationally expensive. Motivated by the large-scale clustering technique described by Cutting *et al.* (1992), an intuitive solution is to create a reduced set of $n' < n$ objects, upon which the validation procedure may be applied. However, any such reduction must be performed in a way that preserves the structures in the data.

Meeting these requirements without any form of supervision is not a trivial task. Bezdek & Kuncheva (2001) noted that reduction approaches utilising class information tend to be far more successful than their purely unsupervised counterparts. Many supervised prototype reduction approaches process each class separately. As a result, the reduced prototypes will be "meaningful" in the sense that they will be constructed from data objects belonging to a single class. In the absence of class labels, we must rely upon intrinsic properties of the data to ensure that all structures in the data are adequately represented. Unfortunately, text corpora often contain unbalanced clusters, which may also differ in their relative densities, making the task particularly problematic. To address these issues, we propose a reduction scheme consisting of two phases. In the first phase, prototype extraction is used to generate a set of candidate prototypes formed from small homogeneous regions of the data. The second phase involves selecting a subset of n' prototypes which are used to build a $n' \times n'$ *reduced kernel matrix*, denoted by $\mathbf{K}'$.

Prototype Extraction

Initially, we create a set of extracted prototypes $\mathcal{S} = \{s_1, \ldots, s_n\}$ in a manner similar to that employed by the supervised BTS reduction scheme (Hamamoto *et al.*, 1997), where new prototypes are formed by locally combining subsets of the original dataset $\mathcal{X}$. Formally, we define a *neighbourhood* $\mathcal{N}_i$ as a subset of $\mathcal{X}$ consisting of a seed object x_i together with its set of p nearest neighbours. A new prototype s_i may be constructed from the mean of the $p+1$ objects in a neighbourhood. Since we wish to work in the kernel-induced space only, we consider s_i to be the pseudo-centroid of the subset $\mathcal{N}_i$ as calculated from the values in $\mathbf{K}$. We note that regions forming cluster structures will normally be locally homogeneous, in the sense that the majority of the set of neighbours of each object are likely to belong to the same cluster as that object (Frederix & Pauwels, 2004; Ding & He, 2004). Therefore, prototypes constructed from the centroid of sufficiently small neighbourhoods will generally be representative of a single natural class.



(a) Full kernel (b) Reduced kernel (random sampling) (c) Reduced kernel (density biased selection)

Figure 7.4: Normalised linear kernels for a subset of 300 documents from the 20NG collection, showing the relation between the original kernel matrix and matrices constructed using prototype reduction.

Density-Biased Candidate Selection

The problem remains of selecting a subset $\mathcal{S}'$ of n' optimal prototypes from the n possible candidates. A possible solution is to apply unbiased random sampling to choose $\mathcal{S}'$, where each reduced prototype has an equal probability of being selected. However, this approach has several drawbacks in the context of validation. Ideally, we wish to select a fraction of prototypes from each class that is proportional to the size of that class in the original data. A single random sample from $\mathcal{S}$ is not guaranteed to achieve this. To illustrate the problem, we consider the two-class subset of 450 documents from the 20NG collection which was considered in Section 7.2.1. Figure 7.4(a) shows the full normalised linear kernel matrix constructed on the original data, while Figure 7.4(b) shows a reduced approximation produced by randomly selecting seed objects. It is evident from the latter that the smaller 'hockey' class is not adequately represented by the random reduction process. We observe that reduced prototypes chosen in this way frequently fail to produce a true proxy for the dataset, resulting in poor estimations for $\hat{k}$ in the subsequent validation process. In these cases, the failure is often due to poor sampling of smaller clusters or important sub-regions within clusters. While we could run the process multiple times and aggregate the results, the computational cost would negate the benefits of performing prototype reduction.

As an alternative, the second phase of our reduction procedure employs a density-biased strategy to select $\mathcal{S}'$. This strategy has similar goals to existing density-biased sampling techniques (*e.g.* Palmer & Faloutsos, 2000), but is both deterministic and significantly less computationally demanding. Firstly, we define the *compactness* (*i.e.* density) of a neighbourhood $\mathcal{N}_a$ as the average of the pairwise affinities between its members:

$$C(\mathcal{N}_a) = \frac{\sum_{x_i, x_j \in \mathcal{N}_a} K_{ij}}{|\mathcal{N}_a|^2} \tag{7.2}$$

where $|\mathcal{N}_a| = p + 1$. This is equivalent to the "self-similarity" of the pseudo-centroid formed from $\mathcal{N}_a$. In the selection process, the prototypes in $\mathcal{S}$ are ranked in descending order according to their compactness. From these, we uniformly choose $n' = \frac{n}{\rho}$ prototypes, where ρ is the *reduction rate* that determines the degree to which the number of data objects should be reduced. Specifically, we select every ρ-th prototype from the ordered list, thereby ensuring that we represent all density patterns in the data. We then build the reduced kernel matrix $\mathbf{K}'$ based on these n' prototypes. Rather than computing explicit representations for the new prototypes in the original feature space, we can make use of the values in the original kernel matrix to directly construct $\mathbf{K}'$. Formally, the affinity between a pair of reduced prototypes s_i and s_j is defined as:

$$K'_{ij} = \frac{\sum_{x_a \in S_i, x_b \in S_j} K_{ab}}{(p+1)^2} \tag{7.3}$$

It is possible that a matrix constructed in this way may not always be positive semi-definite. However, as observed in Section 6.3.5, this does not pose a significant problem for the kernel k-means algorithm.

Referring back to our previous example, we see that, unlike the matrix in Figure 7.4(b), the reduced kernel matrix shown in Figure 7.4(c) is clearly representative of the two classes in the original dataset. In practice, we consistently observe that this density-biased selection strategy produces extracted prototypes that accurately summarise the underlying structures in the data. We contend that this is due to the inclusion of regions representing clusters of varying densities and all sub-regions within those clusters.

Algorithm Efficiency

Once we have constructed the reduced kernel matrix, the validation scheme proceeds as described in Section 7.2.2. The proposed reduction strategy results in a significant decrease in the computational cost of the validation process. Our approach does involve a once-off initialisation step, requiring time $O(n \log n)$ for prototype extraction and $O(n'^2 p^2)$ for the construction of $\mathbf{K}'$. However, the computational gains made in the subsequent validation process are substantial. For each of the τ runs, the costs associated with clustering and prediction assessment are both reduced to $O((\frac{n}{2\rho})^2)$.

7.2.4 Application to Document Clustering

While our proposed method may be used in conjunction with any valid kernel function, for document clustering we suggest the use of a normalised linear kernel (2.36). As discussed in Section 6.3.2, the matrix of this kernel will often suffer from diagonal dominance. To address the problem, we make use of kernel k-means with *algorithm adjustment* as proposed in Section 6.3.3. A summary of the complete validation process is provided in Figure 7.5.

When performing prototype reduction, we assume that regions will be locally homogeneous, which should generally be the case when an appropriate kernel function is chosen. To maximise homogeneity, we select a low value for the number of nearest neighbours. For document corpora, we have observed that the use of $p = 5$ consistently leads to the

---

Initialisation Phase

1. Construct a full $n \times n$ kernel matrix $\mathbf{K}$ from the original data.
2. Extract candidate prototypes $\mathcal{S}$, consisting of n neighbourhood pseudo-centroids.
3. Evaluate compactness of candidates in $\mathcal{S}$ and sort accordingly in descending order.
4. Uniformly select a subset $\mathcal{S}'$ of n' reduced prototypes from the ordered list.
5. Construct the $n' \times n'$ reduced kernel matrix $\mathbf{K}'$ based on $\mathcal{S}'$ using Eqn. 7.3.

Validation Phase

1. Randomly generate τ splits of $\mathcal{S}'$ into training and test sets.
2. For each value of $k \in [k_{min}, k_{max}]$:
   (i) For each split $(\mathcal{X}_a, \mathcal{X}_b)$:
      (a) Apply adjusted kernel k-means to the training set $\mathcal{X}_a$ using kernel $\mathbf{K}'$.
      (b) Predict the assignment of documents in $\mathcal{X}_b$ based on centroids from clustering of $\mathcal{X}_a$.
      (c) Apply adjusted kernel k-means to the test set $\mathcal{X}_b$ using kernel $\mathbf{K}'$.
      (d) Evaluate prediction strength and correct for chance.
   (ii) Compute mean corrected prediction strength for k.
3. Estimate $\hat{k}$ by selecting the candidate value k leading to the highest mean corrected prediction strength.

Figure 7.5: Kernel prediction-based validation scheme, with prototype reduction.

construction of meaningful prototypes that represent a single natural class. Empirical observations also indicate that a value of $\rho = 4$ for the reduction rate substantially reduces the time required for the validation process, without significantly affecting its accuracy. The selection of ρ is also related to the maximum number of runs τ. The computational gains resulting from prototype reduction facilitate the use of a larger value (*e.g.* $\tau = 200$) to guarantee the robustness of the overall validation procedure. It must be stressed that, in practice, the use of these general purpose parameter values proved to be effective on a diverse range of datasets, indicating that the proposed validation method is quite robust to the choice of values for these parameters. This allows us to focus on the more immediate task of estimating the number of clusters.

7.2.5 Experimental Evaluation

In this section, we compare the validation scheme proposed in Section 7.2 with techniques operating on the full set of original data objects. The experimental process involved applying four stability analysis schemes to each dataset:

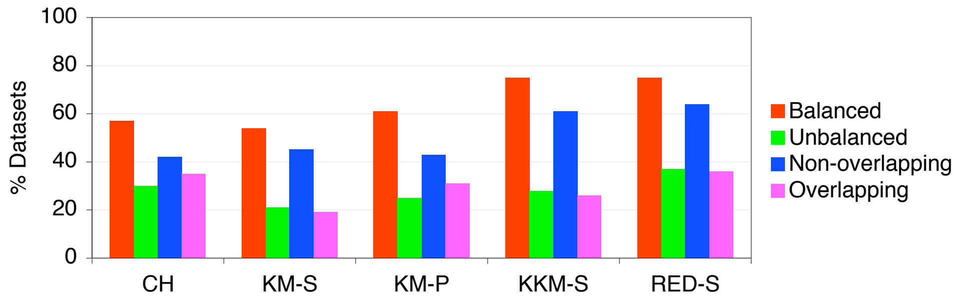
KM-S:	Prediction-based validation using corrected prediction strength (Tibshirani *et al.*, 2001), and k-means with cosine similarity.
KM-P:	Prediction-based validation using partition similarity, and k-means with cosine similarity (Giurcaneanu & Tabus, 2004).
KKM-S:	Kernel prediction-based validation using prediction strength, as proposed in Section 7.2.2.
RED-S:	Kernel prediction-based validation using prediction strength and density-biased prototype reduction, as proposed in Section 7.2.3.

