MILQ

by

Eric Brochu

BSc, University of Regina, 1998
BA, University of Regina, 1997

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

We accept this thesis as conforming
to the required standard

The University of British Columbia

February, 2004

Library Authorization

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Eric Brochu — Name of Author *(please print)*

10/2/2004 — Date (dd/mm/yyyy)

Title of Thesis: MILQ

Degree: Master of Science Year: 2004

Department of Computer Science
The University of British Columbia
Vancouver, BC Canada

Abstract

Computers cannot, of course, appreciate the emotional qualities of music. But can they describe music with emotional adjectives that match what a human might expect? I have implemented a system, MILQ, to explore this hypothesis.

Using a large data set, a selected set of labels (including both genre and style labels like INDIE ROCK and tone labels like CATHARTIC), and proven feature extraction techniques, I was able to construct a set of nonlinear logistic discriminative networks using Neural Network techniques, which computed marginal probabilities for each label. Such techniques and other Machine Learning methods have been used before to construct genre classifiers and my model works well for those.

Estimating the probabilities of the tonal labels is much more difficult, however, as these can have a very strong cultural component, as well as an acoustical one. Therefore, I add a second Bayesian network stage. This uses a set of labels from the logistic network as the priors for the belief of each label, treating the labels as nodes in a directed, loopy Bayesian network. Using a modified version of loopy belief propagation, the posterior of each label conditioned on its neighbours is computed to approximate the cultural component of the labellings by using the co-occurrence frequency of the labels as potential functions on the network. A number of evaluations and examples suggest that the model can be used with a fair degree of accuracy to assign tone-based adjectives to music.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

What MILQ *is.*

Can computers learn the emotional 'tone' of music? Of course, the answer depends on your definitions of things like *tone*, but I think they can, albeit in a limited way. In this thesis I present my current results in attacking the problem.

I set out to devise a system in which a novel song could be given to a computer and it would return a list of adjectives describing the emotional qualities of the music that would be reasonably similar to what a human would say. Give it Nirvana's 'Smells Like Teen Spirit', and we should get back ANGST-RIDDEN and WRY. Give it Sarah McLachlan's 'Possession' and we should get back POIGNANT and BITTERSWEET.

MILQ (Music Interpreted as Lexical Qualifiers) is a software system I have implemented to do just that. As shown in Figure 1.2 it uses a Bayesian network to assign probabilities to a set of labels representing different moods and styles.

For training and illustrative purposes, I have divided the network into two stages, each of which is a complete Bayes net. In the first stage the label probabilities are assigned using signal features extracted from the audio file. This is a fairly common process, often used for genre classification.

In the second stage, I make my main novel contribution – approximating the cultural component of the music. Clearly, properly applying an adjective like IRONIC to a song requires a greater appreciation of irony than our frustratingly literal-minded machines actually possess. However, the system

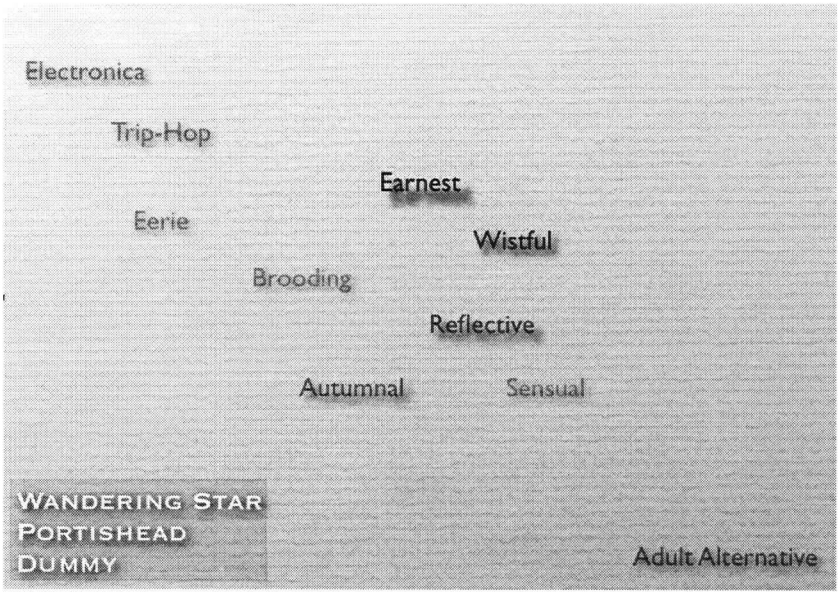

Figure 1.1: *Typical output from the* MILQ *system. The song being analysed is Portishead's 'Wandering Star', a test datum not used for training the model. The system has found the ten most probable labels, based on both the features of the audio and the patterns of occurrence of the labels in the training set. The placement of the labels is based on their co-occurrence in the training set, with frequently co-occurring labels placed closest together. The relative height (shown by drop-shadows) and darkness of the labels indicates the rankings of the label probabilities. A detailed discussion of the results can be found in Chapter 5.*

might be quite successful in determining from the audio qualities alone that a song is Indie Pop, and know from analysis of the labelling patterns in the training set that Indie Pop is very often ironic.

To approximate this, I introduce a second stage to the network, in which each label is based on the stage-one probability of the labels that have the highest correlation coefficients. So IRONIC might be a difficult concept for the model, but by combining the predictions for INDIE POP, HUMOROUS, SARCASTIC and SINGER-SONGWRITER, we can estimate the probability based on the estimates of a variety of labels.

overview

In the remainder of this thesis, I will discuss how I constructed this model and how I implemented it in the MILQ software system. The rest of the current chapter introduces a few more concepts and conventions. Chapter 2 gives a quick overview of previous research in this area, by myself and others. In Chapter 3, I discuss the data, feature extraction methods and labels I used for my experiments. Chapter 4 explains the Bayesian network model I designed for this project. In Chapter 5, I examine the problem of evaluating the system, and the results I managed to achieve with MILQ. Finally, in Chapter 6, I examine some of the applications of the technology and future areas of work.

1.1 Motivation

Computers and music have become increasingly intertwined. It has been argued that music has become the de facto 'killer app' for the Internet at the beginning of the twenty-first century. People are ripping their CD collections to compact MP3, WMA or AAC formats, creating and editing playlists on their computers, loading thousands of songs onto their iPods, burning mix CDs, listening to internet radio, sharing music over peer-to-peer services, and buying songs and albums online at Apple's iTunes Store and its imitators. XMMS, WinAmp or iTunes seems to be active on almost any computer that has someone seated in front of it. Modern computers are used to write, edit, record and perform music. Go to an electronica concert in 2004 and you will more than likely get to watch a professional musician click buttons on an Apple PowerBook. The fact that this revolution has been so sudden, so ubiquitous and yet drawn so little comment is a testament to its success.

Even more exciting is that the revolution opens vast new territory for scientific and artistic exploration. Thanks to CD ripping, it is easy in 2004 to create a centralized library of one's music collection whose size and fidelity would have been nearly unthinkable only ten years ago. In the same time period, computers have become approximately 100 times faster, and disk storage for all that data, and networks to move it around, have become hundreds of times cheaper. More complex algorithms can be run on bigger data sets, faster than ever before.

In particular, statistical learning approaches to music are ideally suited to this technology. Signal Processing and Machine Learning algorithms are often computationally expensive and benefit immensely from less expensive processing power. Detailed models can now be trained on massive data sets,

using audio signal features extracted in seconds.

1.2 milq

The task I set for myself was to come up with a system that would use Machine Learning to learn a model of mapping audio features to various mood-based labels – not only style and genre labels, like ROCK or AMBIENT TECHO, but also emotion-based adjective labels, like FUN or PASSIONATE. Once this was learned, new, unlabelled music could be given to the system, and labels could be assigned, along with values that would allow the labels to be ranked. So a previously unseen Cat Power song, for example, could be given as input to the system, and label values assigned that showed the song was CATHARTIC to a high degree, but not very MANIC.

MILQ is the implementation of this system. It uses a trained logistic discriminative net to map audio features to marginal label probabilities. Since many of the labels are not simply part of the audio, but also have cultural components, the model also takes into account the relationships between the labels themselves. The intuition is to leverage the more easily-predicted labels into the more difficult ones. So if the audio features alone predict that our Cat Power song is GLOOMY, SINGER-SONGWRITER, and not MANIC, these will influence the probability assigned to the CATHARTIC label, even when analysis of the audio alone was unable to successfully determine whether or not the song was CATHARTIC.

1.3 Symbols and notation

The symbols I have used in this thesis are collected in Table 1.3.

When speaking of particular labels as such, they will appear in a capitalized font. The names of albums are italicized and song titles appear in quotation marks. So 'Cemetery Polka' is a song on *Rain Dogs*, by Tom Waits; it is labelled BLEAK, THEATRICAL, QUIRKY, SINGER/SONGWRITER and ROCK. A full list of labels appears in Appendix A.

