Hidden Markov Model Clustering of Acoustic Data

Matthew Butler

Master of Science

School of Informatics
University of Edinburgh

2003

Abstract

This dissertation explores methods for cluster analysis of acoustic data. Techniques developed are applied primarily to whale song, but the task is treated in as general a manner as possible. Three algorithms are presented, all built around hidden Markov models, respectively implementing partitional, agglomerative, and divisive clustering. Topology optimization through Bayesian model selection is explored, addressing the issues of the number of clusters present and the model complexity required to model each cluster, but available methods are found to be unreliable for complex data. A number of feature extraction procedures are examined, and their relative merits compared for various types of data. Overall, hierarchical HMM clustering is found to be an effective tool for unsupervised learning of sound patterns.

Acknowledgements

This dissertation would never have been thought of, let alone completed, without the insightful guidance of Chris Williams.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Matthew Butler*)

Table of Contents

Chapter 1

Introduction

Recognition of patterns in sound is a difficult machine learning problem with a wide range of potential applications. Extensive study in the context of human speech has produced systems capable, under certain conditions, of identifying spoken words with a high degree of accuracy. These efforts have been helped by the fact that the linguistic domain is an intimately familiar one. Researchers typically know *exactly* what output a given speech recording ought to elicit, allowing them to acquire high quality labelled training data and to fine tune every aspect of their systems to boost performance.

Not all acoustic domains are as well understood. For some interesting forms of sound data, information about the generating processes is difficult or impossible to obtain. In such cases, it may be possible to learn about these processes based on sound alone. We seek a technique able to discover underlying structure in an acoustic dataset, without recourse to any external information. One form of data-driven inference is clustering: given an unlabelled set of recorded samples, we attempt to sort them into groups in such a way that samples sharing similar features, and thus (perhaps) generated by a common process, are placed together.

One interesting source of sound data for unsupervised analysis is the vocal behaviour of certain whale species. Underwater recordings of these animals show a high degree of sophistication and structure, and the current understanding of the nature and significance of these sounds is far from complete. It seems clear that the songs are planned

and performed with the intent of being intelligible, in at least some rudimentary sense, to other whales. Much could be learned about how the animals live and interact if the intended patterns underlying the songs could be accurately extracted and compared. This analysis can be carried out by human experts, but is time consuming and inherently biased by the fact that human sound perception is specialised for the features and frequencies that are salient in speech, which may or may not be the ones most relevant to whales. Stastical machine learning offers the potential of a more objective analytical technique. We seek a method to group sounds and discover patterns based on the quantitative structure of the acoustic samples, rather than on how they sound to us.

Of course, whale perception is specialised in its own right. Since the cetacean brain is a radically nonlinear system, there can be no guarantee that all of the distinctions deemed significant by this or that statistical method are perceived as important by the whales, nor that all of the distinctions judged *in*significant are ignored by them. All that can be asked of cluster analysis is that it discover the most *intrinsically* strong similarities in the data.

It turns out that even such computational methods, if they are to be sufficiently powerful to find interesting groups, cannot be completely neutral, must emphasize certain sound characteristics over others. Thus there can be no single, universal solution to all acoustic clustering problems. Instead we seek a flexible set of methods that can be adapted to a particular task based on the best assumptions that can be made about the nature of the data and the latent classes within it, however minimal these assumptions might be. The present work applies a number of techniques to the problem of clustering whale song. However, it is treated as far as possible as an instance of the generic acoustic clustering task. In order to verify that the techniques developed have some degree of domain generality, experimental work is also performed on a very different set of data, a simple human speech corpus.

Chapter 2 reviews relevant past work. This starts with research specific to the automated analysis of whale song. It also includes general computational techniques

useful to the acoustic clustering task, including data clustering methods and the hidden Markov model mixture, which is the basic framework for all clustering techniques developed here.

Chapter 3 presents three HMM-based training methods for finding clusters in time series data, and compares their performance under varying synthetic data conditions.

Chapter 4 explores model selection procedures for automatically optimizing the structural parameters of the modelling architecture: the number of clusters that are identified, and complexity of the model used to define each cluster. These are shown to be effective when the intrinsic cluster structure of the data is unambiguous, but prone to failure on complex real-world datasets.

Chapter 5 discusses several forms of preprocessing that are used to convert raw sound data into a form conducive to clustering. Experiments illustrate some fundamental differences between whale song and human speech, demanding that feature extraction procedures be chosen specific to each.

Chapter 6 offers some final results showing the overall effectiveness of the three clustering systems on whale song, as well as conclusions about acoustic clustering in general, and a number of directions in which future research can progress.

Chapter 2

Background

This chapter presents the task at hand, both as a general machine learning problem and as a specific instance of that problem. It also reviews relevant past research, and describes basic techniques that will be employed in subsequent chapters. Section 2.1.1 introduces the domain of primary interest here, cluster analysis of humpback whale song units, and briefly summarizes past work in this data at this university. Section 2.1.2 discusses another dataset that is used to develop and test techniques, a simple human speech corpus, and 2.1.3 presents the method used to generate synthetic data, which is also used extensively for testing purposes. Section 2.2.1 deals with clustering items and with algorithmic approaches to this task; 2.2.2 with how such algorithms can be extended to complex data by means of statistical models; and 2.2.3 with some particular considerations when the data in question is acoustic. 2.3.1 describes the hidden Markov model, which is the fundamental component of the clustering systems developed here. Finally, 2.3.2 extends this basic model to the hidden Markov model mixture, and mentions a number of ways in which it has been applied to clustering problems involving time series data.

2.1 Datasets

This dissertation is a continuation of a line of research that has been ongoing at Edinburgh for some years: the automated analysis of humpback whale song. Much of the

experimental work that follows was developed and tested in this bioacoustic domain. To avoid over-specificity of methods, and to by way of a touchstone from a domain that has been more thoroughly studied, substantial work is also carried out on human speech data. In order to test techniques in a controlled way on data with known and adjustable properties, randomly generated sequences are also used, produced using a sampling framework that mirrors the model architecture used for clustering.

2.1.1 Humpback Whale Song

The humpback whale (*megaptera novaeangliae*) displays the richest vocalisation behaviour of any marine mammal. The songs are highly structured, lasting many minutes or even hours and composed of hierarchically: units lasting on the order of one second are combined in recurring phrases, which in turn are assembled into protracted, identifiable themes. Most singing is done by males in warm breeding waters, and the most accepted view among biologists is that it serves primarily as a mating display [14], rather than as a form of more meaningful linguistic utterance. The song sung by an individual tends to be quite stable, evolving but still recognizable from season to season. There is also a high degree of similarity among the songs within a social group, and when two groups come into contact, they are influenced by each other's songs.

The complexity and stability of this vocal behaviour makes it interesting to marine biologists as a means of tracking population movements and studying social organization and mating behaviour. Human analysis of this information is tedious and time consuming, which limits the breadth of whale song research. It would be useful to this community to have automated techniqes that can find structure in recorded whale song and can discover recurring and similar elements at various levels of granularity. The current research is concerned with the unit level: we wish to extract and classify the fundamental elements of whale song, discrete segments typically 1-2 seconds long roughly analogous to 'words' or musical 'notes'.

This dissertation builds on past work at the University of Edinburgh, most recently by [46]. A system of unit extraction and preprocessing, was devised by [34] and refined

by [45]. From a 22 minute audio stream (sampled at 20 kHz), units were indentified as contiguous segments of sound above a volume threshold, capped at a maximum duration. A Fourier transform was then performed on these linear fragments to produce a spectrogram representation, with 50 frequency bins and timeslices of 26 ms.

This procedure has produced a corpus of 527 units; the current task is to categorise them. The difficulty is that, unlike human speech data, there is no ground truth for whale song units (we can't ask the whales to label a training set for us), which makes this a clustering problem. Each unit has been given a label based on subjective categorisation of its audible properties, but no authority is ascribed to these labels, and they are not used in training. They can provide some indication as to the performance of a clustering technique: if an automatic system can discover the same distinctions that *we* hear as important, it must be doing at least something right. The converse may not hold: a machine-generated clustering that does not match these labels may nonetheless reflect accurately some intrinsic structure in the data. However, as the labels are the only external information that is available by which to assess clustering output, they will be treated in many cases as if they constituted ground truth.

Welsh [46] performed some preliminary clustering on these units, but performance was limited by the use of static representations, converting the spectrograms into vectors rather than leaving them as time series. Because hidden Markov models can represent the temporal characteristics of the data, they are a promising tool in carrying this research forward.

2.1.2 Human Speech

Speech data offers the advantage that the correct clustering, or even several valid ones, are unambiguously known. This allows performance of a clustering system to be quantified and fine-tuned. The corpus used here consisting of several hundred recordings of

the words "yes" and "no" by different speakers and under varying noise conditions[1]. Although the presence of only two word classes makes this a simple dataset, it is by no means a trivial clustering task; successful separation of the words requires other distinctions, such as those between speakers and genders, to be ignored. Of course, larger word vocabularies will place correspondingly greater demands on a clustering system; scaling the methods developed here to the many-class case is left for future work.

2.1.3 Synthetic Data

In order to assess and compare algorithms and techniques in a controlled way, most verification testing was performed on synthetic datasets. The generating process is a mixture of hidden Markov models, which is the same architecture employed in learning clusters, detailed in section 2.3. Toy data has the advantage that its full structure is known: the number of clusters, the number of Markov states generating each cluster, and the cluster identites of each generated item. It is also possible to compare the learned models to the generating parameters for the cluster being fit. Sequences are generated in the following way: K HMMs are created, each with Q states, with $Q = 5$. The dimensionality was set at 10, as this was typical of the feature representations used for the real data. For each state, $\boldsymbol{\mu}$ is drawn uniformly from the unit hypercube, and Σ is spherical $\sigma^2 I$. By adjusting either σ^2 or K, the degree of overlap among clusters can be controlled, allowing for synthetic clustering tasks of arbitrary difficulty. The transition matrix A is set with high self-transition probability and others uniform among the states, and the initial distribution $\boldsymbol{\pi}$ is uniform. Once all HMM parameters are set, sampling from the mixture is simply a matter of (uniform randomly) choosing a generating cluster, producing a state sequence of a fixed length of 40 steps by drawing from $\boldsymbol{\pi}$ and then repeatedly from A conditioned on the current state, and generating Gaussian output from each state. A record is kept of which component generated each sequence, and these labels become the ground truth by which clustering output is evaluated.

[1]The dataset used is a sample subset of a larger corpus from the Center for Spoken Language Understanding, and is available at http://www.cslu.ogi.edu/corpora/downloads/yesno_sample.zip.

2.2 Clustering Acoustic Data

In addition to its specific focus on whale song units and spoken words, this dissertation treats acoustic clustering as a general problem class. It is hoped that the techniques explored will be applicable to any set of acoustic data, or at least to a broad range of such data. This acoustic task is, in turn a special case of statistical clustering.

2.2.1 Clustering

Clustering is the task of learning, without supervision, the most natural or appropriate way to divide a set of items into groups. We want to assign group labels to a dataset in a way that maximises the similarity among items in the same group and minimises similarity between items in different groups. It is often convenient to think of similarity and difference in spatial terms as proximity and distance, and the task as one of partitioning the data space into regions. When the data items are points in Euclidian space, distance calculations are unique and well-defined. However, with more complex data it is not always clear how to define and measure the distance between two data items, and still less that between two clusters of items. In such cases the choice of a distance metric can be a key element of the clustering task.

Given a distance metric, it is possible to evaluate the relative quality of clusterings, which allows us to search for a good one. Since exhaustive consideration of all possible groupings is not scalable to interesting problems, iterative methods must be used to find an optimal grouping. This must be accomplished with no external information of a correctly grouped training set, nor even (in the general case) knowledge of how many clusters the procedure should find. Cluster membership can be absolute, with each item belonging to one and only one cluster at a given time, or it can be represented probabilistically, assigning each item a degree of membership in each cluster.

Broadly, clustering techniques are either partitional or hierarchical [23]. Partitional methods resolve the dataset into groups all at once, shuffling items among clusters until an optimal fit has been achieved. The target number K of clusters must be chosen

in advance. This static topology is a limitation of partitional methods, but also allows them to be relatively simple and fast, as exemplified by the well-known K-means algorithm [23]. Here each cluster is initially centered on a single randomly selected point in the dataset. Cluster membership is determined by assigning each datapoint to its nearest cluster centre, according to the distance metric. The cluster centres are then moved to the means of their associated data points, and the procedure is repeated until some convergence criterion is met, indicating that the procedure has reached a local optimum.

Because partitional methods like K-means require K to be fixed, the only way to assess the number of clusters required is to run the whole algorithm multiple times and compare the results. Even if this is done, there is a lack of general and principled tests by which to judge the quality of clusterings at different levels of granularity [15], making the ultimate choice of K somewhat subjective.

Hierarchical clustering resolves clusters progressively rather than all at once, exploring all possible values of K. Agglomerative hierarchical clustering [28] starts by treating each data item as a cluster unto itself (a singleton cluster). These small, numerous clusters are iteratively merged, nearest pair first, to form ever larger clusters; this continues until some stopping criterion is reached, or for a complete analysis, until all data is included in a single cluster. The behaviour of agglomerative clustering is heavily influenced by the way distances between clusters are calculated. Common measures include single-link, the point-to-point distance between the *closest* pair in the respective clusters; complete-link, the distance between the *farthest* pair; the *mean* of all point distances between the clusters; and Ward's method, a sum-of-squares measure that favours compact clusters of roughly balanced size [26].

Divisive hierarchical clustering [28] performs the same operation in reverse: a single, all-encompassing initial cluster is split into two maximally dissimilar subclusters, and this splitting is repeated until a stopping condition is met or until all clusters are singletons. Typically each split is designed to bisect the cluster; the bisection can be

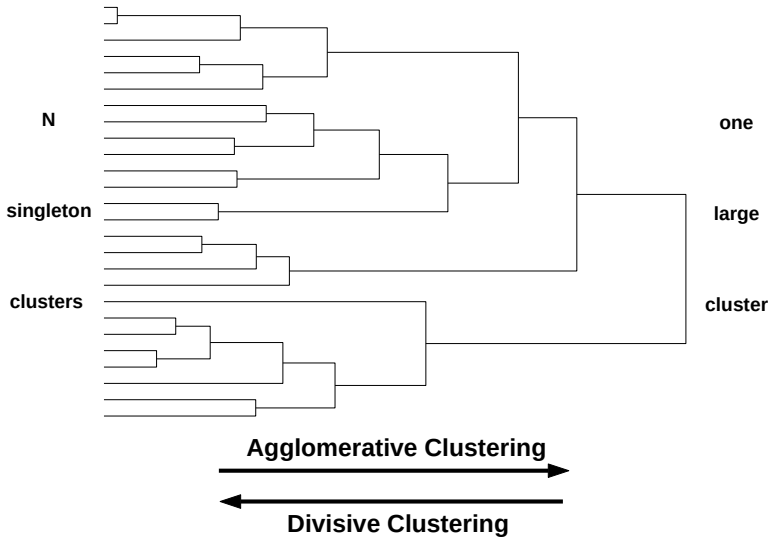


Figure 2.1: The relationship between divisive and agglomerative hierarchical clustering algorithms.

deterministic, separating the most distant pair of points within the cluster or cutting across the direction of maximum variance, or it can be determined stochastically, dividing the cluster orthogonally to a randomly selecting a point and its reflection across the centroid [41].