The validation schemes were applied across a reasonable range of values for k and comparing their output with the “true” number of natural classes. For our experiments, we chose the range $[2, 10]$. Note that, in the case of the kernel-based algorithms, we employ the adjustment described in Section 6.3.3 to lessen the effect of diagonal dominance. In all experiments, we set $\tau = 200$ to minimise any variance introduced by subsampling, and use random cluster initialisation. Algorithm running times were evaluated based on experiments performed on an Intel Pentium IV 3.4GHz server with 2GB RAM, running Ubuntu Linux and Sun Java 1.5.

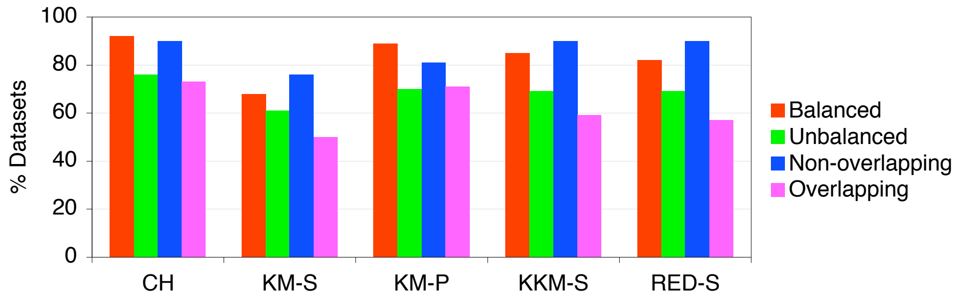
Evaluation on Artificial Data

To illustrate significant differences between the validation strategies, we used the artificial datasets derived from the 20NG collection as described in Section 5.2.2. For the purposes of comparison, we also include results for the Calinski-Harabasz (CH) index, which was shown to be the most suitable index for model selection from among the internal techniques examined in Section 5.4.

Figure 7.6 summarises the relative performance of the validation schemes in terms of the percentage of datasets on which each scheme was successful in identifying $\hat{k}$. When considering the stability analysis schemes, these results indicate that both kernel-based algorithms consistently outperformed those employing the standard k-means algorithm. In these cases, the application of a diagonal dominance reduction frequently lead to significantly higher prediction accuracy. Furthermore, we see that, across the 84 artificial datasets, RED-S generally lead to more instances where the true number of clusters was correctly identified. This is particularly apparent for data with well-separated clusters. The difference was less pronounced on datasets with overlapping clusters, where object neighbourhoods were generally less homogeneous. With regard to the efficiency of these



(a) Correct estimations



(b) Top-3 estimations

Figure 7.6: Summary of the overall percentage of correct and top-3 estimations for $\hat{k}$ as selected by stability-based validation schemes, measured across all artificial text datasets.

Datasets	KM-S	KM-P	KKM-S	RED-S
Balanced	1492	1358	6117	**248**
Unbalanced	1280	1180	4767	**181**
Non-overlapping	1214	1102	4630	**184**
Overlapping	1487	1376	5803	**223**
Overall	1351	1239	5217	**203**

Table 7.1: Summary of the average running times (in seconds) for stability-based validation schemes when applied to all artificial text datasets.

schemes, we observed that the speed-up achieved by working on $\frac{n}{4}$ reduced prototypes was dramatic, as illustrated by the running times listed in Table 7.1.

When comparing the performance of the stability analysis schemes to the results achieved by the internal CH index, we see that RED-S selected the correct value $\hat{k}$ more frequently on all categories of the artificial datasets. However, CH does identify $\hat{k}$ in its top three estimations more often, particularly on those datasets with poorly separated clusters. It should be noted that the results for the CH index are also based on results aggregated over many clusterings, which can be costly for larger values of n. While it is possible to apply the index on a single clustering, the instability of the underlying k-means algorithm could easily result in a misleading estimation for $\hat{k}$.

Evaluation on Real Data

In our second evaluation, we compare the validation schemes on real-world corpora that have previously been used in document clustering. Table 7.2 shows the results of the comparison, indicating the top three estimated values for $\hat{k}$ on the real corpora. In almost all cases, the reduced validation method (RED-S) recommended the same value of k as that chosen when working on the full kernel matrix (KKM-S). Only in the cases of the *reuters5* and *sports* datasets did it fail to rate $\hat{k}$ among its top three choices. However, in Section 5.4.2 we observed that other well-known validation methods also perform poorly on these corpora, which contain significantly overlapping clusters. It is worth nothing that both kernel-based techniques consistently outperformed those employing standard k-means. In these cases, our evaluations indicate that the application of a diagonal dominance reduction strategy leads to a non-trivial improvement in validation accuracy.

When we compare the performance of the stability analysis methods to that of the baseline internal technique, we see that RED-S performed at least as well in estimating

Dataset	$\hat{k}$	CH	KM-S	KM-P	KKM-S	RED-S
bbc	5	**5**,6,4	**5**,4,6	**5**,6,7	**5**,2,6	**5**,6,4
bbcsport	5	6,**5**,7	**5**,4,6	**5**,6,4	**5**,6,4	**5**,4,6
classic	4	3,**4**,5	5,3,**4**	3,5,6	5,**4**,6	**4**,5,3
classic3	3	**3**,4,2	**3**,2,4	**3**,2,4	**3**,4,5	**3**,4,2
cstr	4	3,**4**,5	3,2,**4**	3,**4**,2	3,**4**,5	3,**4**,2
ng17-19	3	5,6,4	5,4,6	5,6,4	4,5,**3**	4,5,**3**
ng3	3	**3**,4,5	**3**,4,2	**3**,4,5	**3**,4,2	**3**,4,2
reuters5	5	2,3,4	2,3,4	2,3,4	2,3,4	2,3,4
reviews	5	2,3,4	2,6,7	2,9,8	2,**5**,4	2,**5**,6
sports	7	6,**7**,5	2,3,6	2,3,6	2,5,4	2,5,6

Table 7.2: Summary of the top-3 estimations for $\hat{k}$ as selected by stability-based validation schemes, when applied to clusterings of real-world text datasets.

the number of clusters as the CH index on all but the *sports* dataset, providing better estimations on four of the other datasets. It is interesting to observe that, in cases where both stability analysis and internal validation fail to identify $\hat{k}$, they still often agree with one another. A prime example of this is the *reuters5* corpus, where each of the five methods under consideration selected $k = 2$ as its first choice. This suggests that it may represent a statistically good value for the number of clusters, although it does not agree with the manual classification for the corpus. In cases where this discrepancy occurs, it is possible that the problem lies with the model used to represent the data or the choice of similarity metric. If the underlying classes in the data are not linearly separable, the use of an appropriate non-linear kernel function may resolve this problem. In this respect, the kernel-based validation methods benefit from the modularity of kernel clustering methods, where an alternative measure of affinity may be readily used without modifying the validation algorithm itself.

We observed that, when using kernel-based validation, the application of prototype reduction with $\rho = 4$ resulted in a 20 fold decrease in the time required for the entire process. For example, selecting a value for k on the *bbc* corpus took 55 minutes when using the full kernel matrix, while the same procedure using RED-S took only 3 minutes. A complete comparison of algorithm running times is provided in Table 7.3, which clearly shows that RED-S was significantly faster than any of the other strategies considered. It is also apparent that the procedures running on the full $n \times n$ kernel matrices took significantly longer than those performed using standard k-means, particularly on the

Dataset	KM-S	KM-P	KKM-S	RED-S
bbc	1503	1400	3199	**139**
bbcsport	434	423	315	**12**
classic	3181	2315	50804	**2107**
classic3	1592	1344	11740	**522**
cstr	144	142	116	**5**
ng17-19	2079	1971	6197	**263**
ng3	1881	1728	6599	**288**
reuters5	648	555	3397	**157**
reviews	4276	3960	11915	**577**
sports	10788	9870	90903	**3137**

Table 7.3: Summary of the average running times (in seconds) for stability-based validation schemes, when applied to real-world text datasets.

larger *sports* and *classic* datasets. This stems from the fact that the implementation of k-means in our toolkit is optimised to take advantage of the sparse nature of text data. The improvement only occurs on datasets where at least 98% of the values in the term-document matrix are zero. However, it is worth nothing that this level of sparsity will generally not be present in other domains, where employing validation on a full kernel matrix may lead to computational savings.

7.2.6 Summary

In this section we proposed a practical approach to stability-based validation suitable for the task of estimating the number of clusters in large, high-dimensional datasets such as text corpora. The use of kernel clustering methods allows us to work on a single kernel matrix rather than repeatedly computing distances in the original feature space. Moreover, we have demonstrated that we can significantly decrease the computational demands of the validation process by employing a form of prototype reduction to construct a reduced kernel matrix. To ensure that the use of a condensed representation does not adversely impact upon the accuracy of the validation process, we have proposed a density-biased strategy for selecting a set of reduced prototypes that adequately represent the underlying classes in the data, regardless of their relative sizes or densities. Notably, the reduction process does not require that we explicitly represent these new prototypes as feature vectors. Extensive experimental evaluations have shown this validation process to be effective on a large number of real and artificial datasets, where it consistently produced good estimates for the optimal number of clusters, often out-performing existing methods that are significantly more computationally expensive. In addition, it also provided estimations for the correct number of clusters k that were generally as good as or better than those achieved by the best internal validation technique.

A potentially interesting avenue of research would be to combine aspects of internal and prediction-based validation to produce an approach for model selection that rewards both high stability and the traditional of clustering objectives of producing separated, compact clusters. Once again, the use of prototype reduction could render such an expensive procedure tractable for larger datasets.