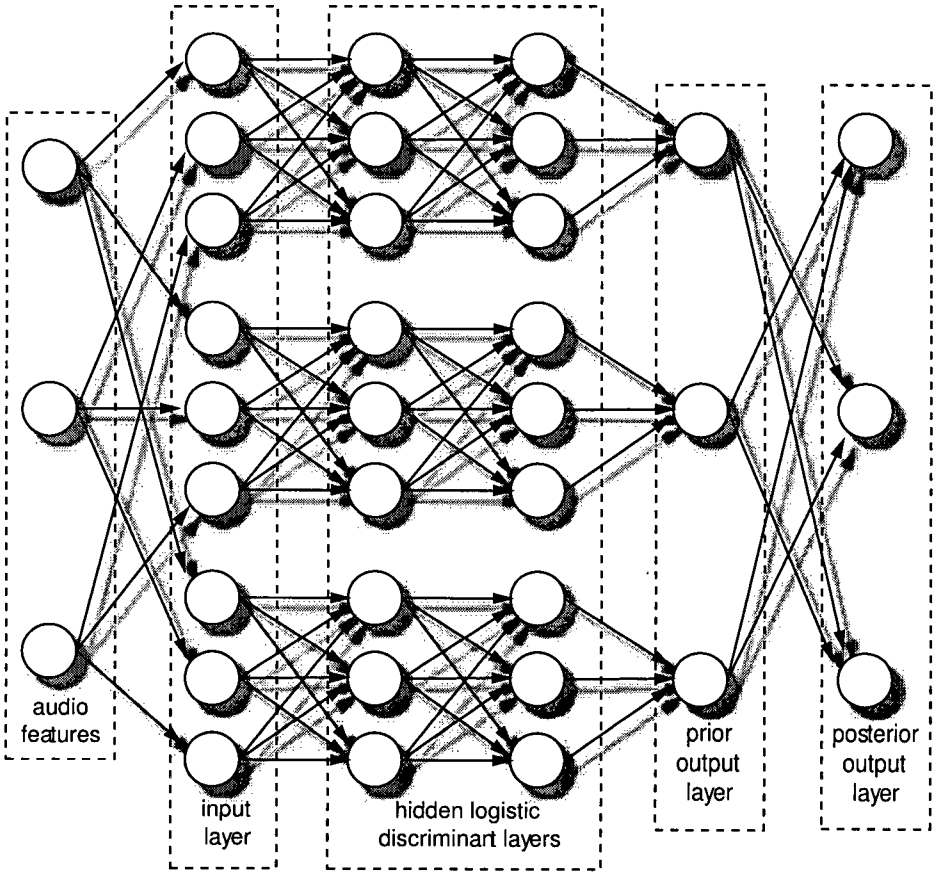

Figure 1.2: *A simplified version of the* MILQ *Bayesian network. The audio features are extracted, and the principal components are sent to the network input layer. The hidden discriminant layers map the input nonlinearly to prior label output probabilities. Finally, the posterior label probabilities are found for each label using a subset of the label prior outputs. The transfer functions of all layers are learned from training data of several thousand labelled songs. The model is discussed in detail in Chapter 4.*

μ	mean
σ	variance
D	data matrix, where each row, x, corresponds to a datum, and each column, a, to a feature
a	feature index
x	datum index
n_a	number or features
n_x	number of data
F	the F-measure, a combination of precision and recall
s	the sum of weighted inputs into a neuron
l	label index
Z_l	classification indicator of label l
L_l	the frequency of label l in the database
β	Dirichlet hyperprior for modelling uncertainty about label frequencies
θ	the set of all parameters of a Neural Network

Figure 1.3: *Symbols used in this thesis.*

Chapter 2

Precedence and Prescience

Where MILQ *comes from.*

The motivation for this thesis originates in work I did in 2002, previously published in [Brochu and de Freitas, 2003; Brochu *et al.*, 2003], though MILQ does not follow directly from that research, *per se.*

In those papers, my collaborators and I described our implementation of a mixed-media database and search engine. [Brochu and de Freitas, 2003] presented a database of musical scores for (mostly) popular songs in GUIDO notation.[1] Each score was associated with a text file containing the song's lyrics. [Brochu *et al.*, 2003] extended the first paper by adding images to the database and On-line EM [Bao, 2003] to the learning model. Those papers were motivated in part by work on Bayesian modelling for multimedia databases published in [de Freitas *et al.*, 2003].

While working on those projects, I was struck by both the potential of mixed-media models and the limitations of my approach. In particular, by representing music in a form derived from musical notation, I was limited to works for which I had the musical notation for. Further, even building the database was a laborious process, involving online searches for MIDI files which could be parsed into GUIDO format, and more searches for the song lyrics. As a result, the data sets I used were fairly small, consisting of around 100 documents – enough to establish the validity of my model and approach, but not enough to build a powerful training set for statistical learning.

[1] GUIDO [Hoos *et al.*, 2001a; 2001b] is a means of representing music notation in an XML-like format.

After working on that project, I decided I wanted a project that would lend itself to recorded music in digital audio file format (such as MP3), which is commonplace and could very easily be extracted and collected. For my personal satisfaction, I also wanted to move away from straightforward Information Retrieval tasks and into something more unique and original.

Moving to digital audio also allowed me to treat the representation of music as a signal processing issue rather than a musicological one. In one sense, this was more a sideways step than a forward one, as I am no more an electrical engineer than I am a musicologist, but it did allow me to base my feature extraction on the large body of literature of Signal Processing, rather than the smaller world of Statistical Musicology.[2]

2.1 Related work

There is a fair body of literature on the use of statistical audio signal processing for music classification. Most of the work is limited, however, to determination of genre.

[Golub, 2000] uses a set of statistical features extracted from audio files to do genre classification. I use his feature extraction methods as part of the feature set I extract for my own classification (Section 3.2.2).

Foote and Cooper and their collaborators have done a great deal of work with Information Retrieval applications based on audio Signal Processing. [Foote and Uchihashi, 2001] and [Foote and Cooper, 2001] present *beat spectra* and *beat spectrograms*, which are audio 'signatures' based on the self-similarity of a signal over time, which shows the placement of regular beats. In [Foote *et al.*, 2002], the authors test similarity measures between different beat spectra, using Euclidean distance and cosine similarity in the spectral feature space. This is somewhat similar to [Pampalk, 2001], which I also incorporate into my feature extraction (Section 3.2.1).

There are a number of other researchers who have worked on the problem of finding suitable feature sets which could be extracted from audio files for classification or retrieval using spectral methods based on FFT, cepstral or mel-cepstral coefficients. A summary of the state of the art as of 2002 can be found in [Pachet, 2003; Aucouturier and Pachet, 2003]. As the subject is not directly relevant to my own work, I do not wish to dwell on the individual contributions here, though I will discuss in some detail the methods I actually use in Sections 3.2.1 and 3.2.2.

[2]This seems to be changing. [Beran, 2004] is a promising overview of recent developments in Statistical Musicology.

The CUIDADO Project[3] [Vinet *et al.*, 2002] includes a Music Browser, which exploits metadata to estimate the similarity of songs based on co-occurrence. The metadatabase is composed of playlists and web sites. When songs co-occur in the same metadata document, their similarity measure in increased. This is combined with more traditional descriptors extracted from the audio signal to find an overall similarity. The user can then set various properties to generate a random playlist of music from the database.

[Platt *et al.*, 2002] also uses metadata to find similarity scores between songs. The authors introduce a system called AutoDJ, which uses co-occurrence in playlists and albums to find similar songs. Users can give one or a few 'seed' songs as training examples, and the system uses the metadata and seeds to find a user preference function over the songs to generate a playlist similar to the seed songs.

MoodLogic[4] is a popular and intriguing program that relies on a huge network of users to provide metadata on songs, which is stored and processed on central servers. Unfortunately, the techniques they use are proprietary and I have been unable to review them.

Whitman *et al*

The work that I have found that most closely resembles my own is that of Brian Whitman and his collaborators at MIT. In [Whitman and Smaragdis, 2002], the authors present a musical style classifier that combines audio signal features with text features. The audio features are used to train a multiclass classifier. Similarity between artists is computed from the number of shared terms, and is used to cluster the artists together. The results section of the paper demonstrates that classification accuracy improves significantly when the models are combined, though the number of data is fairly small: 5 styles, each with 5 artists.