These methods are called hierarchical because they produce a nested series of clusterings from which subcluster/supercluster relationships can be inferred. This information can be valuable for data exploration. Although these methods do not by themselves the problem of determining the optimal number of clusters, they are conducive to comparing the output given by various values of K, both subjectively and by means of some objective metric of cluster quality.

2.2.2 Model-Based Clustering

When dealing with vector data, it is most natural to treat observations as points in space, and to cluster them based on Euclidian distance. This lends itself to visual analogy, and in many cases is adequate to find a good clustering. However, we can-

not assume in general that statistically important clusters will be of similar width or density: the best-fit cluster to a given point is not necessarily the spatially closest. Furthermore, certain forms of data (such as sound) are not well suited to vector representations. It is therefore necessary to define clusters, not merely as point centres in space, but as distributions. The distance measure of a point from a cluster becomes the posterior probability of its having been generated by the associated distribution. In this likelihood space we can attain much more robust clustering than in a simple Euclidian one. This comes at a price, as creating and training models of the cluster distributions can involve significant computational effort.

Within the Bayesian framework clustering has the same basic structure as pattern classification: given a dataset X, we construct one or more statistical models with parameters $\theta_1, \theta_2, ..., \theta_K$ able to generate items like those in X. We can then compute the probability $p(k|x_i)$ that item x_i belongs in class k by computing the posterior probability

$$\frac{p(x_i|\theta_k)p(k)}{\sum_{j=1}^{K} p(x_i|\theta_j)p(j)}.$$

In the simplest case, where prior class probabilities $p(k)$ are assumed uniform, the task of assigning a class label l_i to an item reduces to $l_i = argmax_k(p(x_i|\theta_k))$.

What makes model-based clustering different from (and more difficult than) classification is that it is unsupervised. In classification we have access to a labelled training set X_{train} for which the 'true' class memberships c_i are given a priori. Each class-specific model θ_k is trained only on the subset of X_{train} identified as belonging to class k. In clustering, no labels are available. Groupings of the training data must be inferred from intrinsic similarities among the input patterns themselves, using the sorts of clustering methods mentioned above.

There is an important connection between model-based clustering and mixture models. A mixture training algorithm performs an implicit clustering task: data items are distributed among components, and each item influences only the component(s) with

which it is associated. Model-based clustering take this grouping function and makes it the explicit goal of the system, rather than a means to the end of accurate modelling.

For both mixture training and clustering, items must be allocated to components in order for training to take place. However, since these allocations are themselves based on responsibility scores computed from trained models, the problem is circular. Training and reallocation therefore cannot be separate, sequential activities, but must be performed concurrently, incorporated into a single iterative algorithm. Such an algorithm will have two phases: one that finds parameter settings to model a given subset of X_{train}, and one that reallocates these subsets.

These phases are embodied by the expectation maximisation (EM) algorithm [12], a general procedure that uses gradient descent to find parameters θ to (locally) maximise the data likelihood $p(X|\theta)$. The **E**xpectation step computes the responsibilities

$$p(k|x_i,\theta) = \frac{p(k)p(x_i|\theta_k)}{\sum_{j=1}^{K} p(j)p(x_i|j)}$$

of each component k for item x_i, and the **M**aximisation step adjusts the parameters θ to increase the expected complete-data likelihood $\prod_i^N \sum_k^K p(k|x_i,\theta)p(x_i|\theta_k)$ given these responsibility assignments. In a mixture model, EM produces a soft partitional clustering: the responsibility determines the degree to which item x belongs to component k, and hence how strongly x will influence the training of θ_k

Soft clustering with the EM algorithm is computationally demanding, particularly in contex of HMMs. Though it has been applied successfully [47] to discrete univariate data, its slow rate of convergence makes it infeasible in many real-world problems. In such cases a hard clustering, though sub-optimal with regard to model fit, is a necessary simplification. There are several ways to 'harden' general EM by inserting a classification step after the Expection is calculated, but before the model parameters are updated. In Classification EM (CEM) [10], each x_i is deterministically assigned to

its most likely generating component:

$$l_i = argmax_k\left(\frac{p(k)p(x_i|\theta_k)}{\sum_{j=1}^{K} p(j)p(x_i|\theta_j)}\right).$$

A more sophisticated version, Stochastic EM [9], uses the same calculation, but instead of maximising the posterior probability it draws a label randomly from the distribution

$$p(l_i = k) = \frac{p(k)p(x_i|\theta_k)}{\sum_{j=1}^{K} p(j)p(x_i|\theta_j)}.$$

This method offers fast initial training, and though it does not converge absolutely, it generates a Markov chain whose stationary distribution is an unbiased estimator of the true maximum likelihood settings found by EM.

The Classification EM procedure bears an obvious relationship to *K*-means clustering. With the added simplifying assumption of uniform prior probabilities for all clusters, CEM becomes analogous to *K*-means, except that instead of simple geometric means, its cluster prototypes are statistical models. Some authors [48] call this model-based *K*-means (*MK*-means), but this can be misleading in the general (non-Gaussian) case, as a model need not be the 'mean' of its training points in any literal sense. Here the term *K-models* will be used instead.

The likelihood fit produced by *K*-means is inferior to that found by EM for a Gaussian mixture, particularly in datasets with overlapping classes. However, covergence is much more rapid [8]. It is reasonable to suppose that this speed versus accuracy tradeoff holds in general between *K*-models and EM, regardless of the model class in question.

2.2.3 Acoustic Clustering

The clustering task considered here is that of acoustic patterns. We take as input a set of (relatively brief) samples of recorded sound, and seek to find a grouping that places together the sounds which are most 'similar'. While it is expedient for many

experimental purposes to provide a clustering system with a predetermined number of clusters to discover, a major goal of acoustic clustering is the determination of the number of clusters that exist naturally in the data.

Of course, there are very many ways in which two sounds can resemble one another, and while we hope to be able to discover the most interesting groups, the best we can expect from an automatic clustering technique is to discover the distinctions that are most prominent in some statistical sense. Preprocessing thus has a vital role to play in any acoustic clusterer: if the system is to find clusters based on acoustic properties that we as humans find most interesting, these properties must be selectively emphasised so as to be the most prominent features of the data. The representation and transformation of acoustic features is a key issue in speech recognition, and a large number of signal processing techniques have been developed for this task. Chapter 5 will present some techniques found useful for clustering the acoustic datasets under consideration.

2.2.4 Measuring Clustering Performance

Assessing the quality of cluster output is not as straightforward as doing so for classification. For data whose ground truth class identities are truly unknown (such as the whale song) assessing how well they have been clustered is extremely problematic, and necessitates recourse to some external—quite possibly subjective—basis of judgement. Even when testing clustering performance with known data, there is a difficulty: clusters are not definitively associated with any class, nor classes with any cluster. In cases where the majority of each class falls into one and only one cluster, a one-to-one mapping can be unambiguously made. When the clustering is less clean, it is less clear how to calculate accuracy.

Instead of accuracy, experimental results in this dissertation are quantified according to an information theoretic metric. With an externally given list of true class identities T and one of learned cluster labels L, a useful measure of how well L corresponds to T is the mutual information between them [13]:

$$I(C,L) = H(C) - H(C|L)$$

where H is the entropy function

$$\begin{aligned} H(C) &= -\sum_{c=1}^{K} p(c)\log p(c); \\ H(C|L) &= -\sum_{c=1}^{K}\sum_{l=1}^{K} p(c,l)\log p(c|l). \end{aligned}$$

Since the underlying probabilities are unknown they are estimated empirically:

$$\begin{aligned} p(c) &\approx \frac{|C=c|}{N}; \\ p(c,l) &\approx \frac{|C=c, L=l|}{N}; \\ p(c|l) &\approx \frac{|C=c, L=l|}{|L=l|}. \end{aligned}$$

When the cluster labels match the class identities perfectly, $H(C|L)=0$ and $I(C,L)=H(C)$. In order to fix the value of such a perfect clustering intuitively at 1, and to ensure that results from different datasets can be meaningfully compared, the mutual information is normalized to produce a final clustering score:

$$\hat{I}(C,L) = \frac{I(C,L)}{H(C)}$$

All reported clustering results are expressed in this manner.

2.3 Modelling Techniques

2.3.1 Hidden Markov Models

Sound data is inherently temporal in character. Nothing can be learned from a sound signal at any single instant in time; interesting structure emerges only in a succession of many discrete values. Preprocessing techniqes discussed in Chapter 5 are able to convert instantaneous one-dimensional samples into more informative vectors representing the average content of a signal over a short time period, but if this averaging is extended beyond a few tens of milliseconds, vital information is lost. An acoustic pattern cannot therefore be satisfactorily represented as a single datapoint in space, but must be treated as an ordered sequence of such points. They must be modelled in a

way that is sensitive to changes in the signal through time.

The hidden Markov model (HMM) is a class of probabilistic graphical models able to capture the dynamic properties of ordered observations. They are widely used for sequence analysis in numerous domains, including bioinformatics [29] and web traffic analysis [47], and play a particularly important role in speech recognition [25]. They are examined in the present work as building blocks for acoustic clustering systems.

An HMM relates observations—a known sequence $X = x_1, x_2, ..., x_T$—to a hidden state sequence $S = q_1, q_2, ..., q_T$, with $q_t \in \{1, 2, ..., Q\}$. Each q_t is mapped to its corresponding x_t by means of an output function B that defines the conditional $b_i(x_t) = p(x_t|q_t = i)$. This can be a probability table, if X are discrete-valued, or a distribution such as a Gaussian, if they are continuous. In this latter case, B is parameterised by $B = \{\boldsymbol{\mu}, \Sigma\}$. This can further be extended to a mixture, but in this work only single Gaussian outputs will be used. Regardless of B's architecture, it can *only* refer to q_t: given the current hidden state, the observation is conditionally independent of all other variables.

There is another, rather strong conditional independence property: the sequence of hidden states is assumed to be a Markovian process. This means that only causal influence on q_t is exerted by q_{t-1}. Of course, inferences about the value of q_t can be validly drawn from q_{t+1} and x_t as well, but non-local dependence is forbidden. This assumption greatly simplifies training and inference, while allowing many real-world sequential processes to be modelled quite well. The dynamics of the model are defined by two variables. The initial distribution $\boldsymbol{\pi}$ gives the (prior) probability of a sequence starting in each of the hidden states. The transitions from state to state are then determined by a trainsition matrix A, with $a_{ij} = p(q_{t+1} = j|q_t = i)$. The paramater set $\{\boldsymbol{\pi}, A, B\}$ defining an HMM is collectively denoted λ.

A trained HMM represents a distribution of sequences within a determinate output space. It allows every possible sequence of observations in that space to be assigned

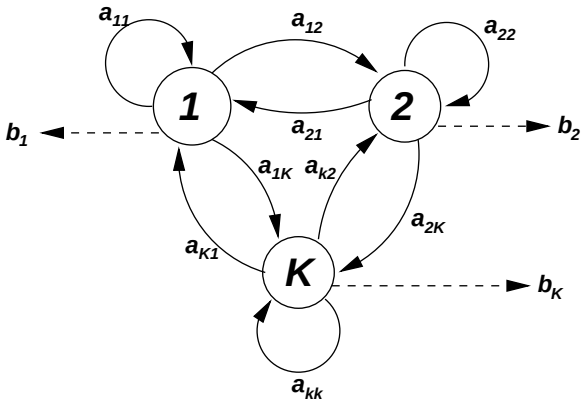


Figure 2.2: HMM shown as a finite state machine. The nodes are Markov states, corresponding to values of the latent variable q_t; the vertices are state transitions, labelled with their probabilities from the transition matrix A; and the dashed lines are outputs, defined by the probability distribution B.

a likelihood, which is just the posterior probability of the sequence being randomly generated by the model (in practice, the likelihood of most arbitrary sequences may be extraordinarily close to zero). This is equal to the sum of the joint probabilities with all possible state sequences:

$$p(X|\lambda) = \sum_S p(X|S,\lambda)p(S|\lambda).$$

Since the number of sequences grows exponentially with T, and the number of steps required to process each sequence grows linearly, computing this sum directly is $O(TQ^T)$. Fortunately, by summing the state probabilities of all paths at each timestep, it is possible to find the posterior in $O(Q^2T)$ [38]. This calculation, known as the forward-backward procedure, is given in Appendix A. The Baum-Welch equations for estimating maximum likelihood parameters can also be found there.

2.3.2 Mixtures of Hidden Markov Models

The familiar Gaussian mixture [2, 5] extends the basic Gaussian to accommodate a broader, more complex range of distributions, and in particular to handle data containing a number of distinct modes. A similar extension can be made to the HMM model

under the assumption that we face 'multi-modal' distributions in sequence space[2]. Mixtures enhance the power of a model family by representing a complex distribution as a combination of simple components. A mixture architecture may be adopted solely to obtain a better fit to data, even when the internal structure of the distribution is itself unimportant. There has been little interest in using HMMs this way because HMMs can generally be made more powerful simply by adding additional Markov states, while avoiding the complications of training a mixture. However, when internal structure in a dataset *is* of interest, the extension to a mixture of HMMs may be appropriate. A mixture architecture can tell us much about the structure of the data that is obscure in a single HMM, because it comprises not only a model of the overall distribution of sequences, but also a number of component models covering more-or-less well-defined subsets. Thus, although on the grounds of enhanced modelling power alone the merits of the HMM mixture are uncertain, it is a promising approach to the difficult task of clustering sequence data.

The Baum-Welch procedure for training a single HMM is can be extended to the mixture case, using soft cluster memberships based on posterior probabilities. Ypma and Heskes [47] have given EM update equations for HMM mixtures, and a full derivation is offered by Zhong [49]. Unfortunately, soft EM clustering with HMMs is computationally demanding. If hard cluster memberships are used instead, we have the HMM version of the CEM algorithm mentioned above; with the further simplification of uniform cluster probabilites, it becomes K-models.