While we have focused on validation in document clustering, we believe that our approach is applicable for a wide variety of other domains and kernel functions, where large datasets would otherwise make stability analysis unfeasible. It is also apparent that the prototype reduction strategy described here has other potential applications. In the next section we develop this idea by applying reduction in the context of the clustering task itself.

7.3 Efficient Ensemble Methods for Document Clustering

Recently, ensemble clustering techniques have been shown to be effective in improving the accuracy and stability of standard clustering algorithms (Strehl & Ghosh, 2002b). Unfortunately, the computational cost of generating and integrating a large number of clusterings can prove problematic when working with large, high-dimensional datasets such as text corpora. In particular, the feasibility of applying popular ensemble clustering algorithms may be greatly limited by the number of documents. The number of unique terms used to represent the documents can also greatly affect the running time of the base clustering algorithm, thereby limiting the application of ensemble clustering techniques when working with high-dimensional data.

While reducing the number of base clusterings appears to be a natural solution to this scalability problem, an ensemble consisting of too few members is likely to result in an unstable solution that is little better than that produced by the base clustering algorithm. An example of this behaviour is shown in Figure 7.7, where ensemble clustering applied to the small *cstr* dataset only achieves reasonable increases in accuracy and stability until at least 40 base clustering have been generated. For larger, more complex datasets, the ideal number of ensemble members may be significantly higher. Clearly an alternative strategy is required for reducing ensemble running time.

In Section 7.2.3, we introduced a novel kernel-based prototype reduction scheme that is effective in producing a smaller representation of a text dataset, while still preserving the underlying class structures. An issue common to both stability analysis and ensemble clustering is the requirement to produce a large, diverse collection of base clusterings. We now expand upon our previous work by showing that the principles underlying the reduction scheme are also relevant in improving the efficiency of other computationally costly learning methods. Specifically, we propose a novel scheme for document clustering that involves aggregating an ensemble of clusterings generated on a reduced kernel matrix.

7.3.1 Kernel-Based Ensemble Clustering

We begin by introducing a correspondence-based ensemble clustering approach that operates on a full kernel matrix, which also addresses a number of the ensemble design issues raised in Section 2.9.

Ensemble Generation

To avoid having to repeatedly recompute similarity values in the original feature space when generating a collection of ensemble members, we represent a full dataset of n objects

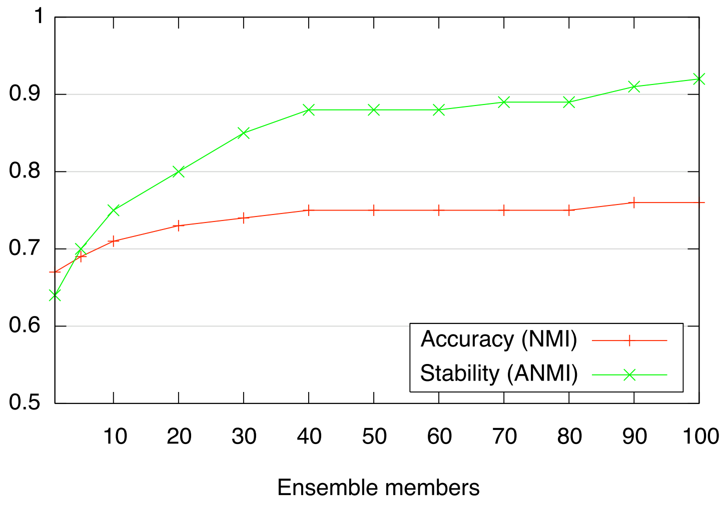


Figure 7.7: Examination of the effect of ensemble size on the accuracy and stability of correspondence-based ensemble clustering, when applied to the *cstr* corpus.

in the form of an $n \times n$ kernel matrix $\mathbf{K}$. As a base clustering algorithm, we use kernel k-means with algorithm adjustment to negate the effects of diagonal dominance. Note that a fixed number of clusters k is generated each time.

To encourage diversity among the ensemble members, unbiased sampling without replacement is applied, with random initialisation used to seed the kernel clustering procedure. Minaei-Bidgoli *et al.* (2004) demonstrated that ensembles created in this way can lead to results that are comparable to bootstrap aggregation, while requiring less computational time to produce the base clusterings, since the size of the individual samples will smaller. We have observed similar behaviour when this generation approach is applied to text data, where we found that a sampling factor of $\beta = 0.8$ generally leads to diverse clusterings that maintain accuracy. After each sample is clustered, membership assignments for the $(1 - \beta)n$ out-of-sample objects are determined by applying a classification scheme. This is analogous to the approach used in prediction-based validation as described in Section 7.2.2, where the classifier predicts the assignments of objects by mimicking the behaviour of the clustering algorithm. In this context, we apply a kernel nearest centroid classifier, where each missing object is assigned to the most similar pseudo-centroid in the base clustering.

Ensemble Integration

Once a collection of base clusterings $\mathbb{C}$ has been generated, we integrate the collection by employing a correspondence clustering scheme similar to the *BagClust1* algorithm proposed by Dudoit & Fridlyand (2003). Unlike other ensemble clustering schemes, the final clustering of the data is constructed incrementally as each ensemble member is generated. Thus, the application of a subsequent clustering procedure to produce a final solution is not required. This scheme also avoids the large storage overhead of maintaining an intermediate representation of $\mathbb{C}$, which is a notable drawback of graph-based integration schemes. We have observed that correspondence-based integration produces more stable results than other schemes such as those based on pairwise co-assignment, which are often highly sensitive to the choice of final clustering algorithm (Greene *et al.*, 2004).

The kernel-based correspondence clustering scheme proceeds as given in Figure 7.8. Having generated the first member $\mathcal{C}_1$, we construct a $n \times k$ membership matrix $\mathbf{V}$:

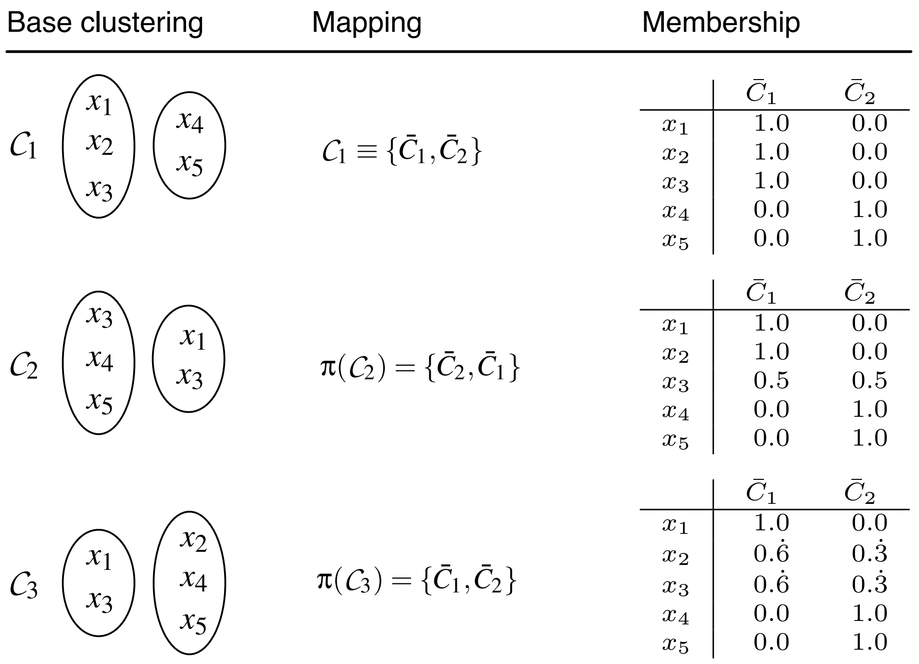
$$V_{ij} = \begin{cases} 1 & \text{if } x_i \in C_j \text{ in } \mathcal{C}_1 \\ 0 & \text{otherwise.} \end{cases}$$

```
1. Construct a full kernel matrix $\mathbf{K}$ and set counter $t = 0$.
2. Increment $t$ and generate a base clustering $\mathcal{C}_t$ as follows:
    (i) Draw a random sample of $\beta n$ objects without replacement.
    (ii) Apply adjusted kernel $k$-means with random initialisation to the sample.
    (iii) Assign each out-of-sample object to the nearest pseudo-centroid in $\mathcal{C}_t$.
3. If $t = 1$, initialise $\mathbf{V}$ as the $n \times k$ binary membership matrix for $\mathcal{C}_1$.
   Otherwise, update $\mathbf{V}$ as follows:
    (i) Compute the current consensus clustering $\bar{\mathcal{C}}$ from $\mathbf{V}$ such that
            $x_i \in \bar{C}_j \;\text{ if }\; j = \arg\max_j V_{ij}$
    (ii) Find the optimal correspondence $\pi(\mathcal{C}_t)$ between the clusters in $\mathcal{C}_t$ and $\bar{\mathcal{C}}$.
    (iii) For each object $x_i$ assigned to the $j$-th cluster in $\pi(\mathcal{C}_t)$, increment $V_{ij}$.
4. Repeat from Step 2 until $\bar{\mathcal{C}}$ is stable or $t = \tau_{max}$.
5. Return the final consensus clustering $\bar{\mathcal{C}}$.
```

Figure 7.8: Kernel-based correspondence clustering algorithm.

As each subsequent clustering $\mathcal{C}_t$ is generated, the values in $\mathbf{V}$ are updated. Unlike when combining voting classifiers, the clusters in each partition will not have a predefined label. Therefore, each new set of clusters must be aligned with those that have been previously generated. The current disjoint consensus clustering, denoted $\bar{\mathcal{C}}$, is computed by taking the majority cluster label for each object, as determined by the maximum value in the associated row in $\mathbf{V}$. The best match between the clusters in $\bar{\mathcal{C}}$ and the existing clusters in $\mathcal{C}_t$ is then identified. Specifically, the optimal permutation $\pi(\mathcal{C}_t)$ may be found in $O(k^3)$ time by solving the minimal weight bipartite matching problem using the Hungarian method (Kuhn, 1955). For each object x_i assigned to the j-th cluster in $\pi(\mathcal{C}_t)$, we increment the entry V_{ij}. When all ensemble members have been added, $\bar{\mathcal{C}}$ represents the final consensus clustering of the data.