[Whitman and Rifkin, 2002] builds on [Whitman and Smaragdis, 2002] by treating the problem not as multimodal classification, but by using the text terms as the labels and training a classifier using the audio features as inputs. Most recently, in [Whitman *et al.*, 2003], the authors use Regularized Least-Squares Regression [Rifkin, 2002] to learn a mapping from extracted audio features to a set of text terms automatically extracted from the web.

Both the techniques and goals are different from mine. Even in [Whitman *et al.*, 2003], the work most similar to mine, the authors use an unsupervised model of the language feature collection to discover semantic pa-

[3] `http://www.cuidado.mu`

[4] `http://moodlogic.com`

rameter spaces for the music – for instance automatically learning from the appearance of the terms 'loud' and 'soft' (and assisted by WordNet [Miller, 1990]), that 'soft' to 'loud' is a continuum on which songs can be placed, and a model for mapping audio features to that parameter is learned. The authors' model for incorporating cultural components of the terms is thus quite different from mine, and, of course, the learning algorithm is quite different (Regularized Least Squares Regression is a kernel method similar to Support Vector Machines, whereas I use discriminative Bayesian networks). Nevertheless, it is certainly a similar domain to my work. The fact that the authors get such good results in their problem space is encouraging.

Chapter 3

Data and Preprocessing

What MILQ *is made of.*

As with many Machine Learning applications, the selection of data sets and labels plays a somewhat ambiguous role. When developing a model, even for a specific type of application, it is desirable that the model be agnostic in regards to the data. At the same time, models are very often affected by properties of the data [LaLoudouana and Tarare, 2002].

In this chapter, I discuss the data that I used, and the features and labels I applied. While the data and the problem I chose to solve influences the structure of the model in Chapter 4, I also wanted to leave the system as open to change as was possible without impairing the quality of the results on the data set I *did* use. I feel that to a large degree, the data or labels could be changed, or the features extracted could be replaced with other features, and that the model would still function as well as the data allowed. Nevertheless, it is important to understand the data features and labels used to understand the model evaluation in Chapter 5.

The data I used for my model training and experiments consists of MP3 audio files ripped from a large collection of CDs, along with labels for each song, which are extracted from the Internet (Section 3.1). It is not practical or even desirable to use an entire binary audio file as an input datum, so I extract a set of features from each audio file and represent the song as a feature vector. The methods for doing this are presented in Sections 3.2 and 3.3.

The labels, similarly, are extracted from larger text files associated with

each album. In Section 3.4, I discuss how I decided on a set of labels and where I got them from.

3.1 The data

The audio data set I use consists of 8556 MP3 files, extracted directly from 714 albums by 315 different artists (or by 'Various Artists'). These are selected from a larger data set (around 13000 MP3s), from which have been removed albums for which no labelling information could be extracted (Section 3.4), or which had genre labels other than ROCK or ELECTRONICA.

In all cases, the albums are complete, with all tracks present. Most of the albums are full-length albums, but a few are EPs.

The two main genres represented in the database are rock/pop and electronica. There were a small number of albums from other genres, such as jazz, blues, rap, hip-hop, country, classical and folk. Since music of different genre usually sounds very different, including the 10% or so of the library that are not rock or electronica comes dangerously close to introducing noise. To avoid this, I removed all songs that were not labelled either ROCK or ELECTRONICA, leaving 8556 songs.

3.2 The features

Because I elected to limit the scope of my contribution to my Machine Learning work, and because signal processing is a very challenging topic in its own right, the audio feature extraction techniques I use are based entirely on work already done in that field.

The audio feature extraction methods I use are based on [Golub, 2000] and [Pampalk, 2001], two theses on the topic of extracting features from audio files for the purposes of classification and browsing. These works present two different methods of extracting features from audio files for Machine Learning purposes. The reader concerned with the details of implementation should consult the individual works, but in the following sections, I will try to present an overview that sufficiently justifies their use.

It is also important to note, however, that any feature extraction method could be used, as long as it maps an MP3 file to a feature vector $X \in \mathbb{R}^n$ for some constant integer n. Other audio signal feature extraction techniques that are used for classifying or finding distances between audio files may be found in, for example, [Wold *et al.*, 1996; Tzanetakis *et al.*, 2001; Foote *et al.*, 2002].

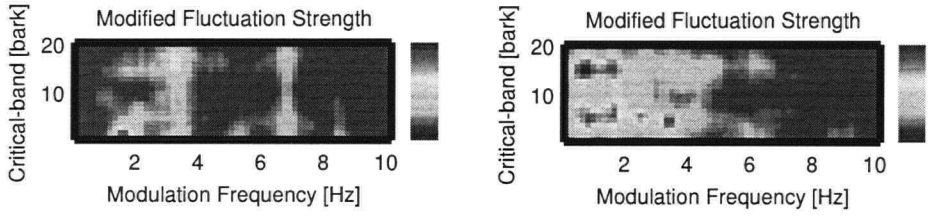

Figure 3.1: *Smoothed fluctuation strength matrix for six-second windows of two songs, taken from* [Pampalk, 2001]. *Robbie Williams' 'Rock DJ' is shown on the left, The Beatles' 'Yesterday' on the right. The intensity of the entry in the n-by-m matrix corresponds to the fluctuation strength of the signal in critical band n for frequency m. A bright pixel indicates that there is a strong repeating rhythm with frequency m in critical band n for that sample.*

3.2.1 Pampalk's psychoacoustic features

In his thesis, Pampalk [2001] presents a means by which audio features can be extracted in such a way that a topography of audio files can be laid out, with similar music placed together in 'islands' and 'continents' using Self-Organizing Maps.

feature extraction

Pampalk's feature extraction work is heavily informed by psychoacoustics [Zwicker and Fastl, 1999], the study of the relationship between physical sounds and the brain's interpretation of them, and mostly operate by extracting beat characteristics.

MP3 files are down-sampled to 5.5 kHz and transformed to *Pulse Code Modulation* (PCM) representation, a discrete approximation of the continuous acoustical wave (virtually any audio player will convert from MP3 to PCM). The downsampling is justified by the observation that higher acoustic frequencies contribute very little to human identification of particular pieces of music.

Actual analysis is performed on *loudness*, the intensity sensation. Loudness is measured by comparing a sound to a reference: a 1 kHz tone at 40dB, called a *sone*. A sound perceived to be four times as loud as the reference tone has a value of 4 sone, for example.

To determine the loudness values of subsamples of the data, the signal is transformed from the time domain to the frequency domain, using *Fourier Transformations*. The frequencies are then bundled into 20 *critical bands*.

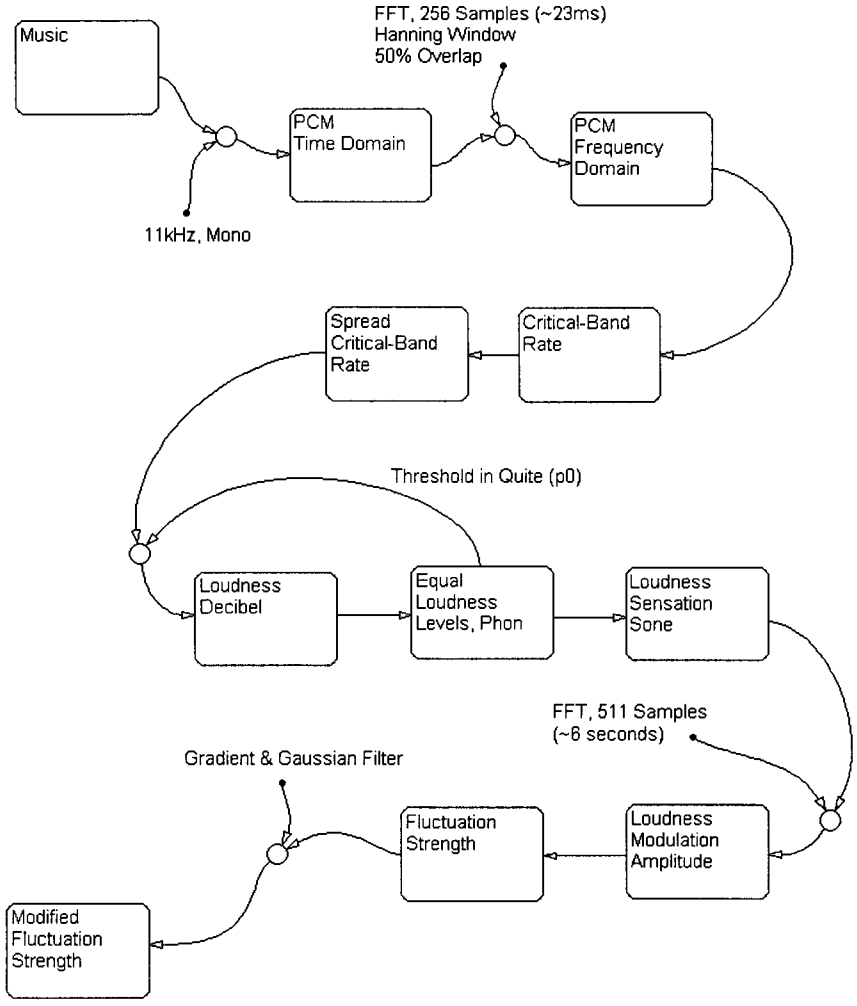

Figure 3.2: *Feature extraction procedure from* [Pampalk, 2001].