In constrast to this standard EM-based training approach, several researchers have applied non-partitional clustering algorithms to HMM mixtures. Smyth [44] presents a hierarchical agglomerative method, where a singleton HMM is trained on every training sequence, and a distance metric based on the likelihood of each sequence under each model is used to iteratively merge clusters. Li & Biswas [31, 32] use a pseudo-hierarchical divisive technique, modelling the training set first with a single HMM and

[2]The metaphor of multi-modality is used here rather loosely, to convey the notion that a set of sequences may be more accurately and succinctly modelled as a number of separate Markov processes than as a single, more complicated process.

then adding more components to the mixture so as to split the set into progressively smaller groups. Law & Kwok [30] apply competitive learning, where a large number of HMMs 'compete' for training items; the most likely generating component for each item is trained on it, while its nearest rival is de-trained to push it away and persistent losers are eliminated. Finally, Oates et al. [35] use dynamic time warping to create an initial partitioning of the data, which is modelled and refined as a mixture of HMMs.

In this dissertation, three forms of HMM cluster training are implemented and compared: a simple K-models partitional algorithm; a hierarchical agglomerative method based on Smyth's [44]; and a hierarchical divisive approach with some similarities to that of Li & Biswas [31, 32]. These are discussed in detail in the next chapter.

Chapter 3

Hidden Markov Model Clustering

This chapter discusses in detail the model architecture that is used for the clustering task, and the methods employed to train it. The tools and techniques mentioned here are not specific to acoustic data, but can be applied to any process that generates time series. Section 3.1.1 discusses some specific characteristics of the HMM components (reviewed in 2.3.1) that are used here, and 3.1.2 gives the procedures by which these models are initialised and trained. 3.2 presents the cluster training algorithms in use: in 3.2.1, a method referred to as *K*-models that finds partitions of data based on a fixed mixture topology; in 3.2.2, an agglomerative hierarchical method that finds cohesive clusters by interatively merging similar groups; and in 3.2.3, a divisive hierarchical algorithm that starts by placing all items together and then splits groups in such a way as to improve the overall fit to the data. Finally, 3.3 gives some preliminary experimental results in which the clustering output of the three algorithms is compared on a range of synthetic data.

3.1 HMM Framework

3.1.1 Model Structure and Parameters

The state output function must generate observations like those in the time series to be modelled. Since the raw acoustic data is transformed into a sequence of real-valued feature vectors, a multivariate Gaussian output is necessary. In many speech appli-

cations, Gaussian mixtures have been found to offer better performance than single Gaussians [25], but they are not used here for several reasons. One is that the sheer complexity of the HMM mixture architecture poses daunting computational demands, and presents a real risk of overfitting on the limited datasets available; an additional layer of flexibility was not desirable. More fundamentally, though, the utility of Gaussian mixtures is to accommodate multiple modes in the data, such as speaker or gender differences; since these are just the sorts of distinction that we might want a clustering algorithm to discover, and since we provide no class information to ensure that the differences glossed over are the correct ones, there is a risk that mixed outputs would work at cross-purposes to the clustering task.

Instead, each state output is modelled as a single multivariate Gaussian distribution, represented by a mean vector $\boldsymbol{\mu}$ and a covariance matrix Σ. To discourage overfitting and improve stability with limited training data, Σ was constrained to be diagonal, forcing the distribution's directions of variance to be axis-aligned.

For many trials, the number of states in each HMM was uniform and constant, set pragmatically based on the outcome of a number of trials. Techniques for finding an optimal model size automatically are discussed in the next chapter. In speech recognition tasks, left-to-right HMMs are often used [25]. These impose the constraint that all entries in the state transition matrix below the diagonal be zero, allowing state transitions to move in only one direction through the model. No such constraint is imposed here: although symmetry and periodicity are not prominent in human speech, they may be important in other domains such as whale song, and there is no reason to limit the ability of the model to capture them. Model initialisation is also simpler with unconstrained transitions.

3.1.2 Initialisation and Training

The general framework for the clustering techniques presented here is a set of K Hidden Markov Models $\{\lambda_1, \lambda_2, ..., \lambda_K\}$, each consisting of

- A Q-dimensional initial state distribution vector $\boldsymbol{\pi}$

- A diagonal $Q \times Q$ state transition matrix A
- Mean vectors $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_Q$
- Covariance matrices $\Sigma_1, \Sigma_2, ..., \Sigma_Q$

The training of these components, and redistribution of items among them, was controlled by one of three clustering algorithms, presented below[1]. Before it is trained, each HMM is given data-dependent initial settings; these settings can be based on the whole dataset, a subset thereof, or a single item, determined by the algorithm. A is set such that the probability of remaining in the current state is relatively high, while the probability of moving to another state is uniform among these states:

$$a_{ij} = \begin{cases} \frac{Q+1}{2Q} & i = j \\ \frac{1}{2Q} & otherwise \end{cases}$$

The other parameters are set based on the data item(s) on which the HMM is being initialised. First, if the initialisation set contains more than one item, these are concatenated together, resulting in a single, potentially enormous set of vectors. If necessary, this set randomly subsampled down to 1000 for efficiency. Standard K-means clustering is then performed, with K set to the desired number Q of Markov states and using randomly drawn vectors as the initial centroids. The cluster centroids after convergence become the HMM output means $\boldsymbol{\mu}_q$. The covariance matrices for the final clusters are computed for use as the HMM covariances Σ_q, and the state probability vector $\boldsymbol{\pi}$ is set to the proportion of vectors falling into each cluster.

This initialisation method ensures that even before training, each HMM will have a rudimentary capacity to model kinds of observations that occur in the data. The computational cost incurred is relatively small, and the resulting model is quicker to train and less likely to get stuck in a very poor local optimum than one given purely random initial settings.

[1] All experiments were implemented in Matlab. Code for HMM training and likelihood evaluation was taken from Kevin Murphy's HMM toolbox, available at http://www.ai.mit.edu/~murphyk/Software/HMM/hmm.html.

3.2 Clustering Methodology

All clustering algorithms used here are 'hard', meaning that on each iteration, every item is assigned unambiguously to a single cluster (represented by a single HMM), and the HMM parameter updates are influenced only by those items currently in the associated cluster. The efficiency benefits of this are twofold. Each training iteration involves processing of each data item only once, instead of once per cluster under a soft scheme, so we perform N instances of the Baum-Welch procedure per cycle instead of NK. More importantly, the expected number of cycles is smaller, because cluster membership tend change by large jumps and then to settle to a static configuration once parameter changes become sufficiently small, leading to rapid convergence.

Cluster redistribution is deterministic, assigning each item x_i to the cluster k that gives it the highest posterior $p(x_i|\lambda_k)$. Cluster priors $p(k)$ are ignored in the current implementations. In principle these should be used to model the relative cluster sizes, and would be helpful in finding highly unbalanced target classes. In practice, however, the classes under study are approximately balanced, so making an implicit assumption to that effect in order to simplify the model should not detrimentally affect performance. In addition, there is a tendency for outliers to be placed in their own tiny clusters; the assumption of uniform priors helps to mitigate this.

3.2.1 K-Models Clustering

This clustering technique is simple and (comparatively) efficient. It is intended primarily as a baseline against which to compare the more complex methods. As a partitional algorithm, it requires K to be set at the beginning, and leaves it static during execution. The HMMs λ_k are each initialised on a single randomly selected data item, and then items are assigned to the clusters of maximum likelihood. The HMMs are trained on the items they are assigned, and the cluster assignments are recomputed based on the new parameters. This is repeated until the cluster memberships reach a steady state.

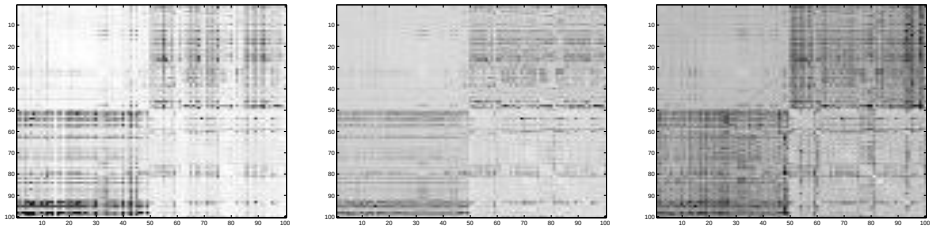
```
for 1 to K
    draw an item i at random (without replacement)
```

```
    create an HMM and train on item i
while cluster memberships change
    for each item i
        for each cluster c
            compute log likelihood of i under c
        assign i to its highest likelihood cluster
    for each cluster c
        retrain HMM on items in c
```

The singlton initialisation is intended to scatter the cluster models randomly through the data. Item are then moved among clusters until a stable arrangement is found. Each iteration of the algorithm involves N instances of HMM training, plus NK likelihood evaluations. The algorithm typically converges in fewer than 10 iterations, even for large datasets.

3.2.2 Agglomerative Clustering

A hierarchical agglomerative HMM clustering system was implemented, based largely on Symth [44]. Initially there are N singleton clusters modelled by HMMs $\lambda_1, \lambda_2, ..., \lambda_N$ each of which is initialised and trained on a single data item. An $N \times N$ distance matrix D is created by evaluating the log-likelihood of each item under each model, such that the less likely one item is under the model trained on another, the greater the distance $D_{ij} = -\log(p(x_j|\lambda_i))$ between them. Because 'distance' is just a measure of dissimilarity rather than any spatial quantity, negative distances are permissible. The columns of D are normalized to zero mean and unit variance. This step prevents outliers from having undue influence on the cluster output by ensuring that all distances are by forcing all inter-item distances into the same general range; without this step, items with generally low likelihood scores tend to dominate the merging and remain singleton. Column-wise normalization (the *in*-distances to each item from all others) was found empirically to be more helpful than row-wise (*out*-distance) normalization. Finally, since $D_{ij} \neq D_{ji}$, the matrix is made symmetrical by setting $Dij = \frac{D_{ij}+D_{ji}}{2}$. Figure 3.1 shows the effect of these manipulations on the distance matrix.



(a) **(b)** **(c)**

Figure 3.1: Manipulations applied to the distance matrix to improve clustering stability (2 classes of whale song): **(a)** The original log-likelihood distance matrix sorted by class identity for illustrative purposes. The 'criss-cross' appearance evinces highly uneven distance relationships. **(b)** Normalization. Outliers (dark bands) have been smoothed to be less remote from neighbouring items. **(c)** Symmetrization. The matrix has been averaged with its transpose, such that its upper and lower triangular portions are mutual reflections across the diagonal, further smoothing irregularities and strengthening the separation between classes.

On each iteration of the algorithm, the closest pair of clusters $argmin_{i,j} D_{ij}$ is selected to be merged, with their associated sets denoted c_i and c_j. The combined cluster replaces the lower-numbered merged cluster $k = min(i, j)$, with $c_k = c_i \cup c_j$ and j discarded. Distances $D_{kl}, D_{lk} \forall l$ are determined according to a merge heuristic. Smyth uses the complete-link (farthest neighbours) method, but this was found in comparative trials to give inferior performance to the average-link method, which is adopted instead:

$$D_{kl} = \frac{(|c_i| \times D_{il}) + (|c_j| \times D_{jl})}{|c_k|}, \forall l$$

and similarly for D_{lk}.

Even this complete-link merging procedure will inevitably introduce some distortion into the distance matrix, as the weighted mean of log-likelihoods of an item under two subcluster models is not necessarily a good predictor of its log-likelihood under a model trained on the combined cluster. Better results can theoretically be obtained by training an HMM λ_k on the combined cluster c_k and recomputing the distances $D_{kl} = p(x_l|\lambda_k)$. The performance gain from this procedure, though measurable is not

large, and the penalty in time and space requirements of the algorithm is substantial, particularly when the number of clusters is large. Distance re-estimation was therefore only used for the purpose of model selection (discussed in chapter 4), and even then, only once the number of clusters had been whittled down to a manageable number (typically 10) by the faster complete-link merging.

```
for each item i
    create an HMM m; train m on i
    for each item j
        compute distance ij as negative log likelihood of j under m
    normalize distances
while number of clusters > 1
    find nearest pair of clusters
    merge into a single cluster
    update distance metric using weighted average
```

Good results have been obtained from this algorithm on both toy and real data. However, it is highly resource-intensive. Computing the initial distance matrix requires N^2 likelihood evaluations plus N HMM training operations. With complex time series data, each of these evaluations is expensive; with the number of these calls quadratic in the size of the dataset, this algorithm has clear scaling problems. If the distance recalculation enhancement is applied, it incurs *another* $N(N-2)$ likelihood evaluations plus $O(NK)$ training operations.

3.2.3 Divisive Clustering

In divisive HMM clustering, we start with a single all-encompassing cluster modelled by a single HMM. This is then split by inserting new HMMs into the mixture, which capture items away from existing clusters. In principle, the algorithm can be run until it produces N singleton clusters, producing a full clustering hierarchy. However, it often makes sense to terminate long before this, as the most interesting information typically lies in the coarse-grained clusterings found in the early iterations. This makes the divisive approach dramatically more efficient and practical than the agglomerative

one for many tasks.

Li & Biswas [31, 32] present a divisive-like clustering procedure with integrated Bayesian model selection, which they call the Matryoshka algorithm. It is not a hierarchical method, because new clusters can capture items from any existing clusters, rather than being limited to a single cluster being split. A version of divisive HMM clustering is presented here, modified to allow only hierarchical splits and to remove the model selection elements, which are discussed in the next chapter.

For hierarchical division, the first step is to select a cluster i to split. The one chosen is the poorest-fit cluster, defined as the non-singleton cluster whose HMM assigns the lowest mean likelihood to its items. The initial HMM is trained on all available data items, to give a global model of the distribution. An item is then selected to serve as a seed for the new cluster. Li & Biswas use the item of lowest likelihood under the current clustering as the seed for new cluster. This approach tends to emphasize outliers, leading to behaviour where newly added clusters remain small, 'chipping away' at the dominant original cluster rather than splitting it fundamentally. Here, a seed item is drawn from the cluster at random. After an HMM has been initialised on this seed, the likelihood of the cluster items is computed under this new model, and the most likely $\frac{|c_i|}{2}$ are used as a training set for the new model. The existing model is retrained on the other half, hopefully allowing it to find a better fit, and then items are redistributed among the two models. This retraining and redistribution is repeated until the membership of the two models becomes stable; this is equivalent to K-models clustering with $K = 2$. The selection and splitting of clusters continues until a stop condition is reached or until all clusters are singleton.

```
train an HMM on the whole dataset
while K is below some threshold
    choose the poorest-fit cluster c to split
    choose a seed item i at random from this cluster
    create an HMM m and train m on i
    for each item i in c
```

```
        find likelihood of i under m
    assign the highest-likelihood half of all i to m
    while cluster memberships change
        retrain c and m on their current items
        for each item i in c or m
            compute log likelihood of i under c and m
            assign i to highest-likelihood cluster
```

This is a true hierarchical algorithm because new clusters are not added to the mixture at large, but to a specific cluster which is divided in two in a manner discrete from all other clusters. While this constraint slightly increases the overall likelihood of the fit that can be obtained, it makes the output more stable: once a division is made, it is permanent. This prevents 'rogue clusters' which happen to be initialised very near to a cluster boundary from pulling together items from previously well-separated clusters. Hierarchical clustering output is also more informative, as it preserves sub-cluster/supercluster relationships helpful in discerning the structure.