To illustrate the correspondence approach to integration, Figure 7.9 provides a simple example where three base clusterings are generated on a set of five data objects for $k = 2$. After each base clustering is generated, the new clusters are mapped to the existing consensus classes $(\bar{C}_1, \bar{C}_2)$ and the 5×2 membership matrix $\mathbf{V}$ is updated. Note that, for the second and third iterations, the maximum row values in $\mathbf{V}$ are used to determine the



$\mathcal{C}_1 \equiv \{\bar{C}_1, \bar{C}_2\}$

	$\bar{C}_1$	$\bar{C}_2$
x_1	1.0	0.0
x_2	1.0	0.0
x_3	1.0	0.0
x_4	0.0	1.0
x_5	0.0	1.0

$\pi(\mathcal{C}_2) = \{\bar{C}_2, \bar{C}_1\}$

	$\bar{C}_1$	$\bar{C}_2$
x_1	1.0	0.0
x_2	1.0	0.0
x_3	0.5	0.5
x_4	0.0	1.0
x_5	0.0	1.0

$\pi(\mathcal{C}_3) = \{\bar{C}_1, \bar{C}_2\}$

	$\bar{C}_1$	$\bar{C}_2$
x_1	1.0	0.0
x_2	$0.\dot{6}$	$0.\dot{3}$
x_3	$0.\dot{6}$	$0.\dot{3}$
x_4	0.0	1.0
x_5	0.0	1.0

Figure 7.9: Example illustrating three iterations of a correspondence-based ensemble clustering procedure, when applied to a set of five data objects.

optimal permutations of the base clusters. For instance, after the third member is added to the ensemble, thresholding $\mathbf{V}$ results in the hard clustering $\bar{\mathcal{C}} = \{\{x_1, x_2, x_3\}, \{x_4, x_5\}\}$.

Ensemble Size

An issue that is often overlooked in ensemble clustering is the choice of a suitable value for the number of ensemble members τ. As discussed previously, if τ is too large, the running time of the ensemble process will be prohibitive. On the other hand, if τ is too small, it is likely that ensemble clustering will result in a solution that is little better than that generated by the base clustering algorithm. A notable benefit of the correspondence approach to ensemble clustering is that, by performing the integration process in parallel with the generation phase, it is possible to readily determine an appropriate point at which the ensemble process may be terminated. We propose to automatically stop generating new ensemble members when the the cluster assignments in $\bar{\mathcal{C}}$ remain unchanged for a fixed number of generations. The process will also be terminated if the number of members t reaches a predefined maximum value τ_{max}.

7.3.2 Efficient Ensemble Clustering

The ensemble clustering approach introduced in Section 7.3.1 allows each base clustering to be generated without referring back to the original feature space. However, for larger datasets, the computational cost of applying τ executions of an algorithm requiring $O((\beta n)^2)$ time may still be prohibitive. Clearly, decreasing the value of n would make the ensemble process significantly less computationally expensive. We now generalise the work described in Section 7.2, showing that the newly proposed kernel-based prototype reduction technique can also be used to improve the efficiency of ensemble clustering.

To demonstrate the extent of the decrease in running time resulting from the use of prototype reduction, we consider the example provided in Figure 7.10. This shows running times for kernel k-means (averaged over 100 runs) when applied to the *classic3* dataset. To investigate the effect of n in this context, we applied the algorithm to samples drawn randomly from the data, for various sampling ratios $\beta \in [0.1, 1.0]$. It is evident that, as the sample size βn increases, the time required for algorithm execution, $O((\beta n)^2)$, will also greatly increase. In contrast, applying kernel k-means to samples drawn from a set of reduced prototypes (in this example, we use $n' = n/4$ prototypes) requires far less processing time, even as $\beta \to 1$. This shows that the time required for aggregation methods that which generate clusterings on many data samples can be dramatically reduced.

Motivated by these observations, we now propose an efficient ensemble clustering ap-

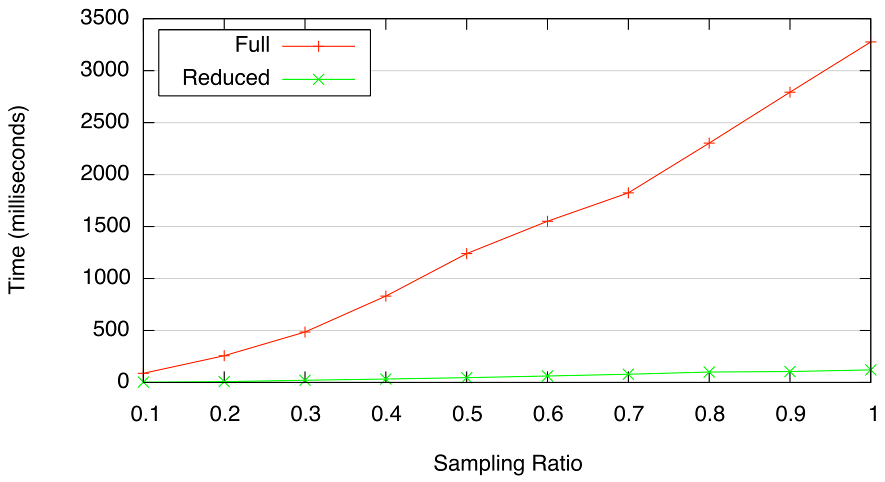


Figure 7.10: Plot of the average running time (in milliseconds) for kernel k-means when applied to samples from the *classic3* corpus, using a full and reduced kernel matrix.

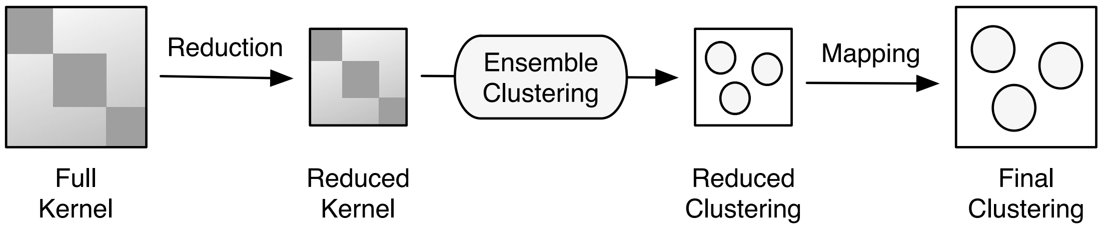


Figure 7.11: Workflow for the ensemble clustering process with prototype reduction.

proach that consists of three steps: apply prototype reduction, perform correspondence clustering on the reduced representation and subsequently map the resulting aggregate solution back to the original data. An outline of this process is illustrated in Figure 7.11.

Ensemble Clustering with Prototype Reduction

The initial reduction process follows that described in Section 7.2.3. Firstly, the original $n \times n$ kernel matrix $\mathbf{K}$ is transformed to a condensed $n' \times n'$ matrix $\mathbf{K}'$, where $n' = \frac{n}{\rho}$ and ρ is a user-defined parameter controlling the reduction rate. Specifically, n extracted prototypes may be potentially constructed by finding the mean of each object together with its set of p nearest neighbours. From these, a subset of $n' < n$ prototypes are chosen using a density-biased selection strategy. The matrix $\mathbf{K}'$ may be directly constructed from the affinity values in $\mathbf{K}$ without referring back to the original feature space. In practice, we use a reduction rate of $\rho = 4$ and consider prototypes constructed from small, homogeneous neighbourhoods (the number of nearest neighbours is set to $p = 5$), as these parameter values worked well in the experimental evaluations described in Section 7.2.5.

Once the reduced kernel matrix has been constructed, ensemble clustering proceeds as given in Figure 7.8. The application of proposed reduction results in a significant decrease in the computational cost of the ensemble process. When generating each ensemble member, the cost of clustering is reduced to $O((\frac{\beta n}{\rho})^2)$. In addition, the time required to construct a cost matrix for the Hungarian matching method and the time needed to update $\mathbf{V}$ are both decreased to $O(n')$.

After the ensemble process has terminated, the problem remains of deriving a final clustering $\hat{\mathcal{C}}$ of the original n data objects from the consensus clustering of reduced prototypes $\bar{\mathcal{C}}'$. Motivated partially by the *buckshot* method (Cutting *et al.*, 1992), we suggest that an intuitive way of achieving this is to assign each original object x_i to the nearest

1. Construct a full $n \times n$ kernel matrix $\mathbf{K}$ from the original data $\mathcal{X}$.
2. Apply prototype reduction to form the $n' \times n'$ reduced kernel matrix $\mathbf{K}'$.
3. Apply kernel-based correspondence clustering using $\mathbf{K}'$ as described in Figure 7.8 to produce a consensus clustering $\bar{\mathcal{C}}'$.
4. Produce a full clustering $\hat{\mathcal{C}}$ of $\mathcal{X}$ by assigning a cluster label to each x_i based on the nearest cluster in $\bar{\mathcal{C}}'$.
5. Apply adjusted kernel k-means using $\hat{\mathcal{C}}$ as an initial partition to produce a refined final clustering of $\mathcal{X}$.

Figure 7.12: Kernel-based correspondence clustering algorithm with prototype reduction.

centroid in $\bar{\mathcal{C}}'$. Just as each reduced prototype can be decomposed into a set of $p+1$ original objects, we can also decompose the centroid of each reduced cluster into the mean of all the original objects which form the reduced prototypes assigned to that cluster. In practice, we can identify the nearest cluster based on values in the original kernel matrix $\mathbf{K}$ and the list of nearest neighbours used to form the reduced prototypes. This mapping of $\bar{\mathcal{C}}'$ to a clustering of the original data objects in $\mathcal{X}$ can be performed in time $O(n'n)$. To further improve the accuracy of this solution, we suggest a refinement procedure that involves applying adjusted kernel k-means to $\hat{\mathcal{C}}$ using the full matrix $\mathbf{K}$. In practice, we observe that this leads to a non-trivial increase in clustering accuracy, while typically requiring very few reassignment iterations before convergence. The complete ensemble process with prototype reduction is summarised in Figure 7.12.