Critical bands are another psychoacoustic tool following from the observation that frequencies within particular frequency bands cannot easily be distinguished by humans. This is because the inner ear separates frequencies and concentrates them at certain location along the basilar membrane. The inner ear, in fact, acts as a series of band-pass filters. Different frequencies can therefore be 'bundled' into critical bands, known in this context as *barks*. The frequency values of the critical bands have been determined experimentally and so there exists a known (nonlinear) mapping from frequency to bark.

The power level (in dB) at each critical band can then be determined, and another nonlinear mapping transforms the critical band power levels to sone.

The end result is a 20-dimensional vector of the perceived intensity at each critical band, over a 23ms interval. The process is repeated over contiguous non-overlapping windows. Using the discrete Fourier transformation again, a time-invariant *loudness fluctuation strength* is found, which shows the loudness fluctuation (which corresponds closely to what we perceive as rhythm) at 1 Hz intervals from 1Hz to 30 Hz. The result is a 20-by-30 matrix, which essentially acts as a time-invariant 'beat signature' representation of the file. The matrix can be transformed to a point in Euclidean space by partially smoothing the 'peaks' (regions of the matrix with high values, surrounded by similar values in both nearby rows and nearby columns) in a Gaussian fashion to suppress the exact location of said peaks and then 'unrolling' the matrix into an 600-dimensional vector.

milq

In MILQ, the 600-dimensional vectors make up the first 600 features in the feature matrix D. The code for the Islands of Music feature extraction is available from `http://www.ai.univie.ac.at/~elias/music/code.html`.

I elected to use this feature extraction method for a number of reasons. Aside from practical concerns such as the availability of the code and the fact that it results in a constant-length vector for each MP3 file in $\mathbb{R}^n$, it captures information in an intuitive way. By building so strongly on psychoacoustics, it results in a principled method of determining which features to extract. By capturing information about beats, it serves as a useful complement to the other features I use, which do not capture this information. And by being designed for a Machine Learning application that depends on finding relative positions between feature vectors, it has already been tested in the general problem domain and found suitable.

3.2.2 Golub's long-term statistics

Golub's feature extraction work [2000] serves as a useful complement to Pampalk's. It is intended for genre classification using various nonlinear, non-probabilistic classifiers.

feature extraction

Various features of the signal are extracted to compute short-term features of a series of contiguous non-overlapping frames. These features are in turn used to compute the long-term features of the entire song, which make up the final feature vector of the song.

short-term features

The short-term features are extracted from 30ms frames.

- The normalized $\log_2$-amplitude of the signal is computed. The features collected from his data give us an indication of the dynamic range of the song, from loudest to softest.
- The centroid is the energy-weighted mean of the log of the frequencies. It is extracted to give a sense of the frequency range of the song.
- The bandwidth is computed as the energy-weighted standard deviation of the log of the frequencies to give a sense of the frequency range of the signal.
- As an approximation of the harmonicity of polyphonic music, the uniformity of energy levels in frequency bands is also extracted.
- The *first difference* (simply the difference in value between frames) is also computed for the latter three features.

long-term features

44 of the 46 long-term features are found by computing the means and standard deviations of the short-term features over 4-second windows, and then by computing the means, standard deviations and weighted means (giving more significance to louder frames) over those windows to extract the feature qualities of the entire song.

The last two long-term features are simply the length of the song (in seconds) and a *loudness scale factor*, the maximum mean loudness over the

4-second aggregate windows, representing an overall impression of how the loudness of the song might be perceived by the listener.

milq

The 46 long-term features make up the last 46 in the 646-dimensional feature vector used in MILQ. The code for his feature extraction is published on Seth Golub's web site, `http://www.aigeek.com/aimsc`.

These features serve as a valuable complement to the ones in Section 3.2.1, as there is very little overlap in the attributes they are extracting.

3.3 The projection

Sections 3.2.1 and 3.2.2 describe two methods of feature extraction, but they are far from the only methods possible. MILQ is intended to be used on any extracted feature vector D_x that exists in $\mathbb{R}^f$.

This opens a few problems, however. The features selected probably exist on different scales. f is likely to be quite large. Many of the features used may be highly correlated and can thus be combined into smaller numbers of features, while others may simply be noise. Fortunately, there are standard methods of dealing with these problems.

3.3.1 Scaling

Let $D^{(unscaled)}$ be the unscaled n_x-by-n_f data matrix, where n_x is the number of data and n_f the number of features. Let D_x be an arbitrary row (datum) of the matrix – the features extracted from a particular song.

I assume that the individual features f of $D^{(unscaled)}$ are normally distributed. To make comparison between features fair, then, I scale them to a Standard Normal distribution, of zero mean and unit variance: $D^{(scaled)}_{x,f} \sim N(0,1)$. Letting the mean of D_f be μ_f and the variance be σ_f, then we simply compute

$$D^{(scaled)}_{x,f} = \frac{D^{(unscaled)}_{x,f} - \mu_f}{\sigma_f^{-1}}$$

3.3.2 Principal component analysis

Principal Component Analysis (PCA) is a means by which we can project a high-dimension space to a lower-dimensional one. It is especially useful

where the dimensions are highly correlated, as they are in the audio features I extract.

PCA is a linear projection from a f-dimensional feature space to an a-dimensional eigenspace, which is guaranteed to minimize the reconstruction error. Since PCA tells us the variance accounted for by the individual eigenvectors of the eigenspace, we can simply choose the amount of variance to account for and select a set of a eigenvectors that does just that. The data can simply be multiplied by the a-by-f matrix of those eigenvectors to project it into the orthonormal a-dimensional eigenspace. So if A is the a-by-f matrix whose rows are the a eigenvectors of $D^{(scaled)}$ with the highest eigenvalues, then

$$D^{(PCA)} = AD^{(scaled)}$$

This results in a projection of the original x-by-f data matrix $D^{(unscaled)}$ to an x-by-a data matrix, $D^{(PCA)}$.

I ran PCA on the data set and took the first 66 principal components, which accounted for 99% of the variance of the data. This allowed me to project from a 646-dimensional space to a much more manageable 66-dimensional one. In the rest of this thesis, D refers to the data matrix $D^{(PCA)}$ that has been projected into this 66-dimensional eigenspace.

3.4 The labels

The label-selection task for this model was come up with a set of labels that could be applied to a training set of audio data. Supervised learning algorithms could then discover the relationships between the labels and the audio features. Labels could then automatically be applied to unlabelled music.

Early on, I made a number of decisions regarding the labels I would train the data set on.

- The labels would have to be extracted automatically on the web. Manually annotating 800 albums, or worse, 10000 songs, was too odious a task for me to even consider, given my time commitments and resource constraints.
- The labels would be *extracted* on a per-album level, rather than a per-artist or per-song level. While some artists are very consistent in their tone and style, others can vary dramatically over the course of their careers. David Bowie, for example, is impossible to pin down.

Further, there simply isn't enough metadata available on individual songs to appropriately populate a metadatabase. Most albums, however, tend to be reasonably homogeneous sounding, and most reviews and opinions available on the web are about albums.

- The labels would nevertheless be *applied* on a per-song level. Since the ultimate goal is to label individual songs, the labels in the training set must be made up of individual songs as well. I make the assumption that the songs on any given album are intended to be listened to together to create a certain mood, and that it is this mood that the album labels apply to. It is not, of course guaranteed that each song can really be seen as having the same overall mood of the album just because it contributes to the mood. However, it seems like a reasonable simplifying assumption to make.

3.4.1 Label extraction

My first inclination had been to find labels by querying the web using the names of albums or bands, on a search engine such as Google. While I think this can be a valid strategy, and it was successfully used in [Whitman *et al.*, 2003], it failed to meet my needs for several reasons:

- There is enormous variation in the amount of information available on different artists and musicians. A Google search on `+flim +helio` turns up 310 pages, while `+radiohead +"kid a"` finds 76700 and `+madonna +"american life"` returns 131000. This would suggest results extracted individually for each album would be more accurate for *Kid A* and *American Life* than *Helio*. While it could be argued that this bias properly represents the fact that the most popular music *should* have the strongest signal in the learning arena, I felt this was contrary to my goals.