3.3 Comparing Clustering Methods

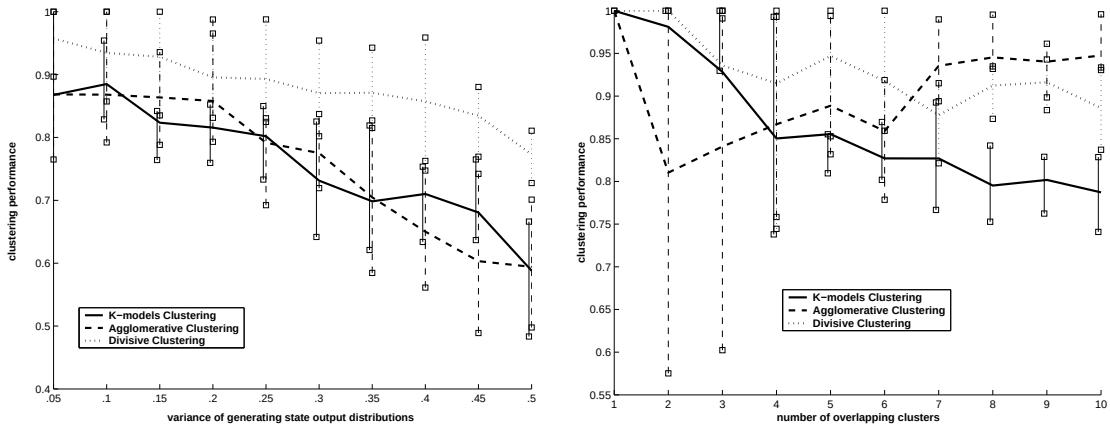
3.3.1 Experiments

The three clustering algorithms were tested on synthetic data to assess their relative performance under various conditions. Figure 3.2 shows the results of two experiments. In the first the number K of generating components was fixed at 5, as was the number Q of Markov states per component, while the variance σ^2 of the state output distributions was gradually increased causing the generating processes to become less distinct and thus less easy for the clustering algorithms to identify. Here, divisive clustering dominates, with its advantage growing as the task becomes more difficult.

In the second experiment (shown on the right) the output variance was held constant at $\sigma^2 = .1$, while more and more components were added to the generating mixture. Since the volume is fixed, this increases the degree of overlap among the generated

clusters. With only a few clusters, the divisive and K-models methods perform well, while agglomerative clustering does considerably worse. As K increases, however, the agglomerative technique comes into its own, ultimately outperforming divisive clustering. This discrepancy between the two experiments is interesting: though K-models is clearly inferior, the choice between the two hierarchical clustering methods is not as clear cut as it first appeared. Agglomerative clustering suffers when K is small, but improves markedly with larger values, even as the performance of the other methods deterioriates. This suggests that distance merging tends to distort inter-cluster similarity, but only in the late stages of the agglomerative procedure, when clusters are largest and farthest removed from the original singleton modelling.

In addition to its difficulty with coarse-grained clusterings, the agglomerative method has a significant liability in its time complexity. Given that divisive clustering offers better accuracy in many conditions and also scales with K^2N rather than KN^2, it would appear to be the superior general choice. However, empirical study on real data gives some surprising results, which are presented in chapter 6. It turns out that in some cases, the weakness of agglomerative clustering due to the approximate nature of its distance merging is outweighed by its powerful item-by-item initial distance calculation.



(a) **(b)**

Figure 3.2: Comparison of HMM clustering performance on an overalapping synthetic dataset: **(a)** constant $K = 5$ clusters, with increasing variance for the state output distributions; **(b)** constant $\sigma^2 = .1$, with increasingly many clusters in a fixed volume. Datasets contained 100 items; result is average of 20 runs, with each algorithm tested on the same 20 random datasets; error bars show lowest and highest values; results expressed in the mutual information validity measure discussed in 2.2.4.

Chapter 4

Model Selection

The Baum-Welch training equations optimize the free HMM parameters $\lambda = A, B, \boldsymbol{\pi}$ of fixed sizes $Q \times Q$, $Q \times 1$, and $Q \times 1$. Chapter 3 discussed how K such models can be trained in a mixture architecture. This chapter is concerned with setting the fixed structural parameters K and Q that define the topology of the mixture and its components respectively. The Bayes information criterion, which is used in both aspects of topology selection, is presented in section 4.1. Section 4.2.1 deals with choosing the number of clusters K, and 4.2.2 with the number Q of Markov states per cluster. 4.2.3 considers some implications of the fact that, for a model selection procedure for be truly automatic, both these unknowns must be optimized concurrently. 4.3 discusses how model selection can be incorporated into the HMM clustering procedures from Chapter 3, and offers some experimental results.

4.1 Evaluation Metrics

In order to perform model selection, it is necessary to perform comparisons among different topologies. Given a metric of how well-suited a model is to its data, it becomes a simple matter to train models of varying topologies and to adopt the most appropriate one. Several metrics are available, the most widely used being the Bayes information criterion.

4.1.1 The Bayes Information Criterion

The choice of model topology τ seeks to maximize $p(X|\tau) \propto p(X|\tau)$. Unfortunately, the computation is complicated by the model parameters θ:

$$p(X|\tau) = \int_{\theta \in \tau} p(X|\tau,\theta)p(\theta|\tau)p(\theta)\delta\theta$$

For most model architectures this integration is intractable [3], and so an approximation must be used. Since only trained parameters will be adopted, we can substitute maximum likelihood settings $\hat{\theta}$ for integration over all θ; the accuracy of the estimate will depend on how closely and consistently the training algorithm finds $\hat{\theta}$. Schwarz [42] developed an approximation, now known as the Bayes information criterion (BIC):

$$p(X|\tau) \approx \log p(X|\hat{\theta}_\tau) - \frac{1}{2}|\hat{\theta}_\tau| \log N$$

where $|\theta_\tau|$ is the number of free parameters in a model of topology τ, and N is the number of items in X. This estimate tends asymptotically to the true value of $p(X|\tau)$ as N increases.

Informally, BIC can be seen as a penalized likelihood metric. The first term is just the log-likelihood, measuring how well the model is able to account for the data; the second term is a penalty on model complexity, which serves to discourage unnecessarily large models that make inefficient use of parameters and promote overfitting. There are related methods that use different penalty terms, the most important being the Akaike information criterion (AIC) [1], which penalizes the log-likelihood by $|\hat{\theta}_\tau|$ instead of $\frac{1}{2}|\hat{\theta}_\tau| \log N$ and which asymptotically approaches the entropy of the model instead of the posterior probability [43]. Theoretical and empirical study has shown that, excluding a few highly specific cases, the model selection performance of AIC is inferior to that of BIC [27].

These methods bear an information theoretic interpretation. Both are closely related to the minimum description length principle [39]. This treats models as tools for stochastically encoding data, and favours those that do so most efficiently. A dataset can be

fully represented by means of a set of model parameters followed by a compressed string from which the data items can be reconstructed, with reference to the parameters. MDL selects the model that would support the most succinct such representation. The theoretical description length can be calculated in a range of ways, and it can be shown that both BIC and AIC are special cases of MDL [20].

Due to its computational simplicity and useful asymptotic properties, the BIC is used widely for model selection, and has been successfully applied to the problems of setting the number of clusters in cluster analysis [16] and to optimize HMM topologies [3]. Since a fully unsupervised clustering procedure must perform *both* of these tasks, the BIC will play a particularly pivotal role.

4.2 Optimizing a Two-Tiered Topology

The ultimate goal of a clustering system is to be able to perform effectively in the complete absence of metadata: we want to be able to produce a 'good' clustering, even when we have no prior knowledge about the number or nature of the processes that generated the data. Thus a general case learning procedure must find appropriate settings for *all* parameters involved in modelling the data, including the structural parameters K and Q. Here again the unsupervised nature of the task poses a challenge: since there is no uniquely correct way to quantify what makes a clustering good, it is not immediately clear how we would even begin to automatically structure a model to promote such output. This problem of model evaluation is compounded by another problem: there may be a very strong interaction between K and Q, and because they are discrete-valued, they are not susceptible to gradient descent methods. The horns of this dilemma are considered in turn, and a (partial) solution based on the Bayes information criterion is applied.

4.2.1 Selection over K: How many clusters exist?

It is possible that in a clustering task—where we do not know the class identities of items in a dataset or which ones belong together—that we may still know what classes

we want to find, or at least how many classes we expect there to be. This is the case in domains where the processes responsible for generating the data cannot be directly observed, but there is nonetheless some theoretical knowledge of these processes. When such knowledge is available, it greatly simplifies the clustering task: fixed-K methods need only be run once, while K-varying methods can cycle until the target value of K is reached and then return only this final clustering. To be fully general, however, a clustering system must be capable of functioning in a *completely* unsupervised way, in order to handle cases in which no external information about the data-generating processes is available. Here it becomes necessary to discover an optimal value of K based solely on the distribution of the data. This is necessary in order to find a good clustering, and may also be an interesting piece of discovered knowledge in its own right, giving an indication as to whether the dataset consists of a few well-separated classes, many small fragments, or even (in the null hypothesis case) is lacking in any intrinsic cluster structure at all.

The whale song dataset is such a case: the size of the distinguishable vocabulary of the whales is unknown (and is one of the interesting pieces of information that could potentially be recovered by cluster analysis). There is every reason to believe that there *is* such a vocabulary: the compositional nature of song patterns suggests that vocal units are tokens of distinct classes, with all instances of a class perceived by the whales as equivalent (at least with respect to the song structure), and that the number of such tokens is finite. Any answer to the question "which sets of units were sung as tokens of the same class?" includes an implicit answer to the question "how many classes are there?". Answering the second, seemingly incidental question turns out to be a crucial and difficult component of the overall task.

In some domains, and to a limited degree, the use of hierarchical clustering can bypass this problem. Items can be arranged in a tree structure showing the complete cluster lineage of each, allowing similarity relationships to be compared across all (or a range of interesting) cluster granularities. In some domains, this will constitute satisfactory output from a clustering system: if a user is given a choice among statistically

cohesive cluster assignments, the most relevant one (or ones) for the problem at hand will be subjectively obvious. If a clustering system can usefully support human analysis in this way, any limitations in its ability to choose K become moot. This motivates, in part, the decision in the previous chapter to favour truly hierarchical methods over ones whose stacked cluster relationships are more complex.

This resort to human interpretation, however, does not solve the problem of full automated cluster analysis, but merely evades it. In the domains where clustering is most potentially useful, human interpretation of data is difficult, tedious, or impossible. This is particularly true of temporal data, which is not conducive to simultaneous comparison. Even if whale song units were arranged hierarchically according to cluster divisions, the task of deciding which divisions are 'real' would be highly problematic. It is therefore desirable to incorporate model selection techniques into the clustering procedures, in order to evaluate and compare mixtures containing varying numbers of clusters.

4.2.2 Selection over Q: How many HMM states are required to model each cluster?

The performance of a clustering system based on an HMM mixture is dependent on each HMM being able to model effectively the set of items assigned to its cluster. In order to find a good fit, one that captures the common structure across cluster members without encoding extraneous noise, requires that model learning take place at the right level of abstraction. This is a function of model topology: the more Markov states that are available, the greater the level of temporal detail available to the model. With too few states, defining features of the cluster may be blurred together into a single, diffuse state that does not differentiate the cluster from others; with too many, idiosyncratic detail from the cluster members may be captured faithfully by excessive dynamic flexibility of the model, masking underlying similarities with outside items.

The problem is not simply one of finding the closest possible fit of model to data. Adding states to a model will *always* enhance its potential fit, until each timeslice of

each training item is modelled by its own zero-variance state and the likelihood tends to infinity. To perform well, an HMM must be able to generalise: outside states sharing the characteristics common to members of the cluster—even if they share no surface similarity—must be assigned relatively high likelihoods under the model, while states with a fundamentally different structure—even if they have superficial features or noise properties in common with a member of the cluster—must be judged dissimilar. Excessive values of Q allow the model to capture too fine a level of detail, which tends to obscure the essential character of the cluster. Model selection techniques attempt to find an optimal balance between power and generality, by quantifying the appropriatness of a given HMM topology to its assigned data.

Of course, in an unsupervised task, terms like 'optimal' are always problematic. With labelled classes, membership can be assumed to be correct and stable, and the modelling goal is only to fit them accurately and efficiently; clusters are inherently tentative, and models must serve to *refine* membership assignments by evaluating, for items outside the cluster as well as within it, how typical they are of the cluster. Model complexity determines the level of detail at which typicality will be assessed, and a data-driven way of making this choice can be based only on the statistical properties of the current, possibly quite poor cluster membership. Thus the level of detail chosen by any generic may not correspond to the features of fundamental interest, and an inappropriate early modelling can self-reinforce through the clustering algorithm. There can even be multiple valid and interesting levels on which a given dataset could be clustered—a speech corpus, for instance, could be grouped by word or by speaker. Clearly, such problems are not ones of model selection, and must be dealt with by removing or de-emphasizing aspects of the data irrelevant to the task at hand; this is dealt with in the next chapter.

4.2.3 Selection over $\{K, Q\}$: Which should be chosen first?

A more serious challenge to model selection, at least in theory, lies in the fact that the size of the global model and the sizes of its component cluster models must be optimized simultaneously. The relationship between K and Q is a circular one: the

greater the number of clusters, the tighter and more specialised each cluster will tend to be, requiring fewer states; and the greater the number of states available to model each cluster, the broader the range of data that each state will accommodate, tending to reduce the number of clusters. In the extreme case, it is possible for a single HMM to model two completely discrete subsets that could be modelled as separate clusters—and for these two represention to be almost completely equivalent in their ability to fit the data. Given hidden Markov models λ_1 and λ_2 with transition matrices A_1 and A_2, it is possible to construct an equivalent model λ_* with

$$A_* = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix}$$

with B and $\boldsymbol{\pi}$ simply concatenated and $\boldsymbol{\pi}$ normalized. The posterior probabilities assigned to data items by these two topologies will be identical, yet there is clearly no natural association between the items modelled by λ_1 and λ_2. Faced with a heterogeneous model like λ_*, a clustering system cannot be expected to isolate the underlying processes on the basis of likelihood alone, no matter how dissimilar they might be. Cases like these underscore the importance of a good model evaluation metric, and it is reassuring to note that BIC will favour the separation of λ_* into λ_1 and λ_2 (because $\log x + \log y < 2\log(x+y)$), whereas AIC would be indifferent between the two. There is no guarantee that a direct comparison will be made, however, and to discourage such pathological cases from arising, care must be taken in combining optimization procedures for K and Q.