7.3.3 Experimental Evaluation

In order to assess the newly proposed ensemble techniques, we conducted a comparison on the set of real-world text datasets discussed in Section 5.2.1. The primary focus of our evaluation was to consider the effects of applying prototype reduction prior to ensemble clustering, in terms of accuracy, stability and running time. We compare three variations of ensemble clustering:

COR-KM:	Correspondence-based integration of clusterings generated by k-means on the original feature space.
COR-AA:	Correspondence-based integration of clusterings generated by adjusted kernel k-means on a full normalised linear kernel matrix (see Figure 7.8).

COR-RED: Correspondence-based integration of clusterings generated by adjusted kernel k-means on a reduced kernel matrix (see Figure 7.12).

For these experiments, we average the results over 25 trials. In each trial, we automatically terminate the ensemble process after 30 stable iterations have elapsed or when $\tau_{max} = 250$ ensemble members have been generated. As a baseline comparison, we also examine the two base clustering algorithms: k-means with cosine similarity (KM) and adjusted kernel k-means using a normalised linear kernel (AA). In these latter experiments, we performed random initialisation and averaged the results over 200 trials to compensate for the inherent instability of both algorithms. In all cases, we set the number of clusters k to correspond to the number of natural classes in the data.

Comparison of Algorithm Accuracy

Table 7.4 summarises the mean and standard deviation of the NMI scores for the five clustering methods under consideration. On all datasets, the kernel-based ensemble techniques lead to an improvement over both base clustering algorithms. These techniques also performed at least as well as correspondence-based ensemble clustering using standard k-means on the original feature space (COR-KM), and frequently achieved higher accuracy. We observe that the application of a diagonal dominance reduction technique, which limits the influence of self-similarity, contributes to this improvement. In addition, the results in Table 7.4 show that in several cases correspondence clustering after prototype reduction (COR-RED) lead to better results than clustering on the full kernel matrix

Dataset	KM	AA	COR-KM	COR-AA	COR-RED
bbc	0.81 ± 0.08	0.85 ± 0.06	**0.88 ± 0.00**	**0.88 ± 0.00**	**0.88 ± 0.00**
bbcsport	0.73 ± 0.10	0.80 ± 0.08	0.87 ± 0.01	**0.90 ± 0.00**	0.89 ± 0.03
classic	0.70 ± 0.04	0.74 ± 0.02	0.69 ± 0.00	**0.75 ± 0.00**	**0.75 ± 0.00**
classic3	0.93 ± 0.08	0.94 ± 0.06	**0.95 ± 0.00**	**0.95 ± 0.00**	**0.95 ± 0.00**
cstr	0.69 ± 0.05	0.74 ± 0.04	0.76 ± 0.01	0.76 ± 0.01	**0.77 ± 0.03**
ng17-19	0.41 ± 0.12	0.42 ± 0.13	0.47 ± 0.04	0.51 ± 0.05	**0.55 ± 0.04**
ng3	0.83 ± 0.10	0.84 ± 0.10	0.89 ± 0.00	0.90 ± 0.00	**0.91 ± 0.00**
reuters5	0.55 ± 0.07	0.59 ± 0.04	0.60 ± 0.00	**0.61 ± 0.00**	**0.61 ± 0.01**
reviews	0.56 ± 0.08	0.58 ± 0.05	**0.61 ± 0.00**	**0.61 ± 0.00**	**0.61 ± 0.00**
sports	0.62 ± 0.05	0.67 ± 0.06	0.66 ± 0.01	**0.70 ± 0.02**	0.69 ± 0.02

Table 7.4: Summary of NMI accuracy results for base and ensemble clustering methods, when applied to real-world text datasets.

(COR-AA). We suggest that the use of neighbourhood centroids as prototypes allows the production of a robust initial solution that may not be readily obtained by clustering on the full dataset. The application of a subsequent refinement phase allows this solution to be further improved, resulting in a more accurate final clustering. The accuracy of the solutions produced using COR-RED was consistently higher than that achieved by the standard partitional and hierarchical algorithms evaluated in Section 5.3 on the same data. Only the min-max agglomerative algorithm with refinement provided comparable results on certain datasets, although it is significantly more prone to the effects of outliers.

Both of the baseline techniques, KM and AA, exhibited considerable instability due to the sensitivity of these algorithms to the choice of initial clusters, which is reflected in the high standard deviation scores in Table 7.4. In contrast, the ensemble methods tend to be far more robust, frequently producing identical or highly similar partitions. Only in the case of the *bbcsport* and *cstr* datasets did the ensemble methods suffer any noticeable degradation in stability due to prototype reduction. This is likely to be due to the small size of the datasets, and we suggest that a higher number of ensemble members may be appropriate when working with small text corpora. As the time required to generate each member for small datasets is extremely low, this should not pose a significant problem in practice.

Comparison of Algorithm Efficiency

Another important aspect of our evaluation was to assess the computational gains resulting from prototype reduction. Table 7.5 provides a list of the mean running times for the ensemble clustering experiments, which were performed on an Intel Pentium IV 3.4GHz, 2GB RAM running Sun Java 1.5. The cost of the mapping and refinement procedures in COR-RED has the effect that the computational savings are not as dramatic as those observed in the experiments performed in Section 7.2.5. However, the gains afforded by working on a reduced kernel matrix are still very significant. Only in the case of the *classic* dataset did reduction fail to significantly reduce computational cost relative to the other ensemble techniques. Note that the application of the early termination technique for correspondence clustering also has a significant influence on the running times in Table 7.5.

We note that it is possible to further reduce the computational time of ensemble generation by using a smaller factor for subsampling (*e.g.* $\beta = 0.4$). However, as discussed by Minaei-Bidgoli *et al.* (2004), a critical sampling size for a given dataset is required to

Dataset	COR-KM	COR-AA	COR-RED
bbc	95	215	**13**
bbcsport	27	34	**1**
classic	**101**	6181	157
classic3	32	359	**26**
cstr	7	14	**1**
ng17-19	85	753	**40**
ng3	57	485	**19**
reuters5	40	395	**26**
reviews	281	1722	**81**
sports	968	17954	**579**

Table 7.5: Comparison of the mean running times (in seconds) for ensemble clustering methods, when applied to real-world text datasets.

match the accuracy afforded by more expensive bootstrapping strategies. We examined this possibility, but found that, for a number of datasets, using smaller samples lead to a less accurate consensus clustering and higher instability. Consequently, we suggest that using prototype reduction with a relatively high sampling rate (*e.g.* $\beta = 0.8$) represents a pragmatic choice for providing sufficient diversity and ensuring stability on a range of datasets.

7.3.4 Summary

In this section, we have proposed an efficient approach for ensemble clustering based on the use of kernel learning methods and density-biased prototype reduction. This approach was evaluated on real-world text corpora, where the reduced ensemble clustering process was shown to frequently afford a significant decrease in running time, while maintaining high clustering accuracy. In several cases, the proposed method out-performed more computationally costly ensemble techniques operating on the original data.

While we have applied prototype reduction in conjunction with a correspondence clustering scheme modelled on that proposed by Dudoit & Fridlyand (2003), kernel-based prototype reduction may also be employed in conjunction with other integration schemes, such as those based on analysing pairwise co-assignments or a graph representation of an ensemble. The modular nature of our approach naturally lends itself to applications in other learning problems, where alternative domain-specific kernel functions may be used. In addition, we suggest that it may be possible to combine the ensemble methods described in this section with the stability analysis approach proposed in Section 7.2. By analysing the stability of the cluster correspondence model as represented by the membership matrix $\mathbf{V}$, it would be possible to perform clustering and model selection simultaneously. Once again, the application of prototype reduction would make such a procedure feasible when working with large datasets.

Chapter 8

Conclusions

8.1 Introduction

Document clustering has been widely used as a tool for knowledge discovery and data exploration by researchers working with large collections of unstructured text. While a substantial body of literature exists in this area, a number of fundamental issues have remained unresolved. When applying document clustering methods in practice, scalability and performance issues arise due to the number of documents in a collection and the dimensionality of the model used to represent them. The complex nature of cluster structures in this type of data, which often take the form of overlapping groups of unbalanced size, can make the clustering task significantly more difficult. In this thesis, we have investigated a range of practical issues that affect the performance of cluster analysis procedures, and proposed novel solutions that are designed for use on text corpora.

8.1.1 Thesis Summary

Chapter 2 provided a comprehensive review of both classical and contemporary clustering algorithms, which are relevant when working with text data. This included a discussion of the impact of the so-called curse of dimensionality on the performance of many common algorithms, and a study of techniques that may be applied to reduce the number of dimensions required to represent documents. Once a clustering solution has been generated, in many situations it will be useful to apply an automatic procedure to quantitatively assess the validity of the solution. Chapter 3 surveyed a range of different approaches for cluster validation, with a particular emphasis on the problem of choosing an appropriate clustering model for a given dataset.

Chapter 4 introduced the *Text Clustering Toolkit* (TCT), a state-of-the-art framework designed to support the development of applications for unsupervised text mining tasks. In our own research, this toolkit has proved invaluable in allowing us to perform comparative evaluations of existing techniques, and providing a flexible test-bed for the formulation of novel analysis procedures.

Chapter 5 provided a detailed comparison of the performance of classical clustering algorithms and validation methods when applied to text data. We examined the limitations of these procedures, highlighting situations where they perform poorly. We also introduced two new benchmark datasets for document clustering, the *bbc* and *bbcsport* collections, which have proved highly useful in our empirical evaluations.

In Chapter 6, we proposed novel techniques for improving the performance of recently developed clustering approaches when applied to text data. Section 6.2 presented a number of algorithms based on spectral dimension reduction, which produce soft co-clusterings of both documents and terms. Experiments performed on high-dimensional text data show that, in particular, the KSSC and RSSC algorithms can produce clustering solutions which are frequently more accurate than those generated by traditional document clustering methods. Section 6.2.5 described complimentary techniques for generating meaningful cluster labels. Notably, it was demonstrated that well-known criteria from supervised feature selection can be employed to produce highly discriminative labels. In Section 6.3 we shifted our focus to kernel learning methods, where we explored the effects of diagonal dominance, a phenomenon which can impair the performance of centroid-based algorithms when applied to text data. A number of strategies were proposed to address this issue, which lead to higher accuracy and stability in experiments performed on real-world corpora.