- Collecting exhaustive statistics from online search engines is often explicitly prohibited by the terms of service. For example, Google has a published API[1], but is restricted to 1000 queries per user per day. Google's Terms of Service[2] explicitly prohibits automated querying that does not originate from the API, even to the extent of explicitly prohibiting noncommercial research purposes.

[1] `http://www.google.com/apis`
[2] `http://www.google.com/terms_of_service.html`

- The names of albums and musicians often make for poor queries. A Google search for `+madonna +music` turns up about 2.6 million pages – needless to say, many more of those are about Madonna's music than Madonna's *Music*. Similarly, searches for *Poem* by Delerium, *1* by Pole, *Infected* by The The and *Help* by The Beatles need to be handled in a fairly sophisticated way to avoid the useful data being drowned out by noise.

- I wanted to manually select a set of features that would both be consistent across different artists and genres, and that satisfied my own goals of complementing learning on genre and style with mood-based adjectives. I didn't want the set of labels to be automatically uncovered by data mining.

the all music guide

As a result, I chose to use the All Music Guide, a very thorough online music database.[3] For each album in the database, human experts have written reviews, added genre, style and tone keywords, suggested similar albums and provided other pertinent data.

The database submissions are by freelance music critics, and overseen by an editorial staff. This makes it attractive, as the information in the database has been vetted by human beings with some level of expertise, which should help keep down the noise that occurs by unfiltered web searching. Furthermore, suitable allowance is made under the terms of service for using the data in a noncommercial venue.

I downloaded and parsed the All Music Guide web pages for the albums in my data set. The 'Genre', 'Style' and 'Tone' entries became the labels for the MP3s in the album. The 'Artist' and 'Album' fields are extracted so that I can confirm that the correct page was downloaded, and other data is extracted, but is not currently used in MILQ.

The system is far from perfect, of course. Many of the more obscure albums have fields missing, and the labelling will pick up the biases of individual critics, particularly in the 'Tone' labels. Some effort seems to have been made to standardize the list of allowable tones, but deciding whether to label a particular album PLAINTIVE or YEARNING, or whether AGGRESSIVE, ANGRY or HARSH is the most appropriate tone is naturally going to be biased by individual preferences.

[3] `http://www.allmusic.com`

Even so, I have elected to use All Music as the 'ground truth' labelling, viewing the biases in the labellings as an acceptable price for bypassing the noise of searching the entire web.

3.4.2 Label selection

There are 470 labels that appear in the database, but many of them occur very infrequently, and hence would make poor candidates for training. I therefore limited my work to 100 of the most common labels. These are listed in Appendix A.

There are 2 genres labels: ROCK and ELECTRONICA. There are 24 style labels, such as INDIE ROCK and TRIP-HOP. The styles can be seen as descending in a strict hierarchy from the genres: no style occurs in conjunction with more than one genre label anywhere in the database, and every style occurs with a genre in at least one datum. There are then 74 tones, from ACERBIC to WRY. Co-occurrence of individual tones is in no way restricted. HYPNOTIC occurs most often with ELECTRONICA labels like TRIP-HOP and AMBIENT TECHNO, but also occurs with ROCK labels like POST-ROCK and PROG-ROCK.

While I am not as interested in predicting the genres and styles of music as I am in the tones, these are valuable to the model. One would expect the genres and styles to be easier to predict from audio features and that fact can be exploited to improve predictions of the tones. Style and genre labels are also less ambiguous and may end up being easier for human beings to interpret. This is discussed in Chapter 4.

Chapter 4

The Model

How MILQ *learns.*

Classification is at the heart of MILQ. Constructing and testing various classifiers and variations of them consumed the majority of my development time.

In order for MILQ to function as a viable and interesting application, the classification system must work well. It is not clear, however, just what 'well' might mean. My initial intuition on this problem was simply that I wanted the classifier to maximize the number of correct labellings. I soon realized that this was not what I actually wanted. What I am actually seeking is classification that closely matches what the user might expect, and what the user expects is a bit more subtle.

The model has to take into account the fact that a *dramatically* wrong classification is much worse than *slightly* wrong classification. Misclassifying the notoriously nihilistic album *The Downward Spiral* by Nine Inch Nails as SUMMERY is not only wrong, it is so wrong that it erodes user confidence in the system. But misclassifying the same album as GLOOMY is not nearly as severe a problem – it's still an error, as it turns out the label isn't actually part of the ground truth in the training set – but it doesn't seem out of place.

Discussion of the actual Bayesian network classifier is therefore broken into two stages: a discriminative logistic network stage, which determines the probability of each classification using only the audio features; and a second stage which uses the marginal outputs of the first stage, as well as the

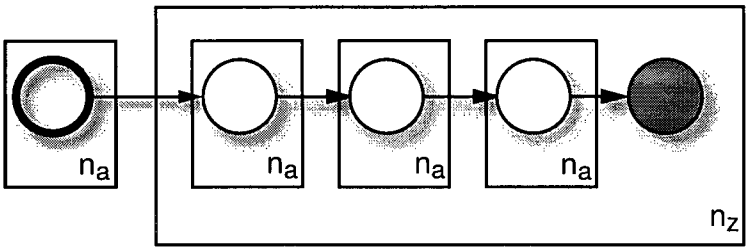

Figure 4.1: *Plate diagram of the prior network (the first stage of the complete Bayesian network). There are n_a input dimensions (empty circle). Each influences three 'hidden' logistic discriminant layers of n_a nodes each (white circles). Each node in the hidden layers has every node in the previous as a parent and acts as a parent to every node in the next layer. The nodes of the last hidden layer are the parents of the output node (grey circle). The entire hidden layer and output structure is repeated for each of the n_z labels. This is equivalent to having a feed-forward three-layer Perceptron network for each label.*

label co-occurrence to generate a final label probability given all sources of information. Since under this two-stage model, the role of the first network is to generate marginals which act as priors for the second network, I will refer to the first network as the prior network and the second as the posterior network.

Structurally, however, these are components of a single Bayesian network (Figure 1.2). Breaking the network into two stages like this is done for illustrative and parameter training purposes only, and in the final MILQ application, there is no separation between the networks.

As we shall see, the prior network (Section 4.1) uses well-understood principals. It is in the posterior network (Section 4.2), which transforms the priors to a consistent set of posterior label probabilities, that I feel my main contribution lies.

4.1 The prior network

The prior network is a discriminative logistic Bayes net, shown in Figure 4.1. This is implemented so as to be equivalent to a multi-layer perceptron Neural Network. For simplicity, I use Neural Network terminology in detailing this portion of the overall Bayesian network, but it is important to note that Neural Networks are simply a type of Bayes net [Jordan, 1995]. In fact, this compatibility is essential to my extensions to the network topology (Section

4.2).

I anticipate the reader is familiar with Neural Networks, and that I do not have to detail them here. The interested reader should consult [Bishop, 1995] for a detailed overview.

The decision to use Neural Networks (NNs) instead of other methods was not an easy one, and was largely motivated by the requirements I had, and the unsuitability of of other schema.

Most significantly, I needed a classifier whose outputs could be interpreted in a ranked order. This immediately caused me to reject popular classifiers such as Support Vector Machines [Vapnik, 1998], and k-Nearest Neighbour classification, which return simple binary-valued labels.

On the other hand, the training set is stable, which ameliorates one of the most common reasons for rejecting Neural Networks: the fact that for optimal performance it must be tailored to the data set being used. Given that my requirements are that training only be done once, but done as well as possible, the extra overhead in tailoring the network was an expense I was willing to pay.

Furthermore, the easy availability of powerful and flexible industrial-strength NN design software was another strong feature in favour of using NNs. I used MATLAB v6.5 and the MATLAB Neural Networks toolbox v4.0.1 for the NN components of my research.

4.1.1 Network design

There are a lot of decisions to be made in designing a NN. The number of layers, and neurons per layer must be decided. Transfer functions must be determined. The type of network must be decided on. The parameter optimization function must be selected. And so on.

Some of these follow easily from the data itself and the problem being solved. Others are best found through empirical evaluation.

network requirements

While a NN could be constructed so that a single network handles training for all outputs, I elected to instead construct a separate NN for each label. On modern computers, the computational cost of doing this is not excessive, even with my set of 100 labels and over 8000 training data. Creating numerous independent small networks allows much more flexibility: labels could be added or removed easily; networks could be tailored on a per-label

level, and so on. The advantages of a single network are savings in speed and storage, neither of which was of major concern to me.

I also required that the output could be interpreted as a marginal probability, so that it could act as a prior observation for a second network stage (Section 4.2). So rather than a hard $-1, +1$ classification, it was necessary that the network generate a probability of membership in the class.

validation

I used four-way cross-validation to evaluate each network. That is, for each NN I trained, the data set was divided into four similarly-sized, mutually-exclusive subsets, and the network was trained four times, each using three of the subsets as the training set. The held-out data was used as the test set. In this way, each data was used in training three times and in testing once. The results were composed of the outputs of all the test data, from all four trials.