Parallel optimization of K and Q can be incorporated into hierarchical clustering. Li & Biswas [32] take an approach of nested linear searches: an outer loop varies K (through cluster division), while an inner loop optimizes Q. Each time the clustering algorithm needs to create or update a cluster model, it initializes and trains a series of HMMs across a range of Q values, calculates the BIC for each, and accepts the one with the highest score. At the outer level, BIC is also used to select the overall best clustering. Each time a new cluster is added, BIC is recalculated; since it is in log-space, the combined score for the mixture is simply the sum of scores of individual clusters.

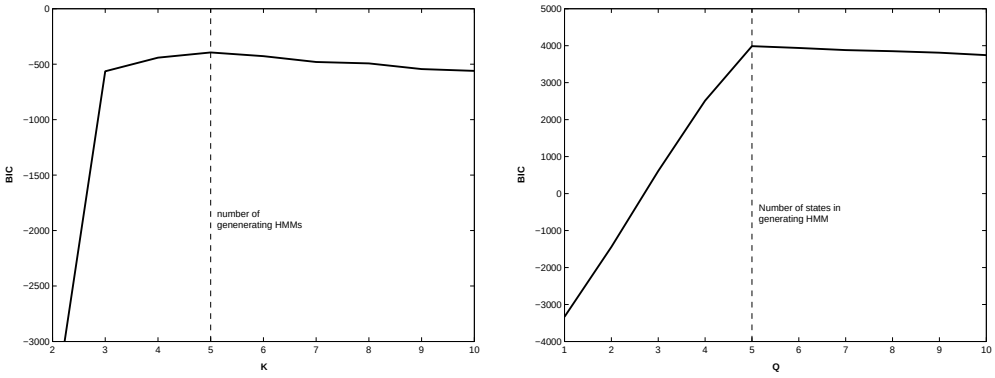
A similar procedure can be built into agglomerative clustering. An HMM is created to represent each newly merged cluster, seaching over Q. The winning model is then used to calculate the BIC.

Both versions return the clustering that gives a peak BIC value In principle, this maximum should be unique, with the BIC increasing monotonically until the growing penalty term comes to dominate the improving likelihood, and decreasing thereafter [32], implying that a BIC decrease can be used as a stopping condition for both searches. In practice, BIC tends to favour much finer-grained clusterings (higher values of K) for real acoustic datasets than the broad categories we typically seek. The inner search optimize Q is also problematic because it adds substantially to the computational complexity of the algorithm. Model self-selection is therefore not practical as part of a normal clustering procedure. Instead, structural parameters are chosen on the basis of experiments performed specifically for model selection, and for subsequent study these parameters are held fixed.

4.3 Experiments

Tests were performed on synthetic data to verify that model selection using BIC is able in principle to correctly discover the underlying structure of the data. Figure 4.1 shows that, when the underlying data classes are well separated, the BIC score peaks at correct values for both K and Q. The first experiment trained mixtures using K-models with Q fixed at the correct value; the second trained models of varying Q values on items from a single generating process. Combining these searches produces a BIC surface like the one shown in figure 4.2: the dashed line, indicating the topology of the generating mixture, passes through the peak BIC value, indicating successful selection of the combined model.

These tests were performed on clean synthetic data. In real acoustic datasets, where classes are typically complex and overlapped, model selection is more problematic. In general there is no uniquely correct value of Q; we simply seek a model topology that



(a) **(b)**

Figure 4.1: Nested BIC evaluations of model topologies, used in divisive HMM clustering. **(a)** Outer search: varying the number of clusters. Synthetic data from 5 generating processes, each with 5 states, with Q held constant at 5. **(b)** Inner search: varying the number of Markov states used to model a single cluster. Synthetic data from a single generating process with 5 states.

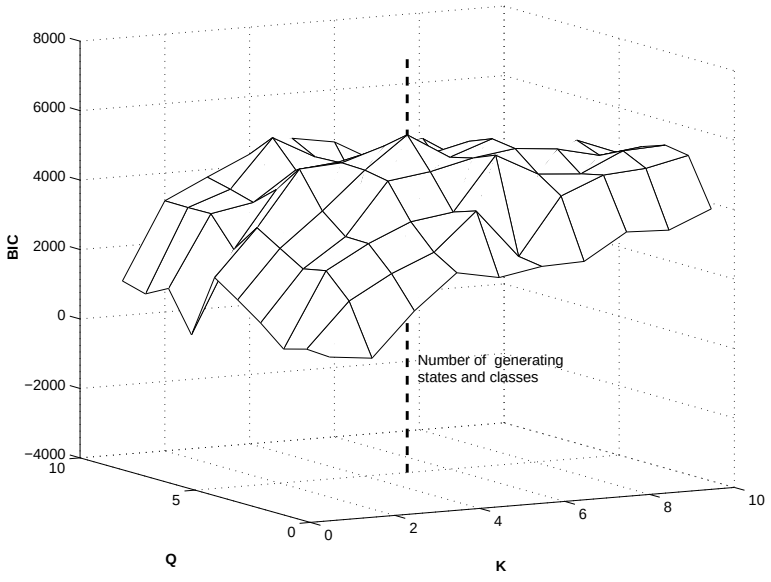
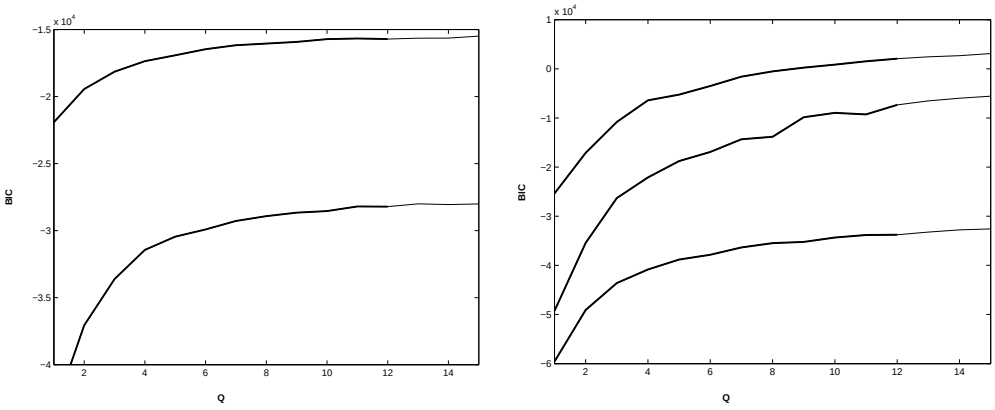


Figure 4.2: BIC surface computed using static topologies trained with K-models: synthetic data, $K = 5$, $Q = 5$.

can fit the data accurately yet parsimoniously. Unfortunately, when modelling complex data the BIC score can encourage rather larger models than are strictly necessary. Figure 4.3 shows BIC curves for HMMs trained directly on data classes. BIC does not reach a peak for any class within the range of Q explored. While still-larger values of Q may be optimal from the perspective of data modelling, they are not desirable in this context for two reasons. As model topologies grow more complex, and particularly when the number of states approaches the typical length of clusters, it becomes more likely that they will overfit the data relative to the high-level categories of interest. More pragmatically, HMM computations scale with Q^2, and so all else being equal, smaller Q values are preferred. Informal trials showed that above a very modest threshold, the impact of Q on classification accuracy was small. For remaining experiments the inefficiency and uncertainty of model selection was bypassed by setting $Q = 5$.

The choice of K is less arbitrary, as we assume it to have a uniquely correct target value that is meaningful in the problem domain. Unfortunately, this value may or may not coincide with the BIC peak. Figure 4.4 shows the BIC scores for various values of K during hierarchcial clustering. The speech data (left) has two categories of fundamental interest, but in Bayesian terms, its intrinsic cluster complexity is greater than 10, as no BIC peaks are found below this value (though agglomerative clustering places a small plateau at the correct value, which might be taken as a clue). In this case, the metric favours a more flexible and fragmented clustering than the high-level groupings of human interest; the improved likelihoods attainable on more compact subsets outweighs the added penalty due to growing model size.

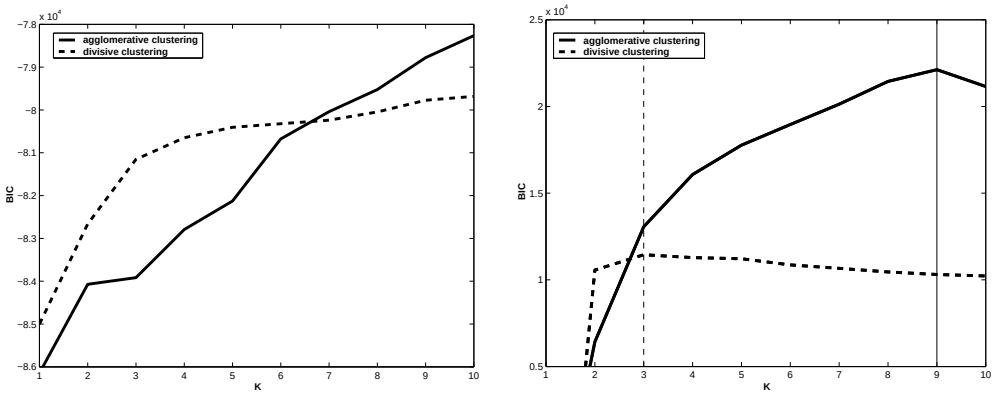
Model selection is more successful with whale song (right): divisive clustering correctly selects the number of subjective class labels in the sample. This suggests that these classes may correspond to strong intrinsic groups in the data, with little extraneous structure—exactly the scenario we hope for in model selection. This structure is not unambiguous, however: a higher BIC was attained by agglomerative clustering using 9 clusters instead of 3.



(a) **(b)**

Figure 4.3: BIC curves for varying Q, class-specific HMMs trained on **(a)** 2 word classes; **(b)** 3 whale song classes. 10-run average.

Model selection is not foolproof, and the BIC does not provide a definitive or automatic solution to the problem of optimizing model topology. However, it is a useful tool for investigating the intrinsic structural properties of a dataset, and can be of assistance in making an informed choice of settings for a cluster model.



(a) **(b)**

Figure 4.4: BIC curves for varying K, from hierarchical clustering of real data: **(a)** speech (2 word classes); **(b)** whale song(3 largest classes). 10-run average. Peaks, when found, are shown as vertical lines.

Chapter 5

Preprocessing

The techniques discussed so far have dealt with time series clustering in general, and are applicable to any data domain that produces ordered sequences of vectors of unknown class identity. Our particular task, however, is the clustering of sounds, which present some specific challenges, before modelling can even begin. This chapter draws on digital signal processing techniques, particularly ones developed in the context of speech recognition, in order to transform a raw acoustic dataset into a form that can be effectively dealt with by the means available for modelling and clustering acoustic data in general and humpback whale song in particular.

Section 5.1 briefly discusses the method by which the units to be clustered were extracted from the continuous audio stream. Section 5.2 presents a number of preprocessing techniques used to convert these sound samples into a form conducive to clustering: Fourier analysis in 5.2.1, cepstral analysis in 5.2.2, independent component analysis in 5.2.3, and feature deltas in 5.2.4. Finally, some results are given comparing the relative merits of these feature transformations in the context of supervised learning in 5.3.1 and clustering 5.3.2.

5.1 Unit Extraction

Given an extended stream of recorded audio, no modelling or clustering of its constituent segments can be performed until these segments can be isolated. In many domains the interesting units of sound may be poorly separated, making segmentation a difficult problem in its own right. However, in his study of the whale dataset, Warin [45] found that simple volume thresholding was able to successfully delineate most of the song units, which tend to be quite discrete. After performing spectral analysis (see below), he computed the sum of squares of Fourier coefficients for each timeslice. He then defined a threshold for this energy value, below which a slice would be considered silent. Four consecutive slices above the silence threshold was considered interpreted as the beginning of a unit, while two below it marked the end, subject to a maximum unit length.

While Warin's method does not seem problematic, there is also a compelling practical motivation to follow him in this regard: the extracted units have been subjectively labelled [46], and a new extraction process would entail discarding this information. In order to make use of these category judgements in evaluating clustering results, as well as for continuity with previous work, Warin's original units have been adopted.

5.2 Feature Transformation

A raw acoustic signal is a sequence of one-dimensional magnitudes corresponding to instantaneous measurements of air pressure, sampled at a rate between 8 and 44 kHz. In this form the signal is not susceptible to available statistical modelling techniques. Each sample is meaningful only in relation to others, and the time resolution afforded by the rapid sampling rate, the generating process is radically non-Markovian: strong correlations exist across large numbers of samples, and the information of the signal lies the waveform sketched out by this intricate web of non-local relationships.

Transforming this one-dimensional sequence into a more usable representation is essential to the success of any modelling task. The information in thousands of cor-

related samples must be extracted and gathered into larger quanta whose properties can be non-trivially examined and compared; only from such substantive chunks can we hope to infer class-specific properties with any degree of generality. The kinds of sound categories that are interesting to humans are defined by very high level features. The relevant level of abstraction must be made explicit in the data, in the sense of being within the discriminative power of the model class in use, before modelling can succeed.

In *unsupervised* modelling, however, feature extraction is especially critical. In classification, the abstract features defining the classes of interest must be represented in a form discoverable by the model; in clustering, this is necessary but not sufficient. If group distinctions are to be discovered automatically, the features on which those distinctions are defined must be, in some statistical sense, the *most prominent* features in the data. If the important and useful properties are clearly and explicitly represented, but produce less variance than do extraneous features or random noise, the clustering will fail to focus on the important distinctions.

This makes clustering substantially more challenging than classification, and it would be unrealistic to expect comparable performance on the two tasks. However, under the assumption that class identities are unattainable or unreliable, we have no choice but to extract and emphasize the meaningful properties of the signal as selectively as possible. As noted in chapter 4, there is an inherent indeterminacy in this task: the features that are meaningful with respect to one clustering goal may be irrelevant to another. Thus there can be no universally applicable preprocessing regime for clustering sounds, and decisions made here are inevitably influenced by the datasets under study. However, an effort has been made to avoid ad hoc, domain specific techniques and to favour those likely to be of value under broadly applicable assumptions.

5.2.1 Spectral Analysis

The first and most fundamental step in the analysis of an acoustic signal is the discrete Fourier transform (DFT). This powerful operation converts a raw signal, said to inhabit

the *time domain* because each value corresponds to an instant in time, to the *frequency domain*, where values refer to specific frequencies spanning the whole time period. The basic formula [7] is

$$DFT_f(x) = \frac{1}{n}\sum_{t=0}^{n-1} x_t e^{-2\pi i f t}$$

where n is the length of the signal and f is the frequency for which the transform is being calculated, expressed as a fraction of n. Naïvely this is $O(n^2)$, but an efficient algorithm exists (the fast Fourier transform (FFT)) to compute it in $O(n \log n)$, making it feasible for long signals.