Chapter 7 examined the idea of aggregating information obtained from multiple clusterings generated on the same dataset. While this can provide additional insight regarding the underlying structure of a dataset, we observed that the computational cost of common procedures based on this idea can often be prohibitive when working with large document collections. To address this scalability problem, we suggested the application of prototype reduction, thereby rendering aggregation procedures more feasible for use on text data. Specifically, in Section 7.2 we proposed a novel kernel-based prototype reduction strategy, which produces a smaller set of representative objects that accurately summarise the natural structures in a dataset. This strategy was employed as part of an efficient scheme for prediction-based validation, with the effect that running time was substantially reduced, without impairing the scheme's ability to correctly estimate the number of clusters. Subsequently, in Section 7.3 we demonstrated that the scalability of ensemble clustering techniques can also be improved by employing kernel-based prototype reduction, without any loss in clustering accuracy.

8.2 Thesis Findings

The major theoretical and empirical findings of this thesis can be summarised as follows:

- Classical approaches to document clustering, such as the generalised k-means algorithm and agglomerative hierarchical clustering, are prone to producing inaccurate and inconsistent solutions when applied to high-dimensional text data. The accuracy of the latter can be improved by employing a partitional refinement procedure to correct erroneous cluster assignments.

- While spectral clustering methods have previously focused on generating disjoint partitions, these methods can be adapted to produce soft clusterings of both documents and terms. The resulting co-clusterings often achieve significantly higher accuracy than that afforded by existing algorithms based on matrix decomposition.

- Well-known supervised feature selection criteria, such as Information Gain and the χ^2 measure, can be used to identify highly discriminative terms when generating human-interpretable cluster labels.

- The phenomenon of diagonal dominance, which causes overfitting in kernel classifiers, also impacts upon the performance of centroid-based kernel clustering algorithms. This is particularly problematic for kernel matrices representing sparse data, such as text corpora. The application of strategies to reduce these effects can alter algorithm reassignment behaviour, leading to a non-trivial increase in accuracy and stability.

- The efficiency of information aggregation procedures can be substantially improved by applying prototype reduction to minimise the number of objects required to represent the data. By ensuring that the new prototypes provide a true proxy for the original dataset, running times can be greatly reduced without any noticeable loss in algorithm performance. Consequently, procedures such as those based on stability analysis and ensemble clustering can be rendered practical for use on large text corpora.

- Prediction-based validation methods provide a robust means of identifying the correct number of groups in a collection of documents, consistently leading to better estimates than those produced by many well-known validation techniques.

- Correspondence-based ensemble clustering can be used to improve the accuracy and stability of standard clustering algorithms. By employing kernel methods in this context, an ensemble can be constructed in a modular manner that is independent of the original representation of the data.

8.3 Future Work

Following the work presented in this thesis, we now highlight several promising directions for future research.

8.3.1 Toolkit Expansion

With TCT, we have largely focused on providing a comprehensive underlying framework, together with a set of implementations for popular cluster analysis procedures. A natural progression would be to add visualisation capabilities to TCT to further aid researchers in the interpretation of large datasets and clustering solutions generated on those datasets. We believe that the dimension reduction functionality currently supported by the core learning library provides a suitable basis for the development of these capabilities. In addition, to provide non-technical users with access to the functionality of the toolkit, we envisage the development of a web-based or graphical front-end. The generic nature of the architecture of TCT readily facilities the creation of alternative interfaces above the core libraries described in Section 4.2.

8.3.2 Semi-supervised Learning

Recently, an increasing amount of attention has been paid to the problem of applying algorithms in scenarios that do not perfectly correspond to the existing distinction between supervised and unsupervised learning problems (Chapelle *et al.*, 2006). It is apparent that, for many real-world tasks, a limited degree of supervision may be available when performing exploratory data analysis. This may not necessarily correspond to the traditional notion of a subset of labelled training examples. For instance, the supervision could be derived from user feedback regarding the relations between pairs of objects in a small subset of a given dataset. We can represent this information as a set of pairwise constraints, where each constraint indicates that a pair of objects should either always be assigned to the same cluster or should never be assigned to the same cluster. This form of supervision can be used to guide a traditional clustering algorithm, either by providing a good set of initial clusters (Basu *et al.*, 2002), by using a "learnable" similarity function that adapts based on a small amount of label information, or by modifying the objective function of the algorithm to incorporate constraint information (Bilenko, 2003).

Cohn *et al.* (2003) tentatively explored the possibility of employing limited user feedback to guide a traditional clustering algorithm in order to produce a superior partition of a document collection. We believe that significant potential exists to develop these techniques. For example, by presenting a user with pairs of documents representing potential boundary cases, it may be possible to achieve higher accuracy than that afforded by purely unsupervised algorithms. When employing semi-supervised methods, work by Xu *et al.* (2005) has shown that the selection of appropriate constraints can play a pivotal role in determining the level of improvement in accuracy. We believe that clustering aggregation methods, similar to those described in Chapter 7, may be useful in this context. For instance, it may be possible to identify pairs of documents whose co-assignments are inconsistent over many runs of a clustering algorithm. We suggest that the modular nature of the TCT libraries makes it a useful starting point for work in the area of semi-supervised learning.

8.3.3 Applications in Other Domains

While our focus in this thesis has been on the general task of document clustering, we expect that many of the novel techniques proposed here will also be applicable in unsupervised learning problems arising in other domains. In particular, we believe that there is significant scope for applying the aggregation methods described in Chapter 7 to improve the effectiveness and efficiency of knowledge discovery procedures in areas such as gene expression analysis and image processing, where large datasets have previously made the use of similar techniques unfeasible.

Bibliography

Agrawal, R., Gehrke, J., Gunopulos, D. & Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. ACM SIGMOD International Conference on Management of Data*, 94–105.

Aizerman, M., Braverman, E. & Rozonoer, L. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, **25**, 821–837.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 716–723.

Basu, S., Banerjee, A. & Mooney, R.J. (2002). Semi-supervised clustering by seeding. In *Proc. 19th International Conference on Machine Learning (ICML'02)*, 27–34.

Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour*. Princeton University Press.

Ben-Hur, A., Elisseeff, A. & Guyon, I. (2002). A stability based method for discovering structure in clustered data. In *Proc. 7th Pacific Symposium on Biocomputing*, 6–17.

Bezdek, J.C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York.

Bezdek, J.C. & Kuncheva, L. (2001). Nearest prototype classifier designs: An experimental study. *International Journal of Intelligent Systems*, **16**, 1445–1473.

Bezdek, J.C. & Pal, N.R. (1995). Cluster validation with generalized dunn's indices. In *Proc. 2nd New Zealand Two-Stream International Conference on Artificial Neural Networks and Expert Systems*, 190.

Bilenko, M. (2003). Learnable similarity functions and their applications to record linkage and clustering. Tech. rep., Department of Computer Science, University of Texas, Austin.

Bingham, E. & Mannila, H. (2001). Random projection in dimensionality reduction: applications to image and text data. In *Proc. 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 245–250.

Boley, D. (1998). Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, **2**, 325–344.

Bolshakova, N. & Azuaje, F. (2002). Cluster validation techniques for genome expression data. Tech. Rep. 2002-33, Dept. Computer Science, Trinity College Dublin.

Brand, M. & Huang, K. (2003). A unifying theorem for spectral embedding and clustering. In *Proc. 9th International Workshop on AI and Statistics*.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, **24**, 123–140.

Brodley, C. & Lane, T. (1996). Creating and exploiting coverage and diversity. In *Proc. AAAI Workshop on Integrating Multiple Learned Models*, 8–14, Portland, Oregon.

Brunet, J.P., Tamayo, P., Golub, T.R. & Mesirov, J.P. (2004). Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academy of Sciences*, **101**, 4164–4169.

Calinski, T. & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics*, **3**, 1–27.

Cancedda, N., Gaussier, E., Goutte, C. & Renders, J.M. (2003). Word sequence kernels. *Journal of Machine Learning Research*, **3**, 1059–1082.

Chan, P.K., Schlag, M.D.F. & Zien, J.Y. (1994). Spectral K-way ratio-cut partitioning and clustering. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, **13**, 1088–1096.

Chapelle, O., Schölkopf, B. & Zien, A., eds. (2006). *Semi-Supervised Learning*. MIT Press, Cambridge.

Cohn, D., Caruana, R. & McCallum, A. (2003). Semi-supervised clustering with user feedback. Tech. rep., Cornell University.

Comon, P. (1994). Independent component analysis, a new concept? *Signal Processing*, **36**, 287–314.

Creator, R., Kaufman, L., Source, R. & Zentralblatt, M. (1984). K-means-type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**, 81–87.

Cristianini, N. & Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines*. Cambridge University Press.

Cutting, D.R., Pedersen, J.O., Karger, D. & Tukey, J.W. (1992). Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. In *Proc. 15th Annual International Conference on Research and Development in Information Retrieval*, 318–329.

Dalamagas, T., Cheng, T., Winkel, K. & Sellis, T. (2004). Clustering XML Documents by Structure. In *Proc. 3rd Hellenic Conference on Artificial Intelligence*.

Dash, M. & Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis*, **15**.

Dash, M. & Liu, H. (2000). Feature selection for clustering. In *Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'00)*, 110–121, Springer.

Davies, D.L. & Bouldin, W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **1**, 224–227.

Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W. & Harshman, R.A. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, **41**, 391–407.

Dempster, A.P., Laird, N.M. & Rubin, D.B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, **39**, 1–38.

Dhillon, I., Guan, Y. & Kulis, B. (2004a). A unified view of kernel k-means, spectral clustering and graph cuts. Tech. Rep. TR-04-25, UTCS.

Dhillon, I.S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. In *Knowledge Discovery and Data Mining*, 269–274.

Dhillon, I.S. & Modha, D.S. (2001). Concept decompositions for large sparse text data using clustering. *Machine Learning*, **42**, 143–175.

Dhillon, I.S., Guan, Y. & Kogan, J. (2002a). Iterative clustering of high dimensional text data augmented by local search. In *Proc. IEEE International Conference on Data Mining (ICDM'02)*.