Because the outputs were probabilistic, and because ordering by probability provides an optimal ranking [van Rijsbergen, 1979], I could rank the test set and evaluate the results using standard Information Retrieval techniques (my source here is [Manning and Schütze, 1999]). A document is said to be returned when its labelling is among the highest n ranked values of Z, where n is determined by some attribute of the data, such as a threshold for $p(Z|D_x, \theta)$, or fraction of true positive labellings in the training set.

Precision is the ratio of the correct labels in the results returned to the total number of labels returned. *Recall* is the ratio of the correct labels returned to the total number of labels in the database. So if we let tp be true positives, fp false positives, and fn false negatives, then for recall R and precision P

$$\begin{aligned} P &= \frac{tp}{tp + fp} \\ R &= \frac{tp}{tp + fn} \end{aligned}$$

Clearly, there is a trade-off between precision and recall. You can get perfect recall be returning all the documents in the data set, and you can optimize the expected precision by returning only the highest-ranked document. So a means of measuring performance jointly is necessary. A common

measure combining precision and recall is the F-measure

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}}$$

where α is a value between 0 and 1 which controls the weighting of the importance of precision and recall. Arbitrarily, I set $\alpha = 0.5$ which reduces the F-measure to

$$F = \frac{2PR}{P+R}$$

4.1.2 Experimentation

I examined and evaluated several different NNs – *softmax* activation networks [Bridle, 1990], *probabilistic neural networks* [Wasserman, 1993], single-layer logsig perceptrons – but by far the greatest success I had was with multi-layer logsig perceptrons.

multilayer perceptron with logsig

The logistic sigmoid – or *logsig* – transfer function has proven very popular in the Neural Network community. The logsig function was originally motivated in the single-layer perceptron by the goal of ensuring that outputs represent posterior probabilities. The assumption is made that class-conditional densities can be approximated with normal distributions. This assumption, and the logsig function, have since been extended to the multi-layer NN [Rumelhard *et al.*, 1995; Bishop, 1995].

The logsig function is defined on s, the sum of the weighted inputs, as

$$\text{logsig}(s) = \frac{1}{1+\exp(-s)}$$

and goes from 0 to 1 as s goes from $-\infty$ to ∞ (Figure 4.2).

I experimented with several different logsig network configurations. My initial experiments were with a single-layer network, which performed very poorly, suggesting the classification was highly nonlinear.

I then tried three different multi-layer network configurations of increasing complexity: n_a-5-1, n_a-n_a-1 and n_a-n_a-n_a-1 (recall that n_a is the number of dimensions in the feature vector, 66 in the case of MILQ). While I tried a number of learning algorithms, I found the *scaled conjugate gradient* algorithm [Moller, 1993] to offer the best performance, without being

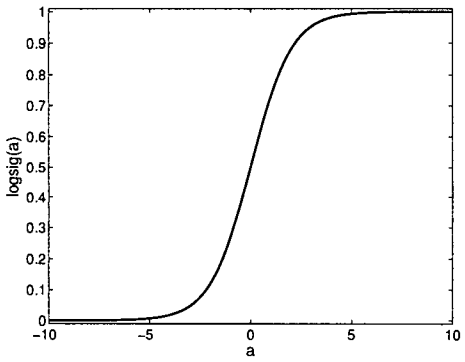

Figure 4.2: *The logistic sigmoid, or* logsig *function. As the sum of the weighted inputs goes from $-\infty$ to ∞, the function output goes from 0 to 1. The output of the logsig network approximates the posterior of the classification.*

significantly slower than any other algorithm I tried.

4.1.3 Results

In Figure 4.3, I plot the performance of networks trained with a few selected methods. In each case, I trained 100 networks – one for each label. The data set for each label was made up of all the data in the full data set that had the label, plus a equal-sized random selection of the unlabelled data. This was done to evaluate the unbiased network performance. The F-measure for each network was computed using $\alpha = 0.5$, and the results were sorted and plotted. The baseline is the expected value of F using random labellings. Three-layer logsig NN clearly dominates.

Figure 4.4 shows the mean and standard deviation of F for each type of network I evaluated (including some I didn't plot). Where applicable, I show the results of testing both on the unbiased data and the full, biased data set.

The clear winner of the NNs I evaluated was the three-layer logsig network. For every label, from acoustically-straightforward ones like ELECTRONICA to labels like ACERBIC with strong cultural components, it did better than chance in assigning labels, usually much better.

The three-layer set of networks is what I use for generating priors for the posterior network (Section 4.2). I used unbiased training sets for maximum flexibility, interpretability and training speed. Bias can always be introduced later in the process if need be.

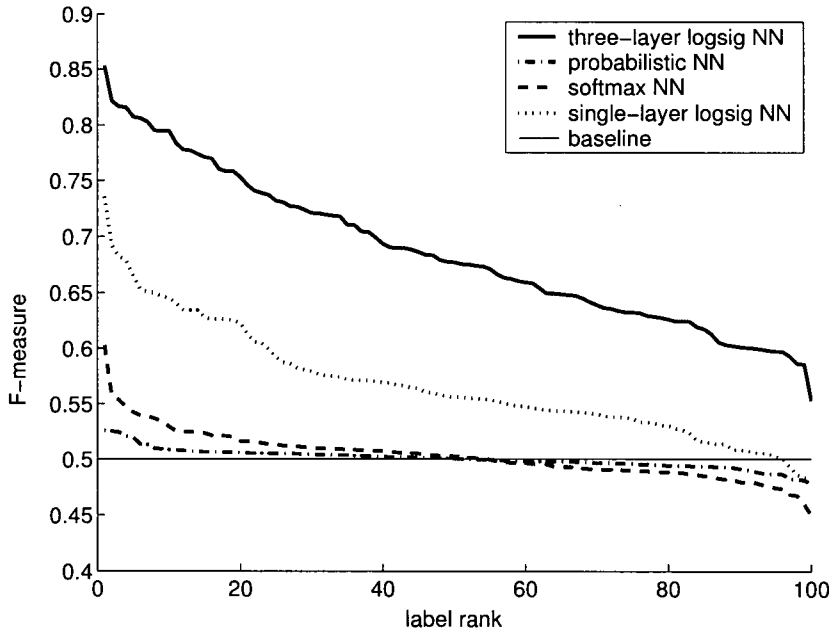

Figure 4.3: *F-measures of various types of networks, using 3-way cross-validation. The F-measure combines the precision and recall scores – a higher F-measure indicates success in both scores. The F-measures were computed for each label, and then sorted to visualize the distribution of results. The plots show the F-measures for all 100 labels using each type of network, from highest F-measure to lowest. The baseline is the expected prior F-measure of the test data using no training. Clearly, the multi-layer logsig NN is the best-performing network.*

	unbiased		biased	
network type	**mean**	**stdev**	**mean**	**stdev**
three-layer logsig NN	0.686	0.067	0.132	0.104
two-layer logsig NN	0.674	0.070	0.130	0.105
three-layer logsig NN, GDM	0.605	0.052	0.127	0.095
single-layer logsig NN	0.581	0.048	0.102	0.051
two-layer softmax NN	0.510	0.019	0.110	0.070
probabilistic NN	0.503	0.009	0.112	0.070
three-layer logsig NN, single network	–	–	0.124	0.088
two-layer logsig NN, single network	–	–	0.110	0.071

Figure 4.4: *F-measures for various NNs. Except for the entries labelled 'single', a separate NN was trained for each label. Parameters were discovered using scaled conjugate gradient, except for the network noted 'GDM', which used gradient decent with momentum. The mean and standard deviations of the F-measures of all the labellings is show for both the unbiased networks (which used data sets equally composed of positive and negative labellings), and biased (which used the entire data set and had many more negative labellings than positive).*

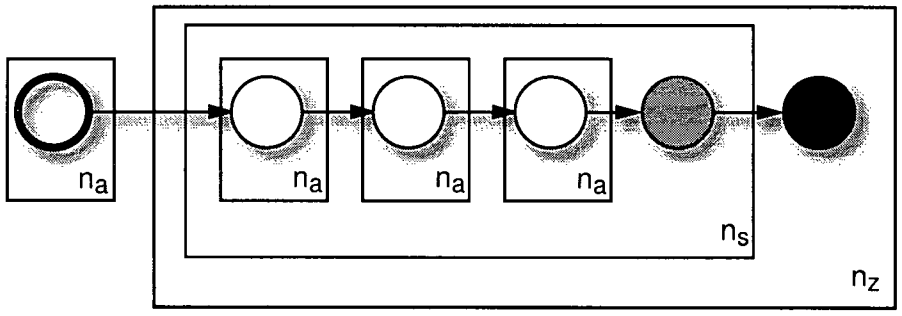

Figure 4.5: *Plate diagram of the posterior network given an n_a-dimensional audio features vector as input. The first five nodes form the prior network (Figure 4.1). The posterior node is the label probability for each of the n_z labels. Each has a subset of n_s prior labels (the outputs of the prior network) as parents.*

4.2 The posterior network

The prior network uses fairly common and well-understood Machine Learning techniques. In introducing a posterior network stage to the model, however, I stepped into less certain territory.