An FFT is performed for a number of values of f, decomposing the signal into a number of bins estimating the energy at determinate frequencies. Of course, the frequency content of a sound recording is not static for its entire duration—the pitch can rise or fall, and the pattern of overtones can change substantially. Clearly these dynamic properties are important and cannot be discarded. However, the timescale of such change is (necessarily) very much larger than the sampling rate. Therefore, the frequency signature of a signal can be calculated as a series of discrete timeslices; the span of the window (on the order of several hundred samples) is chosen to be large enough that frequency content across each slice can be computed accurately, but short enough that the succession of timeslices can capture the dynamic behaviour of the signal. The result is a series of timeslices, each consisting of a vector of frequency bins; this time-versus-frequency representation is a spectrogram. In this study, as in most acoustic modelling work, the spectrogram is adopted as a baseline representation: all methods require at least this degree of preprocessing, and further transformations assume input of this form.

The width of the moving FFT window can be modified to adjust the characteristics of the resulting spectrogram: larger window sizes provide more precise estimates of frequency content but coarser time resolution. Warin [45] investigated various FFT parameters for the whale dataset, and found non-ovelapping windows of 512 samples each, over 50 linearly spaced frequency bins capped at 4 kHz, to give a good combination of time and frequency resolution. These parameters have been adopted here.

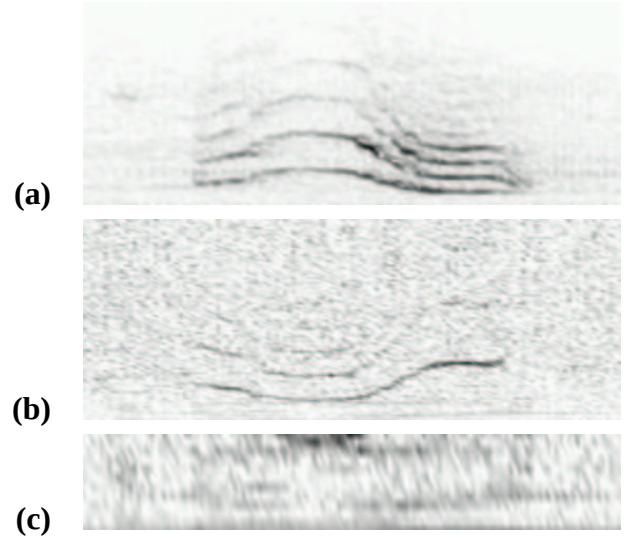


Figure 5.1: Spectral and cepstral representations of a whale song unit: **(a)** spectrogram; **(b)** cepstrogram; **(c)** first 12 cepstral coefficients.

5.2.2 Cepstral Analysis

A spectrogram is a representation of the frequency content of a signal, changing over time. However, many interesting sound features are not frequency-specific; a particular word, for instance, can be spoken at a broad range of different pitches. A means of extracting the pitch-invariant properties of a signal, widely used in speech recognition [17], is an extension of spectral analysis known as cepstral analysis. The calculation takes the inverse FFT of the logarithm of the spectrogram. This transforms the spectral representation back into the time domain, but now each value refers not to a particular instant, but to periodicity across the spectrum. Low-numbered cepstral values correspond to features spread smoothly across adjacent frequency bands, while higher cepstra capture more scattered harmonic components. This makes cepstral analysis a powerful tool for speech recognition: the effect is to *deconvolve* the vocal sound source, which is highly frequency-periodic, from the resonance produced in the vocal tract that defines the phonetic character of the sound [18]. This transformation is usually enhanced by spacing the original FFT frequency bins on the mel scale [33], which uses linear spacing below 1000 kHz and logarithmically thereafter; this is intended to approximate psychoacoustic properties of the human ear. The resulting mel-filtered

cepstral coefficients (MFCC) representation has become standard in speech recognition applications [24]. Typically only the first 9-16 coefficients are used, as these contain most of the linguistically interesting information in human speech.

In domains outside of speech, however, it is not clear that such a representation is appropriate. Where the generating process is unrelated to human perception, there is no a priori reason to suppose that human perception is particularly well-adapted to process it; indeed, one of the motivations for acoustic clustering is that it can bypass our perceptual limitations. For this reason, mel scale filtering should not be considered helpful in general clustering problems, unless the desired behaviour of the system is to mimic human performance. The whale song domain is clearly one where human-specific preprocessing would be out of place, as the sounds are generated not only by another species but in a different acoustic medium, and current research in this domain [11, 6] avoids the use of MFCCs.

Cepstral analysis itself, however, is less domain specific. In nontrivial acoustic domains, there is variability in spectral shape, the resonance characteristics of the sound, independent of pitch. If clustering is to focus on these properties, it is essential that they be represented in a way that is relatively pitch-invariant. Cepstral analysis, in its ability to deconvolute resonant features from harmonic ones, is general enough to be potentially useful across a broad range of clustering problems. Whether or not it should be employed, and which cepstral coefficients should be retained—low ones for phonetic-like features, higher ones for harmonic information—can only be decided in the context of a particular task.

5.2.3 Decorrelation

To avoid the domain specificity of cepstral analysis, it is desirable to adopt a feature extraction technique that relies only on general statistical properties of the data, making no assumptions about the generating processes. Among the numerous techniques available for transforming vector data into a (possibly lower dimensional) feature set that highlights the most salient information, one that shows particular promise for sig-

nal deconvolution is independent component analysis (ICA) [22]. This is a latent variable model based on the principle of minimum mutual information among the latent variables. Observation vectors $\boldsymbol{x}$ are modelled as

$$\boldsymbol{x} = M\boldsymbol{s} + \boldsymbol{n}$$

where $\boldsymbol{s}$ is a set of independent features (that is, $p(\boldsymbol{s}) = \prod_i p(s_i)$), M is a mixing matrix, and $\boldsymbol{n}$ is a Gaussian noise term (which is often ignored to simplify computation). M need not be square, allowing ICA to serve as a dimension reduction technique. To transform $\boldsymbol{x}$ to $\boldsymbol{s}$, we need to find a demixing matrix W such that $WM = I$ and consequently $W\boldsymbol{x} = \boldsymbol{s}$; efficient algorithms are available for this purpose[1].

Because ICA minimises the mutual information among features, it is useful for disentangling unrelated features of a dataset. It has been used for blind source separation and signal deconvolution[22], including feature extraction for speech recognition [37]. It is clear that a decorrelated feature set will improve class separability, allowing the models to concentrate on only those independent features associated with class identity. The effect of decorrelation on clustering is less certain, and is examined empirically below.

5.2.4 Deltas

The statistical modelling of time series is predicated on the belief that the (important) dynamic properties of the data can be learned by the model. The HMM requires as input only a sequence of discrete observations; the manner in which they change through time is inferred and stored in the transition matrix. Under this assumption, there is no reason to try to encode dynamics in the observations themselves. However, if an HMM interpreted *every* change in sound features as a transition to a new Markov state, it would be badly overfit. Clearly there are some local dynamic properties that are not captured in the Markov chain, and these may be useful. It might be, for example, that the salient aspect of a particular sound category is a rising or falling pitch, while the absolute pitch itself is irrelevant.

[1] Independent components for the experiments presented here were computed using Hyvarinen's FastICA toolbox for Matlab, available at http://www.cis.hut.fi/projects/ica/fastica/.

This speculation holds true empirically. HMM classification performance has been found to improve if the static feature vectors it is given are supplemented with their first order and in some cases second order deltas [21]. Thus each HMM state can model not only the chosen set of features at a given instant, but also their rate and direction of change, and the acceleration of that rate. The calculation is a windowed linear regression for each time-value of each feature [18]:

$$\Delta x_t = \frac{\sum_{d=-D}^{D} d x_{t+d}}{\sum_{d=-D}^{D} d^2}$$

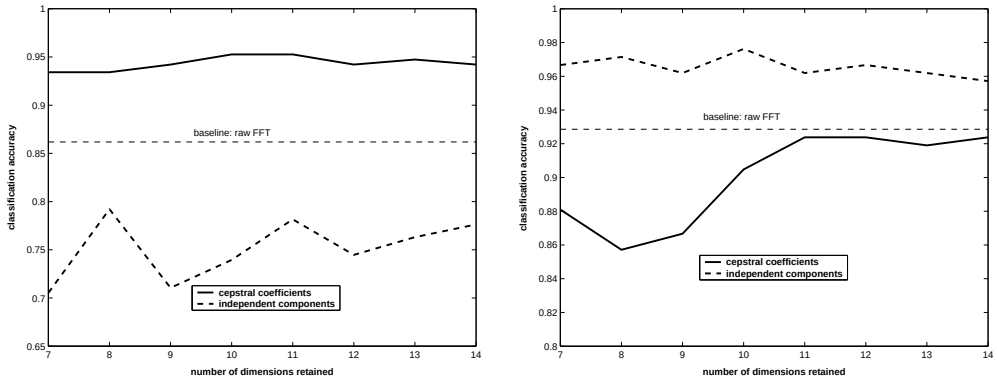
where D is number of frames ahead and behind to include in the window. The second order deltas can be found simply by applying this formula recursively. The results are appended to the base feature vectors.

As usual, techniques found useful in the context of speech cannot be automatically extrapolated to general acoustic tasks. Still, the dynamic behaviour of sound features is clearly relevant, on some level, to almost all modelling tasks. The use of deltas to make local changes explicitly available to the model should, in at least some cases, enhance clustering performance.

5.3 Experiments

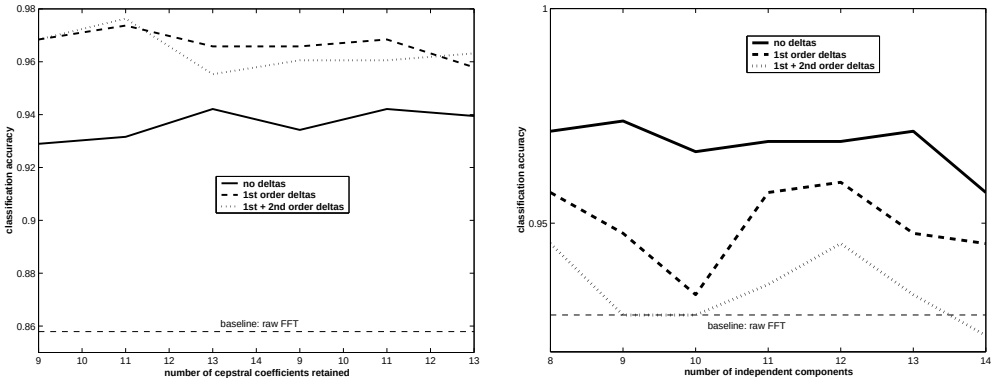
5.3.1 Supervised Trials

Initial testing of various feature transformations was conducted using supervised HMM learning of the separate data classes. As mentioned, for clustering to succeed it is a necessary but not sufficient condition that the data classes be separable by the modelling architecture used; supervised learning allows this separability to be verified in a controlled way, avoiding the complexity of clustering. Thus, while supervised tests cannot alone establish an optimal clustering representation, manipulations that *harm* classification accuracy significantly are unlikely to be helpful in clustering.



(a) **(b)**

Figure 5.2: Supervised learning with cepstral coefficients and independent components: **(a)** speech; **(b)** whale song (3 largest classes).



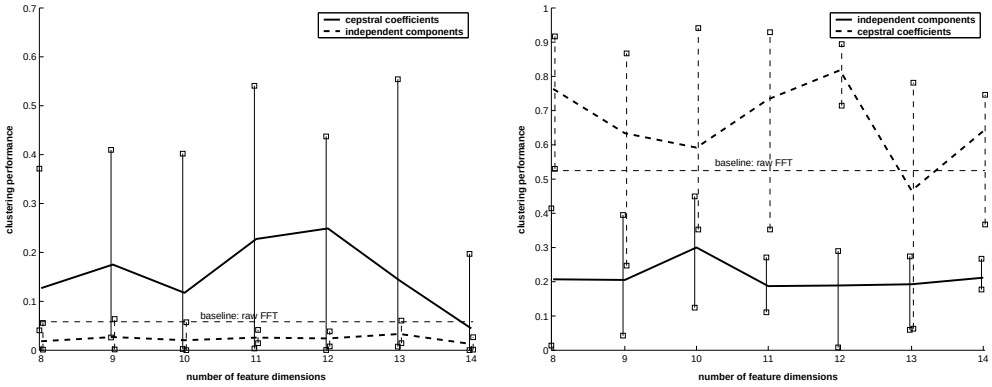
(a) **(b)**

Figure 5.3: Supervised learning with cepstral coefficients and deltas: **(a)** speech; **(b)** whale song (3 largest classes).

All experiments were performed on both the speech and whale song datasets. Results are shown side by side, with speech on the left and whale song on the right. The subjective labels for the whale dataset were treated as ground truth classes; in order to simplify this investigation of preprocessing techniques, only the three largest such classes of were used. Classification accuracy was measured with 10-fold cross validation. Items in the held out test set simply assigned to their highest likelihood models, ignoring class sizes (which were approximately uniform).

Figure 5.2 compares classification accuracy using the cepstral coefficient and ICA representations. The contrast between the datasets is stark: the whale classes prove to be substantially more separable when represented as independent components than as spectral or cepstral coefficients, while the opposite is true of speech. This effect is so strong that in each case, the inappropriate representation was actually less separable than the original spectrogram. This suggests that classes are defined by qualitatively different criteria in these datasets: word distinctions inhere primarily in the resonant properties emphasized by the cepstral transform, while labels have been applied to whale data based on a broader set of sound characteristics including harmonics and pitch.

Figure 5.3 illustrates the effect of deltas on class separability. In each case, the transform appropriate to the dataset was fist applied, as indicated by the previous experiment (cepstra for speech, ICA for whale song). For speech data, first-order cepstral deltas clearly allow for better classification that cepstra alone, while the additional effect of second-order deltas is marginal and generally negative. With whale song, even first-order independent component deltas are harmful to classification, and second-order ones more so. We may infer from this that the whale song categories with which we are working are less defined by their dynamic features than human speech. This may indicate that whale song has less fine temporal structure, or simply that this structure was not the main basis of classification used by the human labeller.



(a) **(b)**

Figure 5.4: Comparison of clustering performance with cepstral coefficients and independent components: **(a)** speech; **(b)** whale song (3 largest classes).