Dhillon, I.S., Kogan, J. & Nicholas, M. (2002b). Feature selection and document clustering. *Survey of Text Mining*, 73–100.

Dhillon, I.S., Guan, Y. & Kulis, B. (2004b). Kernel k-means: spectral clustering and normalized cuts. In *Proc. 2004 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 551–556.

Dimitriadou, E., Weingessel, A. & Hornik, K. (2002). A combination scheme for fuzzy clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, **16**, 901–912.

Ding, C. & He, X. (2002). Cluster merging and splitting in hierarchical clustering algorithms. In *Proc. IEEE International Conference on Data Mining (ICDM'02)*, 139.

Ding, C. & He, X. (2004). K-nearest-neighbor consistency in data clustering: incorporating local information into global optimization. In *Proc. ACM Symposium on Applied Computing (SAC'04)*, 584–589.

Ding, C. & He, X. (2005). On the Equivalence of Non-negative Matrix Factorization and Spectral Clustering. In *Proc. SIAM International Conference on Data Mining (SDM'05)*.

Ding, C., He, X., Zha, H., Gu, M. & Simon, H. (2001). Spectral min-max cut for graph partitioning and data clustering. In *Proc. 1st IEEE International Conference on Data Mining*, 107–114.

Donath, W. & Hoffman, A. (1973). Lower bounds for the partitioning of graphs. *IBM Journal of Research and Developement*, **17**, 420–425.

Drucker, H., Wu, D. & Vapnik, V. (1999). Support Vector Machines for Spam Categorization. *IEEE Transactions on Neural Networks*, **10**, 1048–1054.

Dubes, R.C. (1987). How many clusters are best? - an experiment. *Pattern Recognition*, **20**, 645–663.

Dudoit, S. & Fridlyand, J. (2002). A prediction-based resampling method for estimating the number of clusters in a dataset. *Genome Biology*, **3**, 1–21.

Dudoit, S. & Fridlyand, J. (2003). Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, **19**, 1090–1099.

Dunn, J.C. (1974a). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, **3**, 32–57.

Dunn, J.C. (1974b). Well separated clusters and optimal fuzzy-partitions. *Journal of Cybernetics*, **4**, 95–104.

Dy, J.G. & Brodley, C.E. (2000). Feature subset selection and order identification for unsupervised learning. In *Proc. 17th International Conference on Machine Learning (ICML'00)*, 247–254, Morgan Kaufmann, San Francisco, CA.

Fern, X. & Brodley, C. (2004). Solving cluster ensemble problems by bipartite graph partitioning. In *Proc. 21st International Conference on Machine Learning (ICML'04)*.

Fern, X.Z. & Brodley, C.E. (2003). Random projection for high dimensional data clustering: A cluster ensemble approach. In *Proc. 20th International Conference on Machine Learning (ICML'03)*, Washington.

Fiedler, M. (1973). Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, **23**, 298–305.

Fischer, B. & Buhmann, J.M. (2003). Path-based clustering for grouping of smooth curves and texture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**, 513–518.

Forgy, E.W. (1965). Cluster analysis of multivariate data: efficiency vs interpretability of classifications. *Biometrics*, **21**, 768–769.

Fowlkes, E. & Mallow, C. (1983). A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, **78**, 553–569.

Fred, A. (2001). Finding consistent clusters in data partitions. In *Proc. 2nd International Workshop on Multiple Classifier Systems (MCS'01)*, vol. 2096, 309.

Frederix, G. & Pauwels, E.J. (2004). Shape-invariant cluster validity indices. *Lecture Notes in Computer Science*, **3275**, 96–105.

Ghosh, J. (2003). Scalable clustering methods for data mining. In N. Ye, ed., *Handbook of Data Mining*, chap. 10, Lawrence Erlbaum.

Ghosh, J., Strehl, A. & Merugu, S. (2002). A consensus framework for integrating distributed clusterings under limited knowledge sharing. In *Proc. NSF Workshop on Next Generation Data Mining*, 99–108.

Giurcaneanu, C. & Tabus, I. (2004). Cluster structure inference based on clustering stability with applications to microarray data analysis. *EURASIP Journal on Applied Signal Processing*, **1**, 64–80.

Greene, D. & Cunningham, P. (2005). Producing accurate interpretable clusters from high-dimensional data. In A. Jorge, L. Torgo, P. Brazdi, R. Camacho & J. Gama, eds., *Proc. 9th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'05)*, 486–494, Springer.

Greene, D., Tsymbal, A., Bolshakova, N. & Cunningham, P. (2004). Ensemble clustering in medical diagnostics. In *Proc. 17th IEEE Symposium on Computer-Based Medical Systems (CBMS'04)*, 576–581, IEEE Computer Society.

Guha, S., Rastogi, R. & Shim, K. (1998). CURE: an efficient clustering algorithm for large databases. In *Proc. ACM SIGMOD International Conference on Management of Data*, 73–84.

Gusfield, D. (2002). Partition-distance: A problem and class of perfect graphs arising in clustering. *Information Processing Letters*, **82**, 159–164.

Hall, K.M. (1970). An r-dimensional quadratic placement algorithm. *Managment Science*, **17**, 219–229.

Hamamoto, Y., Uchimura, S. & Tomita, S. (1997). A bootstrap technique for nearest neighbor classifier design. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**, 73–79.

Hamerly, G. & Elkan, C. (2004). Learning the k in k-means. In S. Thrun, L. Saul & B. Schölkopf, eds., *Advances in Neural Information Processing Systems 16*, MIT Press, Cambridge, MA.

Hautamäki, V., Cherednichenko, S., Kärkkäinen, I., Kinnunen, T. & Fränti, P. (2005). Improving k-means by outlier removal. In *Proc. 14th Scandinavian Conference on Image Analysis (SCIA'05)*, 978–987.

Ho, T.K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**, 832–844.

Hochbaum, D.S. & Shmoys, D.B. (1985). A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, **10**, 180–184.

Hubert, L. & Levin, J. (1976). A general statistical framework for accessing categorical. *Psychological Bulletin*, **83**, 1072–1082.

Hubert, L.J. & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, **2**, 193–218.

Jaccard, P. (1912). The distribution of flora in the alpine zone. *New Phytologist*, **11**, 37–50.

Jain, A.K. & Dubes, R.C. (1988). *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Jain, A.K. & Fred, A. (2002a). Data clustering using evidence accumulation. In *Proc. 16th International Conference on Pattern Recognition (ICPR'02)*, vol. 4, 276–280.

Jain, A.K. & Fred, A. (2002b). Evidence accumulation clustering based on the k-means algorithm. *Structural, Syntactic, and Statistical Pattern Recognition*, **LNCS 2396**, 442–451.

Jardine, N. & van Rijsbergen, C.J. (1971). The use of hierarchical clustering in information retrieval. *Information Storage and Retrieval*, **7**, 217–240.

John, G.H., Kohavi, R. & Pfleger, K. (1994). Irrelevant features and the subset selection problem. In *Proc. 11th International Conference on Machine Learning (ICML'94)*, 121–129, Morgan Kaufmann, New Brunswick, NJ.

Juan, A. & Vidal, E. (2000). Comparison of four initialization techniques for the k-medians clustering algorithm. In *Proc. Joint IAPR International Workshops on Advances in Pattern Recognition*, 842–852, Springer-Verlag, London, UK.

Karypis, G. & Han, E.H. (2000). Fast supervised dimensionality reduction algorithm with applications to document categorization and retrieval. In *Proc. 9th ACM International Conference on Information and Knowledge Management*, 12–19, ACM Press.

Katsavounidis, I., Jay Kuo, C.C. & Zhang, Z. (1994). A new initialization technique for generalized Lloyd iteration. *IEEE Signal Processing Letters*, **1**, 144–146.

Kernighan, B.W. & Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, **49**, 291–308.

Kim, S.W. & Oommen, B.J. (2005). On using prototype reduction schemes and classifier fusion strategies to optimize kernel-based nonlinear subspace methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**, 455–460.

Kluger, Y., Basri, R., Chang, J. & Gerstein, M. (2003). Spectral Biclustering of Microarray Data: Coclustering Genes and Conditions. *Genome Research*, **13**, 703–716.

Kollios, G., Gunopulos, D., Koudas, N. & Berchtold, S. (2003). Efficient biased sampling for approximate clustering and outlier detection in large datasets. *IEEE Transactions on Knowledge and Data Engineering*, **15**.

Kruengkrai, C., Sornlertlamvanich, V. & Isahara, H. (2004). Document clustering using linear partitioning hyperplanes and reallocation. In *Proc. Asia Information Retrieval Symposium (AIRS'04)*, 36–47.

Kuhn, H.W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quaterly*, **2**, 83–97.

Kummamuru, K., Dhawale, A. & Krishnapuram, R. (2003). Fuzzy co-clustering of documents and keywords. In *Proc. 12th IEEE International Conference on Fuzzy Systems*, vol. 2, 772–777.

Lafferty, J. & Lebanon, G. (2004). Diffusion kernels on statistical manifolds. Tech. Rep. CMU-CS-04-101, Dept. Computer Science, Carnegie Mellon University.

Lange, T., Roth, V., Braun, M.L. & Buhmann, J.M. (2004). Stability-based validation of clustering solutions. *Neural Computation*, **16**, 1299–1323.

Larsen, B. & Aone, C. (1999). Fast and effective text mining using linear-time document clustering. In *Proc. 5th ACM SIGKDD International Conference on Knowledge discovery and Data Mining (KDD'99)*, 16–22, ACM Press, New York, NY, USA.

Lau, H.F. & Levine, M.D. (2002). Finding a small number of regions in an image using low-level features. *Pattern Recognition*, **35**, 2323–2339.

Law, M. & Jain, A.K. (2003). Cluster validity by bootstrapping partitions. Tech. Rep. MSU-CSE-03-5, University of Washington.

Lee, D.D. & Seung, H.S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, **401**, 788–91.

Lee, S.H. & Daniels, K. (2005). Gaussian kernel width selection and fast cluster labeling for support vector clustering. Tech. Rep. 2005-009, Dept. Computer Science, University of Massachusetts Lowell.