The prior network performs quite well for assigning probabilities to labels based on audio features alone. But it still performs worst on the most culturally-dependent ones. Even with the best training, which uses approximately 27000 neurons (nodes) all told, the performance is limited not only by the quality of the feature extraction, but by the lack of a cultural context for the labels. ELECTRONICA, can be learned reasonably accurately, but audio features alone do not perform anywhere near as well on many labels that rely on more subtle cultural information, such as ETHEREAL. In order for the system to work at a level beyond genre classification, it needs some way of modelling the cultural attributes of music.

Fortunately, there is a source of information that can be used to capture these cultural priors: the labels themselves, and their patters of co-occurrence. It is very likely, for example, that a song labelled ETHEREAL will also be labelled ELECTRONICA.

Ideally, I would like to be able to express the probability of a given label, $p(Z_i)$ as the posterior of the other labels, $p(Z_i|Z_{j\neq i})$. To take advantage of the co-occurrence information in the labels, however, a conditional probability table of k random variables requires 2^k entries. For example, expressing WRY in terms of IRONIC and ROCK requires values for $p(\text{WRY}|\text{ROCK}, \text{IRONIC})$, $p(\text{WRY}|\neg\text{ROCK}, \text{IRONIC})$, $p(\text{WRY}|\text{ROCK}, \neg\text{IRONIC})$ and $p(\text{WRY}|\neg\text{ROCK}, \neg\text{IRONIC})$. Expressing each label in terms of the other 99 labels would not only require a table with 2^{99} entries, but would depend on the data set being expressive enough to estimate probabilities for each of those 2^{99} entries from their frequencies in the database. Not an easy task, given that there are only about 2^{13} data in the database.

To get around this problem, I simply approximate the full conditional probability with a much smaller subset of n_s labels, defined separately for each label. So if S_i is the set of conditional labels for label i, then I need only evaluate $p(Z_i|Z_{s_i})$. If n_s is reasonably small, the problem becomes easily tractable by summing exhaustively over all the conditional probability table:

$$p(Z_i|Z_{S_i}) = \sum_{s\in S_i, u_s \in U} p(Z_i, Z_s = u_s) p(Z_{S_i})$$

where U is boolean.

The outputs from the NN thus become the observations of the priors in the complete posterior Bayesian network shown in Figure 4.5.

4.2.1 Prior label selection

Of course, that leaves the problem of determining the set of label priors S_i, to use for each posterior. While I tried several methods (which I will discuss shortly), the method I ultimately used was as follows:

1. Compute C, the matrix of correlation coefficients for all the labels in the database, from the label occurrence matrix, an n_x-by-n_z matrix in which each entry is 1 if training datum x is labelled with z and 0 otherwise.

2. Select an arbitrary n_s, the number of prior labellings for each posterior labelling.

3. For each posterior label i, use the n_s labels other than i that have the highest correlation coefficients – that is, for i, take row C_i, and use the indices (other than i) of the n_s highest values.

The result is that each posterior takes as its priors the labels that have been observed to co-occur with it the most frequently. This seems like a reasonable approach, and it gave the most satisfactory results of those I tried.

example

Suppose n_s is set to 2. Then to find a label, say, CYNICAL, MILQ finds the other 2 labels with the highest correlations coefficients, which in this case would be ACERBIC and WITTY. Then the frequency of co-occurrence of the labels is found:

labelling	frequency
CYNICAL ∧ ACERBIC ∧ WITTY	258
¬CYNICAL ∧ ACERBIC ∧ WITTY	105
CYNICAL ∧ ¬ ACERBIC ∧ WITTY	87
¬ CYNICAL ∧ ¬ ACERBIC ∧ WITTY	342
CYNICAL ∧ ACERBIC ∧ ¬ WITTY	120
¬CYNICAL ∧ ACERBIC ∧ ¬ WITTY	102
CYNICAL ∧ ¬ ACERBIC ∧ ¬ WITTY	338
¬ CYNICAL ∧ ¬ ACERBIC ∧ ¬ WITTY	6932

From this can be computed the conditional probabilities:

$p(\text{CYNICAL} \mid \text{ACERBIC},\text{WITTY}) = 0.7107$
$p(\text{CYNICAL} \mid \neg\text{ACERBIC},\text{WITTY}) = 0.2028$
$p(\text{CYNICAL} \mid \text{ACERBIC},\neg\text{WITTY}) = 0.5406$
$p(\text{CYNICAL} \mid \neg\text{ACERBIC},\neg\text{WITTY}) = 0.0464$

If MILQ then observes the prior network marginals $p(\text{CYNICAL}) = 0.1$, $p(\text{ACERBIC}) = 0.8$, $p(\text{WITTY}) = 0.9$, then $p(\text{CYNICAL}|Z_{S_i})$ is simply:

$$\begin{aligned} p(\text{CYNICAL}|\text{ACERBIC}, \text{WITTY}) &= 0.7107(0.8)(0.9) + 0.2028(0.2)(0.9) \\ &\quad +0.5406(0.8)(0.1) + 0.0464(0.2)(0.1) \\ &= 0.5928 \end{aligned}$$

Note that the prior network output $p(\text{CYNICAL})$ is completely discarded – the label does not send a message to itself, but relies completely on it's parents. The prior $p(\text{CYNICAL})$ of 0.1 will, however, be used to compute the posteriors of other labels that CYNICAL is a parent of.

alternatives

The method I have described is based on loopy belief propagation [Murphy *et al.*, 1999], which is simply Pearl's polytree algorithm [Pearl, 1988] applied to loopy graphs. In fact, my algorithm is simply a single iteration of loopy parent-to-child π-message passing.

When I attempted to iterate loopy belief propagation to convergence, I soon found that while the overall pairwise pseudo-likelihood improved, it was at the cost of 'flipping' certain label probabilities from near-zero to near-one, or vice-versa. The labels that were flipped were usually the same ones with each datum, and the flips were rarely correct. More investigation would be required to understand why this happened, but it seems that the labels that flip are ones in which the potential functions are uninformative (that is, close to chance), which permits the message propagation to alter them arbitrarily to maximize the likelihood in terms of the more strongly-defined potentials. I am currently investigating ways of learning more robust potential functions for the belief network.

I also experimented with using labels that have the highest *absolute* correlations, so that labels with strongly negative correlations would also be included, but the results were quite poor, possibly because the infrequency of many labels results in less useful values of C for labels that do not co-occur.

4.2.2 Implications

As the results of the posterior network are the heart of my research, I have devoted Chapter 5 to them.

Using co-occurrence to capture cultural information is not a new idea. Many Statistical Natural Language Processing tools rely on term co-occurrence for classification and other problems [Manning and Schütze, 1999]. But to my knowledge, it has not been applied in the context of a Bayesian network to the problem of approximating cultural information in a non-linguistic medium, though the work of Whitman *et al.* (Section 2.1) is similar.

One of the problems this model opens up is balancing two competing optimization problems. The prior network stage is concerned with optimizing each label locally, while the posterior network is designed to optimize the self-consistency of the labels. Unfortunately, it seems that one comes at the expense of the other. Without properly balancing of the two stages one will dominate. This is the problem explored in the next chapter.

Chapter 5

Results

What MILQ *learned.*

Evaluating a system like MILQ is not straightforward. It is ultimately designed not to solve a specific theoretical problem, but to be the basis of a real-world application. User studies can be valuable, and I would like to eventually do some, but it was not practical to do user evaluations on each of the dozens of model combinations I experimented with. Nor would it be reasonable to have a single user evaluate even a substantial portion of the database (listening to the 8551 songs in the database would take a single listener around three weeks of continuous listening).

Therefore, I had to come up with evaluations that conformed to what I saw as my own expectations from the system. This meant not only accuracy in the labellings, but a certain internally-consistent logic to the inevitable incorrect labellings. The system would never be exact, but certain errors are more acceptable than others. Incorrectly labelling R.E.M.'s 'Everybody Hurts' as ELEGANT should be more acceptable than labelling it EXUBERANT.