5.3.2 Unsupervised Trials

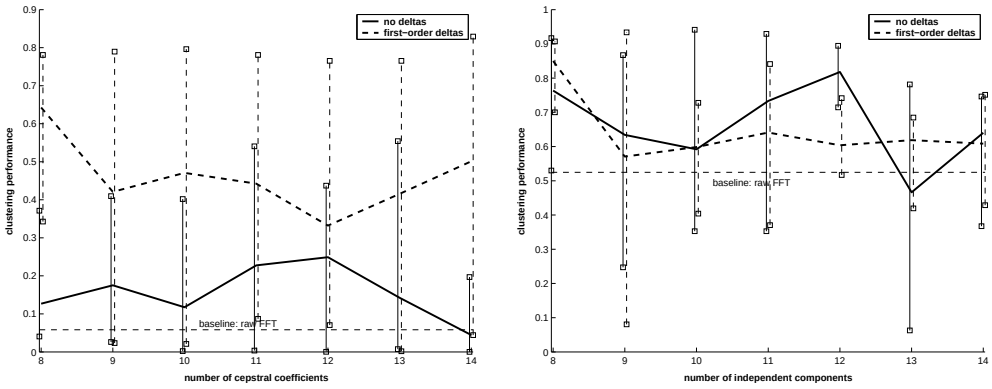
Using the supervised trials to narrow the range of promising transformations, feature representations were then evaluated on the more rigorous task of clustering. Results shown are 5-run averages, with error bars showing the best and worst cases. The performance metric used is the empirical mutual information between clusters and external class labels, explained in 2.2.4. The clustering method applied is hierarchical agglomerative clustering.

Figure 5.4 shows the results of clustering with cepstral and independent component representations. On the left, speech is clearly better represented by cepstral coefficients; clustering on independent components gives near-chance performance, and spectrograms fared little better. On the right, whale song spectrograms allow for modestly successful clustering, which is improved by ICA, but significantly damaged by cepstral analysis. These findings confirm the supervised results: the characteristics of these datasets, or at least of the groupings we are attempting to find, are fundamentally different, and different procedures are required to extract the relevant features. The speech clustering task also proves to be a more difficult one, as performance on this

data in markedly worse than that seen on whale song. It would seem that the distinctions between words are extremely subtle in spectral form, and are partially obfuscated by irrelevant variation in the data even after cepstral analysis has been applied.

Figure 5.5 shows how first order deltas affect clustering. Second order deltas were not used because, by increasing the feature dimension, they would exacerbate the already substantial computational overhead of these experiments, and because they were of little benefit in supervised classification. For speech (left), first-order cepstral deltas again show their value: their addition to the basic cepstral representation yields a significant performance gain. With whale data (right) deltas were generally unhelpful and perhaps even harmful. The supports the previous speculation about the relative importance of dynamic features in the two datasets.

In none of these results is there a strongly evident trend with respect to the number of feature dimensions used. This is attributable in part to the small size of these experiments, in terms of both the number of runs and the range of parameters, dictated by limitations on time and computational resources. Averaging a larger number of runs would produce smoother curves, which would hopefully reveal an unambiguous optimum, possibly one outside the range explored (it is puzzling that the two optimal dimensions for speech appear to be at the endpoints). However, the available results are adequate to guide the choice of reasonable preprocessing settings for these datasets; finding absolute optima is not necessary for our purposes. The feature representations adopted are as follows: for speech, the first 10 cepstral coefficients with first-order deltas; and for whale song, 12 independent components without deltas. These representations are treated as standard and used in all experiments on these data sets, including those presented in chapter 4.



(a) **(b)**

Figure 5.5: Comparison of cepstral coefficients and independent components, including first-order deltas, for clustering: **(a)** speech; **(b)** whale song (3 largest classes).

Chapter 6

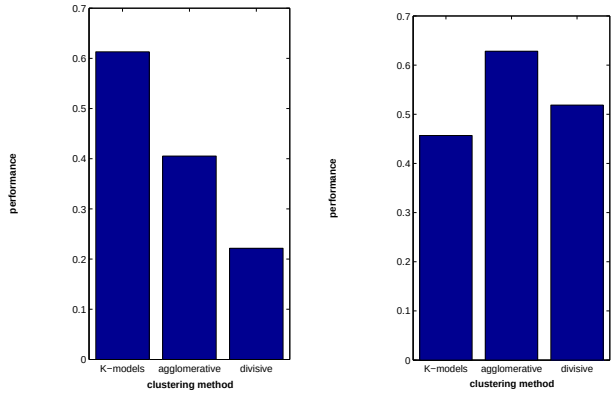
Results and Conclusion

The previous chapters have offered a collection of techniques potentially useful in acoustic clustering tasks. This chapter addresses the problem as a whole, considering the overall performance of a clustering system composed of the aforementioned techniques, applied to the two datasets under study. Section 6.1.1 shows the empirical performance characteristics of the three clustering methods tested on the two datasets, and draws some inference about their general characteristics. Section 6.1.2 discusses the results with reference to the speech dataset, and 6.1.3 gives some additional results for the whale song task, partially addressing the indeterminacy of 'pseudo-labelled' data. 6.2 summarizes key findings and offers some concluding remarks. In 6.3 a number of avenues for further research are proposed: 6.3.1 gives ideas pertaining to general HMM clustering of acoustic data while 6.3.2 offers thoughts for the ongoing automatic analysis of the humpback whale song corpus.

6.1 Discussion of Results

6.1.1 Clustering Algorithms

Figure 6.1 gives a summary of the relative performance of the three clustering methods. These contain a number of surprises, relative to the preliminary investigation involving synthetic data reported in chapter 3. The first is that divisive clustering is inferior to agglomerative clustering under both datasets. As agglomerative clustering



(a) **(b)**

Figure 6.1: Relative performance of K-models, agglomerative, and divisive clustering methods: **(a)** speech **(b)** whale song (3 largest classes)

works by means of a long sequence merge operations, each of which sacrifices some information, it might be expected to be unstable in large and complex datasets, and early experiments seemed to verify this. However, the initial item-by-item modelling captures a detailed and accurate map of item relationships, and we now find that the merge procedure manages to preserve the information necessary for accurate and ongoing judgements about class similarity. Though counterintuitive due to nomenclature, agglomerative clustering can be seen as a divide-and-conquer approach to clustering: rather than try to analyse a whole set of items at once, it processes them in examined in the simplest possible context—in isolation—and then uses simple binary relationships to bootstrap a model of of the whole global structure.

Conversely, divisive clustering is faced with the task of breaking up a potentially large set of items into cohesive subsets. When there really are distinct, well-defined groupings latent in the data—as there were in the synthetic task—this can be done quite effectively. However, when the set is relatively amorphous, with no naturally clean line of division, the way it is split can be somewhat arbitrary: the algorithm can chip off a tiny subcluster, or cleave the larger set evenly in two, depending in part on its random initial state. Its chief advantage is speed: because it starts from the top and works down, bypasses the laborious problem of modelling each item, and can start finding

large and potentially interesting groups right away.

The most interesting fact of these results, though, is the unexpectedly good performance of *K*-models clustering on speech data, particularly since it was not replicated in the whale song experiments. It would seem that the speech dataset has certain characteristics that make it ill-suited for hierarchical methods, but that do not pose a problem for *K*-means. What characteristics these might be, and how general they are, are left as open questions. However, it seems fair to assume that, although agglomerative clustering is the preferred method for generic acoustic, data, it is not effective on speech data and in these cases the *K*-models should be used instead.

6.1.2 Speech

As the speech dataset contains only two categories, it is an apparently minimal clustering task. However, the acoustic properties that distnguish one word from another are extremely subtle and masked by numerous forms of irrelevant variation, making good clustering elusive and challenging. Effective *supervised* modelling of speech has occupied researchers for decades; attempting to infer word categories *without* guidance seems downright quixotic. The results attained, though far from clean, demonstrate in principle that HMM clustering systems can discover even deeply hidden structure in sound. The magic lies in the preprocessing: if a feature transformation can be devised that selectively emphasizes those properties in the dataset which are salient with respect to certain category distinctions, then clustering should be able to find these distinctions. For this dataset, a feature representation consisting of 10 cepstral coefficients with first-order deltas was chosen. This allowed for a cluster validity measure of as high as .613, which corresponds to a classification accuracy of .9; this is deeemed a rather encouraging success.

Of course, there is some question begging going on here: once results from preliminary experiments, evaluated against ground truth, are used to tune the clustering system, the overall methodology can no longer be considered unsupervised. However, we have already that no *purely* unsupervised analysis of complex sound data can be expected to

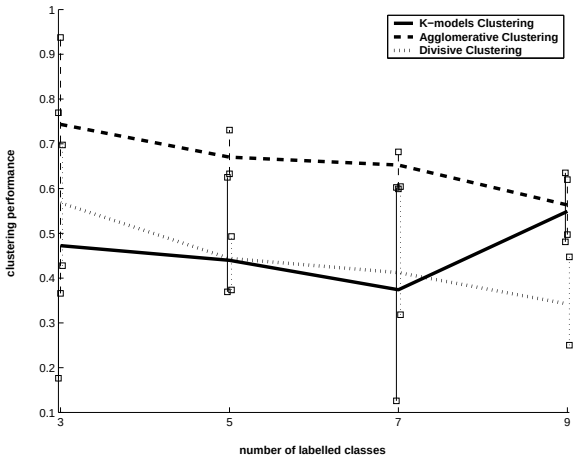


Figure 6.2: Clustering performance on increasing subsets of the whale song corpus. 10-run average; error bars show best and worst cases.

produce interesting or useful results, because the task is too unconstrained. By making a choice of one feature representation over another, the user encodes a preference for one sort of cluster over another. While such a decision is bound to be problematic for completely unknown datasets, it can plausibly be made without recourse to external class labels (as were used here) by making educated guesses about the sorts of sound features likely to be useful, and by drawing analogies to more familiar domains. The study of preprocessing in Chapter 5 provides the beginnings of a general 'palette' of transformations with well understood properties, available to be applied to new problem instances based on what is known or suspected about the dataset and the classes within it.

6.1.3 Whale Song

Most experiments reported for whale song used only three classes of data, as defined by the the subjective labels that have been applied to the set. This was intended to provide a stable problem of moderate difficulty on which to develop and test techniqes. This simplifying step was taken because the original labelling scheme presents some particularly stringent challenges to clustering—it contains 23 classes, and the class are extremely unbalanced, ranging from 2 to 102 items and with only 9 classes

larger than 20. Here, we ease this constraint slightly in order to scale back the complexity of the dataset. Clustering is performed on the K largest classes of whale song units, $K \in \{3,5,7,9\}$. As mentioned in Chapter 5, feature extraction consisted of independent component analysis down to a 10-dimensional feature vector, with no deltas added. The validity of cluster output was assessed relative to the subjective class labels. The results are shown in figure 6.2. The relative effectiveness of three clustering methods mimics those of figure 6.1. It is encouraging to note that clustering performance deteriorates only gradually as more clusters are added.

However, it does deteriorate. The normalised mutual information between cluster assignments and class labels for agglomerative clustering of 9 classes is only .563. From this modest result, the simplest conclusion is that the HMM clustering techniques discussed here are effective only when the number of classes is small, and produce increasingly arbitrary output as K grows. However, recall that the class identities in use are not'ground truth' in any absolute sense, but merely a human judgement about which units are sufficiently similar to be grouped together. It is possible that the clusters identified by the automatic methods are *more* internally similar in some substantive sense than those identified by the human labeller. Indeed, the descriptions given to the labelled classes give cause for suspicion that the objective overlap may be substantial: the class defined as "some harmonics and an up sweep" is distinct from the one called "up sweep with some harmonics." To test the hypothesis that the learned clusters allow for *better* modelling of the data, the subjective and machine-generated labellings were compared by means of supervised learning. The results are shown in figure 6.3: the data likelihood of HMMs trained on the output of the clusterinng algorithm is consistently higher than that achived using the human labels. This doesn't establish that the human labels are wrong or that the automatic ones are more meaningful in the domain, but it is reassuring that the clustering procedures do what they are supposed to do, which is to find clusterings that allow high likelihood modelling of the data.

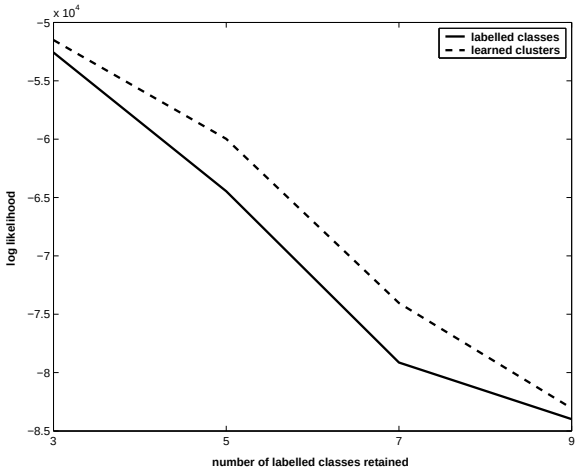


Figure 6.3: Comparison of data likelihoods of HMMs trained on the subjective classes and learned clusters (generated by agglomerative clustering).

6.2 Conclusion

Cluster analysis of unknown acoustic data is a hard problem, and not one that admits of a fully general solution. Sounds can be similar to, and different from, one another in vastly many different ways. In some domains—such as human speech—the interesting classes may be defined by very specific and very subtle distinctions. No statistical technique can discover, in an arbitrary dataset, all and only the distinctions that we might consider important, because importance in this sense is not well-defined. What we can hope to achieve is a means of grouping sounds based on some domain general and statistically grounded measure of similarity, under the (reasonable) assumption that interesting groups tend to share gross physical characteristics. Hidden Markov models give us such a measure.

Cluster analysis based on HMMs can be carried out in a number of ways. Partitional clustering, termed K-models here, uses a static topology. Hierarchical clustering varies the number of clusters as they are learned, which can improve clustering performance and allow for model selection. Agglomeritive clustering merges clusters using a distance measure based on HMM posterior likelihood. It provides good performance, but is slow because it requires an HMM to be trained on each item in the dataset. Divi-

sive clustering works in the opposite direction, by splitting which have low average likelihoods under their existing models. This algorithm is more efficient, but provides sub-optimal clustering performance for complex data.

The HMM mixture architecture has two structural parameters that can be learned from data using Bayesian model selection. Selection of the number of Markov states in each HMM is intended to improve model fit, compared with the use of a static global value. Unfortunately, this can lead to unnecessarily complex models that are slow and prone to overfitting. Selection of the number of clusters can be done quite easily using the hierarchical algorithms. It seems, however, that in even moderately complex datasets, information theoretic criteria tend to promote finer-grained clusterings than those of human interest. The lack of an effective means of choosing the number of clusters is mitigated by the ability of the two hierarchical algorithms to produce a series of clusterings spanning an arbitrary range of granularities, allowing the user to choose the most natural or useful grouping in lieu of a fully automated selection.