Lehoucq, R.B., Sorensen, D.C. & Yang, C. (1997). *ARPACK Users' Guide. Solution of large eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM.

Leisch, F. (1999). Bagged clustering. Working Paper 51, SFB "Adaptive Information Systems and Modeling in Economics and Management Science".

Levine, E. & Domany, E. (2001). Resampling method for unsupervised estimation of cluster validity. *Neural Computation*, **13**, 2573–2593.

Li, S.Z., Hou, X.W., Zhang, H. & Cheng, Q. (2001). Learning spatially localized, parts-based representation. In *Proc. Computer Vision and Pattern Recognition (CVPR'01)*, vol. 1, 207.

Liu, T., Liu, S., Chen, Z. & Ma, W.Y. (2003a). An evaluation on feature selection for text clustering. In T. Fawcett & N. Mishra, eds., *Proc. 20th International Conference on Machine Learning (ICML'03)*, 488–495, AAAI Press.

Liu, W., Zheng, N. & Lu, X. (2003b). Non-negative matrix factorization for visual coding. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03)*, vol. 3, 293–296.

Liu, X., Gong, Y., Xu, W. & Zhu, S. (2002). Document clustering with cluster refinement and model selection capabilities. In *Proc. 25th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 191–198, ACM Press New York, NY, USA.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N. & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research*, **2**, 419–444.

MacQueen, J.B. (1967). Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symposium on Math, Statistics, and Probability*, vol. 1, 281–297, University of California Press.

Meila, M. (2002). Comparing clusterings. Tech. Rep. 418, University of Washington.

Milligan, G. & Cooper, M. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, **50**, 159–179.

Minaei-Bidgoli, B., Topchy, A.P. & Punch, W.F. (2004). A Comparison of Resampling Methods for Clustering Ensembles. In *Proc. International Conference on Artificial Intelligence (IC-AI'04)*, vol. 2, 939–945.

Monti, S., Tamayo, P., Mesirov, J. & Golub, T. (2003). Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Machine Learning Journal*, **52**, 91–118.

Ng, A., Jordan, M. & Weiss, Y. (2001). On Spectral Clustering: Analysis and an Algorithm. *Advances in Neural Information Processing*, **14**, 849–856.

Ng, R. & Han, J. (1994). Efficient and effective clustering methods for spatial data mining. *Proc. 20th International Conference on Very Large Data Bases*, 144–155.

Opitz, D.W. & Shavlik, J.W. (1996). Generating accurate and diverse members of a neural-network ensemble. *Advances in Neural Information Processing Systems*, **8**, 535–541.

Palmer, C.R. & Faloutsos, C. (2000). Density biased sampling: an improved method for data mining and clustering. In *Proc. SIGMOD International Conference on Management of Data*, 82–92.

Park, L., Palaniswami, M. & Ramamohanarao, K. (2004). Fourier domain scoring: A novel document ranking method. *IEEE Transactions on Knowledge and Data Engineering*, **16**, 529–539.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science, Sixth Series*, **2**, 559–572.

Pelleg, D. & Moore, A. (2000). X-means: Extending *k*-means with efficient estimation of the number of clusters. In *Proc. 17th International Conference on Machine Learning (ICML'00)*, 727–734.

Porter, M. (1980). An algorithm for suffix stripping. *Program*, **14**, 130–137.

Pothen, A., Simon, H.D. & Liou, K.P. (1990). Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal of Mathematical Analysis and Applications*, **11**, 430–452.

Punj, G. & Stewart, D. (1983). Cluster analysis in marketing research: Review and suggestions for applications. *Journal of Marketing Research*, **20**, 134–148.

Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

Rand, W.M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, **66**, 846–850.

Raychaudhuri, S., Stuart, J.M. & Altman., R.B. (2000). Principal components analysis to summarize microarray experiments: application to sporulation time series. In *Pacific Symposium on Biocomputing*, 455–466.

Rijsbergen, C. (1975). *Information Retrieval*. Butterworths.

Roth, V., Braun, M., Lange, T. & Buhmann, J. (2002). A resampling approach to cluster validation. In *Proc. 15th Symposium in Computational Statistics*.

Rousseeuw, P. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, **20**, 53–65.

Saigo, H., Vert, J.P., Ueda, N. & Akutsu, T. (2004). Protein homology detection using string alignment kernels. *Bioinformatics*, **20**, 1682–1689.

Salton, G. & Buckley, C. (1987). Term weighting approaches in automatic text retrieval. Tech. Rep. 87-881, Department of Computer Science, Cornell University, Ithaca, NY, USA.

Salton, G. & McGill, M.J. (1983). *Introduction to Modern Retrieval*. McGraw-Hill Book Company.

Salton, G., Wong, A. & Yang, C.S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, **18**, 613–620.

Schölkopf, B. & Smola, A.J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Schölkopf, B., Smola, A. & Müller, K.R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, **10**, 1299–1319.

Schölkopf, B., Weston, J., Eskin, E., Leslie, C. & Noble, W.S. (2002). A kernel approach for learning from almost orthogonal patterns. In *Proc. 13th European Conference on Machine Learning (ECML'02)*, 511–528.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, **6**, 461–464.

Sedding, J. & Kazakov, D. (2004). Wordnet-based text document clustering. In *Proc. 3rd Workshop on Robust Methods in Analysis of Natural Language Data (ROMAND)*, Geneva.

Shi, J. & Malik, J. (1997). Normalized cuts and image segmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR '97)*, 731–737.

Shi, J. & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**, 888–905.

Sirovich, L. & Kirby, M. (1987). Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A*, **4**, 519–524.

Steinbach, M., Karypis, G. & Kumar, V. (2000). A comparison of document clustering techniques. In *Proc. KDD Workshop on Text Mining*.

Strehl, A. (2002). *Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining*. Ph.D. thesis, University of Texas, Austin.

Strehl, A. & Ghosh, J. (2002a). Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, **3**, 583–617.

Strehl, A. & Ghosh, J. (2002b). Cluster ensembles - a knowledge reuse framework for combining partitionings. In *Proc. Conference on Artificial Intelligence (AAAI'02)*, 93–98, AAAI/MIT Press.

Strehl, A., Ghosh, J. & Mooney, R.J. (2000). Impact of similarity measures on web-page clustering. In *Proc. AAAI Workshop on AI for Web Search*, 58–64, AAAI/MIT Press.

Surdeanu, M., Turmo, J. & Ageno, A. (2005). A hybrid unsupervised approach for document clustering. In *Proc. 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, 685–690, ACM Press, New York, NY, USA.

Tang, B., Shepherd, M., Milios, E. & Heywood, M. (2005). Comparing and combining dimension reduction techniques for efficient text clustering. In *SIAM International Workshop on Feature Selection for Data Mining - Interfacing Machine Learning and Statistics*, Newport Beach, California.

Tao, Q., Scott, S., Vinodchandran, N.V., Osugi, T.T. & Mueller, B. (2004). An extended kernel for generalized multiple-instance learning. In *Proc. 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'04)*, 272–277.

Tibshirani, R., Walther, G. & Hastie, T. (2000). Estimating the number of clusters in a dataset via the gap statistic. Tech. Rep. 208, Dept. Statistics, Stanford University.

Tibshirani, R., Walther, G., Botstein, D. & Brown, P. (2001). Cluster validation by prediction strength. Tech. rep., Dept. Statistics, Stanford University.

Topchy, A., Jain, A. & Punch, W. (2005). Clustering ensembles: Models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**, 1866–1881.

Torre, F.D.L. & Black, M.J. (2003). A framework for robust subspace learning. *International Journal of Computer Vision*, **54**, 117–142.

Turnbull, D. & Elkan, C. (2005). Fast Recognition of Musical Genres Using RBF Networks. *IEEE Transactions on Knowledge and Data Engineering*, **17**, 580–584.

Verma, D. & Meila, M. (2003). A comparison of spectral clustering algorithms. Tech. rep., University of Washington.

Voorhees, E.M. (1986). *The effectiveness and efficiency of agglomerative hierarchic clustering in document retrieval*. Ph.D. thesis, Cornell University.

Wu, G., Chang, E. & Zhang, Z. (2005). An analysis of transformation on non-positive semidefinite similarity matrix for kernel machines. Tech. rep., UCSB.

Xu, Q., des Jardins, M. & Wagstaff, K.L. (2005). Active constrained clustering by examining spectral eigenvectors. In *Proc. 8th International Conference on Discovery Science (DS'05)*.

Yang, Y. & Pedersen, J.O. (1997). A comparative study on feature selection in text categorization. In D.H. Fisher, ed., *Proc. 14th International Conference on Machine Learning (ICML'97)*, 412–420, Morgan Kaufmann Publishers, San Francisco, US, Nashville, US.

Yao, Z. & Choi, B. (2005). Automatically discovering the number of clusters in web page datasets. In *Proc. 2005 International Conference on Data Mining (DMIN'05)*, 3–9.

Yu, S.X. & Shi, J. (2003). Multiclass spectral clustering. In *Proc. 9th IEEE International Conference on Computer Vision*, 313.

Zhang, T., Ramakrishnan, R. & Livny, M. (1996). BIRCH: an efficient data clustering method for very large databases. *Proc. 1996 ACM SIGMOD International Conference on Management of Data*, 103–114.

Zhao, Y. & Karypis, G. (2001). Criterion functions for document clustering: Experiments and analysis. Tech. Rep. 01-040, University of Minnesota.

Zhao, Y. & Karypis, G. (2002). Evaluation of hierarchical clustering algorithms for document datasets. In *Proc. 11th ACM Conference on Information and Knowledge Management*, 515–524.

Zhao, Y. & Karypis, G. (2004). Soft clustering criterion functions for partitional document clustering: a summary of results. In *Proc. 13th ACM Conference on Information and Knowledge Management*, 246–247.

Zhong, H., Shi, J. & Visontai, M. (2004). Detecting unusual activity in video. In *Proc. Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)*, vol. 2, Washington, D.C., USA.

Zhong, S. (2005). Efficient online spherical k-means clustering. In *Proc. 2005 IEEE International Joint Conference on Neural Networks*, vol. 5, 3180–3185.