The performance of these general methods of time series clustering on a particular acoustic dataset depends in large part on how the sound is preprocessed. The feature representation that gives optimal results depends heavily both on the characteristics of the dataset and nature of the classes by which performance is judged. For speech data, cepstral coefficients were found to give good clustering performance, and the inclusion of feature deltas brought a substantial gain. This corresponds with established results in supervised speech recognition. It is reasonable to suppose that the mel scale filtering widely used in these applications would further enhance clustering performance. The subjective whale song classes were discovered more reliably using decorrelated features generated by independent component analysis, and deltas were of little or no benefit. This suggests that the classification does not focus on the phonetic properties that figure prominently in human speech, but on more general acoustic characteristics. Thus, if the goal of a clustering task is to group sounds based on 'speech-like' properties, which inhere primarily in resonance and are invariant to pitch, cepstral analysis may be useful; for truly unknown data, where the clustering goal is simply to find the

most prominent distinctions of any sort, ICA is more appropriate. Deltas can be used depending on the extent to which the sounds' temporal characteristics are considered important.

The need to make judgements about which feature extraction method to apply and how many clusters to keep means that the acoustic clustering techniques explored here are not fully general. In applications where nothing is known about the data, these decisions will be made somewhat arbitrarily. In most clustering problems, however, there will be some prior intuition about what sorts of features are interesting. Once an appropriate preprocessing procedure has been applied, HMM-based clustering can generate a high quality partitioning of an acoustic dataset.

6.3 Future Work

6.3.1 Acoustic Clustering

The quality of model fitting, and consequently of overall clustering output, would be enhanced if model topology could be learned independently for each cluster. The BIC-driven search implemented here is highly inefficient, requiring the entire HMM initialisation and training process to be repeated for each candidate topology, and this search itself to be repeated each time the membership of a cluster changes. An efficient method could be developed for model topology optimization based on growing or shrinking existing models as needed rather than training new ones; it may be possible to base such a method on and existing technique known as successive state splitting [36], which has been applied to HMM topology design for speech recognition.

Selection of the number of clusters, as presented here, is also unsatisfactory. This is indicitive, not of any flaw in the BIC per se, but of the impossibility of any single model selection criterion satisfying arbitrary clustering goals. The BIC should therefore be extended to include a sensitivity parameter, allowing the relative weights of the likelihood and penalty terms to be adjusted in a principled way. It is possible to interpret the BIC in an informal way, by plotting it and choosing a point after a large jump

or before a relatively flat section, in preference to its true global peak. This flexibility of interpretation should be formalised.

A promising method for model selection is reversible jump Markov chain Monte Carlo (RJ-MCMC) [19]. MCMC refers to a family of statistical techniques for sampling from a complex distribution (one that can be evaluated but not sampled directly) by generating *proposals* from a simple distribution conditioned on the current state of the simulation. The proposal is accepted or rejected stochastically according to its likelihood in the target distribution relative to that of the current point: more likely points are always accepted, while less likely ones stand a chance of rejection. If the proposal is rejected, the current point is copied into the next iteration. The succession of accepted points constitutes a Markov chain, and given sufficiently many iterations, it will settle into an equilibrium distribution: the fraction of the total iterations spent in a given state converges to the posterior probability of that state. Reversible jump MCMC is an extension of this sampling technique to vectors of variable length. This allows HMM parameter sets to be treated as sample points, and the distribution density the data likelihood under those parameters. Provided the proposal distribution is capable of generating samples of larger or smaller dimensionality that are nonetheless valid HMM parameter sets, it is possible to find an equilibrium distribution that approaches, to arbitrary precision, the true likelihood over a range of model sizes. This stochastic approach to model selection has been applied to selection both of model topology and number of components in HMM mixtures [40]. However, it is not yet clear whether it will prove to be computationally tractable for real problems.

6.3.2 Whale Song

Using the methods discussed here, the whale song units can now be divided into groups in a . Given the limitations of the Bayesian model selection techniques applied here, there is still a need to determine the number of clusters that produces the most meaningful grouping; in the absence of more powerful statistical methods, this could be done simply by human analyisis of a learned cluster hierarchy. Once this has been done it should be possible to model the higher order structure of the songs. This could

also be done with Markov models; because the unit data has been converted to the form of symbolic cluster labels, there might be no need for a hidden variable. These models would have to be nested, with one level representing phrases as sequences of units, another combining phrases into longer themes, and possibly a third for the global song itself as a succession of these themes. While the overall song process is clearly non-Markovian, such a temporally nested approach might allow a very good Markovian approximation to be made. Alternatively, deterministic methods like sequence mining might be able to explicitly discover repeated song fragments.

Appendix A

HMM Equations

The standard method for computing the posterior probability of a sequence under an HMM, given in Rabiner's classic treatment [38] and elsewhere, is reproduced here for reference. It is part of the forward-backward procedure, although when dealing with complete sequences, only the forward calculation is required. The forward variable

$$\alpha_t(i) = p(x_1, x_2, ..., x_t, q_t = i|\lambda)$$

represents the probability of the latent variable taking on a given value at a given time, conditioned only on the observations thus far. It is possible to calculate α recursively:

$$\begin{aligned} \alpha_1(i) &= \pi(i)b_i(x_1) \\ \alpha_{t+1}(j) &= b_j(x_{t+1})\sum_{i=1}^{Q}\alpha_t(i)a_{ij} \end{aligned}$$

The overall sequence probability is just the sum of these probabilities for values of the final state:

$$p(X|\lambda) = \sum_{i=1}^{Q}\alpha_T(i)$$

Maximum likelihood parameter settings $\hat{\lambda} = argmax_\lambda p(X|\lambda)$ are learned using a set of iterative updates based on the EM algorithm, known collectively as the Baum-Welch equations (also given in [38]). First, some intermediate values are defined:

$$\beta_t(i) = p(x_{t+1}, x_{t+2}, ..., x_T|q_t = i, \lambda)$$

is the backward variable, quantifying the probability of all future observations given a current state value. The calculation mirrors that for α:

$$\begin{aligned}\beta_T(i) &= 1 \\ \beta_t(i) &= \sum_{j=1}^{Q} a_{ij} b_j(x_{t+1}) \beta_{t+1}(j)\end{aligned}$$

Next we define

$$\gamma_t(i) = p(q_t = i | X, \lambda),$$

the probability distribution of the latent variable at a given time, computed by way of the forward-backward terms:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{Q} \alpha_t(j)\beta_t(j)}$$

Finally, define

$$\xi_t(i,j) = p(q_t = i, q_{t+1} = j | X, \lambda),$$

the probability of a particular *transition* at a given time. The formula for this is:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(x_{t+1}) \beta_{t+1}(j)}{\sum_{k=1}^{Q} \sum_{m=1}^{Q} \alpha_t(k) a_{km} b_m(x_{t+1}) \beta_{t+1}(m)}$$

With these quantities available, the parameter updates are relatively straightforward to compute:

$$\begin{aligned}\tilde{\pi}_i &= \gamma_1(i) \\ \tilde{a}_{ij} &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}\end{aligned}$$

Where the output function B is Gaussian, as it is in the HMMs used in this dissertation, the state means and covariances are updated as follows:

$$\begin{aligned}\tilde{\boldsymbol{\mu}}_i &= \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot \boldsymbol{x}_t}{\sum_{t=1}^{T} \gamma_t(i)} \\ \tilde{\Sigma}_i &= \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot (\boldsymbol{x}_t - \boldsymbol{\mu}_i)(\boldsymbol{x}_t - \boldsymbol{\mu}_i)'}{\sum_{t=1}^{T} \gamma_t(i)}\end{aligned}$$

Bibliography

[1] Akaike, H. (1974). A New Look at the Statistical Model Identification, *IEEE Transactions on Automatic Control* **19**(6), 716-723.

[2] Banfield, J. D. and Rafferty, A. E. (1993). Model-Based Gaussian and Non-Gaussian Clustering, *Biometrics* **49**, 803-821.

[3] Biem, A., Ha, J.-Y., and Subrahmonia, J. (2002). A Bayesian Model Selection Criterion for HMM Topology Optimization, *ICASSP* (1), I989-992.

[4] Bilmes, J. A. (1998). *A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*. Technical report TR-97-021, Internation Computer Science Institute.

[5] Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

[6] Black, A. (2003). Lecture, University of Edinburgh, 17/14/2003.

[7] Bloomfield, P. (2000). *Fourier Analysis of Time Series: An Introduction* (second edition). John Wiley & Sons.

[8] Bottou, L., and Bengio, Y. (1995). Convergence Properties of the K-Means Algorithms, *Advances in Neural Information Processing Systems* **7**, 585-592, MIT press.

[9] Celeux, G., Chauveau, D., and Diebolt, J. (1995). *On Stochastic Versions of the EM Algorithm*. Technical report 2514, Institut National de Recherche en Informatique et en Automatique.

[10] Celeux, G. and Govaert, G. (1992). A Classification EM Algorithm for Clustering and Two Stochastic Versions. *Computational Statistics and Data Analysis* **14**(3), 315-332.

[11] Datta, S., and Sturtivant, C. (2002). Dolphin Whistle Classification for Determining Group Identities, *Signal Processing* **82**, 251-258.

[12] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum-Likelihood from Incomplete Data via the EM Algorithm, *Journal of the Royal Statistical Society* (series B), **39**(1), 1-38.

[13] Dom, B. E. (2001). *An Information-Theoretic External Cluster-Validity Measure*, IBM research report RJ 10219.

[14] Evans, P. G. H. (1987). *The Natural History of Whales and Dolphins*. Christopher Helm.

[15] Everitt, B. S. (1993). *Cluster Analysis* (third edition). Edward Arnold.

[16] Fraley, C. and Raftery, A. E. (1998). How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis, *Computer Journal* **41**(8), 578-588.

[17] Furui, S. (2001). *Digital Speech Processing, Synthesis, and Recognition* (second edition). Marcel Dekker.

[18] Gold, B. and Nelson, M. (2000). *Speech and Audio Signal Processing*. John Wiley & Sons.

[19] Green, P. (1995). Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination, *Biometrika* **82**(4), 711-732.

[20] Hansen, M. H. and Yu, B. (2001). Model Selection and the Principle of Minimum Description Length, *Journal of the American Statistical Association*, **96**(454), 746-774.

[21] Holmes, J. and Holmes, W. (2001). *Speech Synthesis and Recognition* (second edition). Taylor & Francis.

[22] Hyvärinen, A. (1999). Survey on Independent Component Analysis, *Neural Computing Surveys*, **2**, 94-128.

[23] Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data Clustering, *ACM Computing Surveys* **31**(3), 264-323.

[24] Jankowski, C. R., Hoang-Doan, H. V., and Lippmann, R. P. (1995). A Comparison of Signal Processing Front Ends for Automatic Word Recognition, *IEEE Transactions on Speech and Audio Processing* **3**(4), 286-293.

[25] Jelinek, F. (1999). *Statistical Methods for Speech Recognition*. MIT Press.

[26] Kamvar, S. D., Klein, D., and Manning, C. D. (2002). Interpreting and Extending Classical Agglomerative Clustering Algorithms Using a Model-Based Approach, *ICML*.

[27] Kass, R. E., and Raftery, A. E. (1995). Bayes Factors, *Journal of the American Statistical Association*, **90**(430), 773-795.

[28] Kaufman, L. and Rousseeuw, P. J. (1990). *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley & Sons.

[29] Krogh, A. et al. (1994). Hidden Markov Models in Computational Biology: Applications to Protein Modelling, *Journal of Molecular Biology* **235**, 1501-1531.

[30] Law, M. H. and Kwok, J. T. (2000). Rival Penalized Competitive Learning for Model Based Sequence Clustering, *International Conference on Pattern Recognition*, 195-198.

[31] Li, C. and Biswas, G. (2000). A Bayesian Approach to Temporal Data Clustering Using Hidden Markov Models, *ICML*.

[32] Li, C. and Biswas, G. (2002). Applying the Hidden Markov Model Methodology for Unsupervised Learning of Temporal Data, *International Journal of Knowledge-Based Intelligent Engineering Systems* **6**(3), 152-160.

[33] Mermelstein, P. (1980). Comparison of Parametric Representations for Monosyllabic Word Recogntion in Continuously Spoken Sentences, *IEEE Transactions on Acoustics, Speech, and Signal Processing* **28**(4), 357-366.

[34] Mitsakakis, N. (1996). *Classification of Humpbck Whalesong Units using a Self Organizing Feature Map* MSc dissertation, Department of AI, University of Edinburgh.

[35] Oates, T., Firoiu, L., and Cohen, P. R. (2000). Using Dynamic Time Warping to Bootstrap HMM-Based Clustering of Time Series, R. Sun and L. Giles, eds., *Sequence Learning: Paradigms, Algorithms, and Applications*. Springer-Verlag.

[36] Ostendorf, M. and Singer, H. (1997). HMM Topology Design Using Maximum Likelihood Successive State Splitting, *Computer Speech and Language* **11**, 17-41.

[37] Potamitis, I., Fakotakis, N., and Kokkinakis, G. (2000). Spectral and Cepstral Projection Bases Constructed by Independent Component Analysis, *ICSLP* **6**(3), 63-66.

[38] Rabiner, L. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *IEEE* **77**(2), 257-285.

[39] Rissanen, J. (1986). Stochastic Complexity and Modeling, *The Annals of Statistics* **14**(3), 1080-1100.

[40] Robert, C. P., Ryden, D., and Titterington, D. M. (2000). Bayesian Inference in Hidden Markov Models through the Reversible Jump Markov Chain Monte Carlo Method, *Journal of the Royal Statistical Society* (series B) **62**(1), 57-75.

[41] Savaresi, S. M. et al. (2002). Cluster Selection in Divisive Clustering Algorithms, *SIAM Internation Conference on Data Mining* **2**, 299-314.

[42] Schwarz, G. (1978). Estimating the Dimension of a Model, *Annals of Statistics* **6**(2), 461-464.

[43] Sin, C.-Y., and White, H. (1996). Information Criteria for Selecting Possibly Misspecified Parametric Models, *Journal of Econometrics* **71**, 207-225.

[44] Smyth, P. (1997). Clustering Sequences with Hidden Markov Models, *Advances in Neural Information Processing Systems* **9**, 648-654, MIT Press.

[45] Warin, T. (1997). *Automatic Inference of Humpback Whalesong Grammar*. MSc dissertation, Department of AI, University of Edinburgh.

[46] Welsh, N. (2000). *Approaches to Classifying Whale Song*. MSc dissertation, Department of Informatics, University of Edinburgh.

[47] Ypma, A. and Heskes, T. (2002). Categorization of Web Pages and User Clustering with Mixtures of Hidden Markov Models, *WEBKDD*, 31-43.

[48] Zhong, S. and Ghosh, J. (2002). *A Unified Framework for Model-based Clustering and Its Application to Clustering Time Sequences*. Technical report, University of Texas at Austin. Accessed at www.ece.utexas.edu/˜szhong/papers/modelbased.pdf, 03/08/2003.

[49] Zhong, S. (2003). *Probabilistic Model-based Clustering of Complex Data*. PhD dissertation, University of Texas at Austin. Accessed at www.ece.utexas.edu/˜szhong/thesis.pdf, 31/07/2003.