I decided on two methods of evaluation. In Section 5.1, I evaluate the correctness of the labellings from an Information Retrieval point of view. In Section 5.2, I evaluate from a Statistical point of view. While the model performs quite well in both evaluations, it is enlightening to look at its successes and limitations. I finish with the discussion of a few examples in Section 5.3.

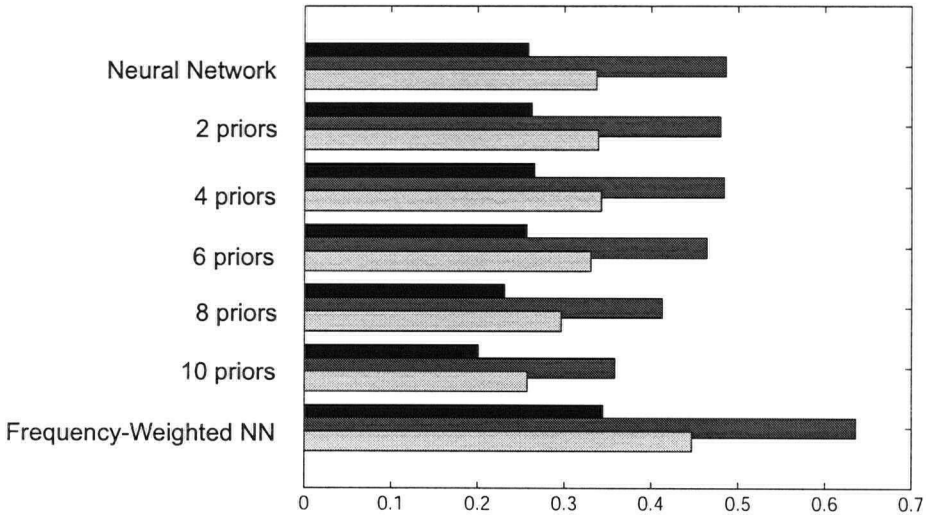

Figure 5.1: *The mean per-datum precision (black), recall (grey) and F-measure (white). Shown for the Neural Network alone, and for the complete posterior network using 2, 4, 6, 8 and 10 priors. Using 6 or fewer priors generates results similar to the Neural Network. However, by inducing a bias to the results of the Neural Network, it performs better than using prior observation. I will show that this comes at the cost of failing to take into account global properties of the labels.*

5.1 Accuracy evaluation

As discussed in Section 4.1, the fact that the labellings are probabilistic means that they can be ranked, and that this allows for evaluation of the model as an information retrieval one.

5.1.1 F-measures

As in Section 4.1.1, I used the F-measure to evaluate the success of the model in label prediction. However, instead of evaluating the labels as I did there, this time I was more concerned with the quality of results for each datum.

Using the labellings from the NN alone, and the Bayes net model using 2, 4, 6, 8 and 10 priors, I computed the F-measures using equally-weighted precision and recall, on the 10 highest-ranked labels for each datum. The mean precision, recall and F-measure of each model are shown in Figure 5.1.

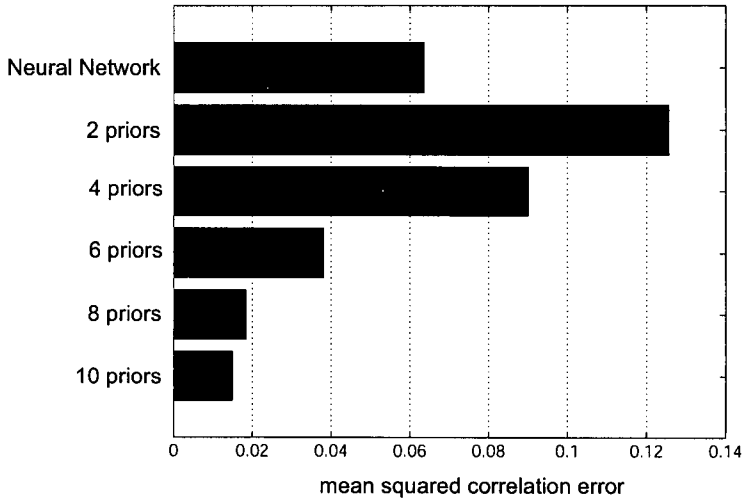

Figure 5.2: *Correlation errors for labelling schema. Shown is the mean squared error of the correlation matrix of each labelling from the correlations of the true labellings.*

The Neural Network alone already has fairly impressive results for such a difficult problem space. As long as the number of priors remains small, using prior observation performs approximately the same. However, if the outputs from the Neural Network are biased, so that each labelling is scaled by the frequency of the label in the whole database, the Neural Network model outperforms the Bayes net model. This is to be expected: by using observed label priors, some precision of the individual labelling will be sacrificed. This is the price paid for improving the global quality of the labelling.

5.2 Correlation evaluation

While precision and recall and derived evaluations are informative regarding the simple accuracy of the results, there is more to be taken into account. As mentioned above, it is desirable to look at each label in the context of other labels. FUN and BROODING are unlikely to occur together, but FUN and BOISTEROUS are much more reasonable.

To evaluate this error, I compared the correlation coefficient matrix of the original data with the test data. As with all my experiments, I used 4-way cross validation for each model, with each data used three times for

training and once for testing. In this way, a complete set of label predictions for the entire data set was computed for each model. To compute the error, I simply took the mean of the square of the difference of the correlation of the predictions and the correlations of the original labels. So if $\texttt{corr}$ is the matrix of correlation coefficients in the data set, and $\widehat{\texttt{corr}}$ is the correlation coefficient matrix of the predictions, then

$$error = \frac{1}{n_l^2} \sum_i^{n_z} \sum_j^{n_l} (\texttt{corr}_{i,j} - \widehat{\texttt{corr}}_{i,j})^2$$

Figure 5.2 shows this statistic for the Neural Network outputs without the Bayes net, and then for the labellings using 2, 4, 6, 8 and 10 label observations as priors for each labelling. Interestingly, the Neural Network alone performs quite well, presumably because it so successful at labelling to begin with. Using 2 or 4 prior observations actually performs worse than not using the Bayes Net at all, probably because such a small number of priors is not sufficient to approximate the effect of all the priors. Using 6 or more priors, however, gives a significant reduction in the error over the NNs alone.

5.3 Examples

Since these evaluations are meant to give some kind of measure of the abstract user experience, it is useful to look at how they actually affect that experience.

Figures 5.3, 5.4 and 5.5 show the labellings applied to a set of fairly well-known songs using the prior outputs alone, the prior outputs biased by the frequency of the labels in the database and the Bayes net outputs, respectively.

The Neural Network outputs alone (Figure 5.3) perform well, considering the difficult of the task, but do not do equally well for all tasks. Daft Punk's 'Da Funk' is labelled quite well, and the results for Portishead's 'Wandering Star' and John Lennon's 'Instant Karma' are not correct, but seem reasonable anyway. But the two Moby songs are almost comically wrong. 'Find My Baby' is labelled both PRECIOUS and HIP-HOP, and 'Porcelain' is simultaneously MANIC and SOOTHING, and GLOOMY and RAUCOUS. Results like this would not be likely to instill much confidence in users.

Weighting the prior outputs by the label frequencies (Figure 5.4) improves the accuracy remarkably, but the results are still unsatisfying. The

Portishead	'Wandering Star'	SOUNDTRACK	1.000
		LITERATE	1.000
		PRECIOUS	0.999
		ORGANIC	0.999
		DRUGGY	0.996
Daft Punk	'Da Funk'	•HOUSE	1.000
		•PARTY/CELEBRATORY	1.000
		•CLUB/DANCE	0.998
		HIP-HOP	0.997
		•EXUBERANT	0.998
Moby	'Find My Baby'	SOUNDTRACK	0.999
		•HOUSE	0.998
		PARTY/CELEBRATORY	0.994
		PRECIOUS	0.998
		HIP-HOP	0.991
Moby	'Porcelain'	MANIC	1.000
		GLOOMY	1.000
		TENSE	1.000
		RAUCOUS	1.000
		SOOTHING	1.000
Leonard Cohen	'I'm Your Man'	PRECIOUS	1.000
		JAZZ	0.999
		LAID-BACK/MELLOW	0.999
		ORGANIC	0.996
		•FOLK-ROCK	0.994
John Lennon	'Instant Karma'	DRUGGY	1.000
		WRY	1.000
		WITTY	0.999
		PROG-ROCK/ART-ROCK	0.999
		CHEERFUL	0.999

Figure 5.3: *Five highest-ranked labels for various songs using the Neural Network outputs. Correct labellings are marked with bullets.*

Portishead	'Wandering Star'	•ELECTRONICA	0.439
		•ALTERNATIVE POP/ROCK	0.236
		•STYLISH	0.239
		HYPNOTIC	0.208
		TRIPPY	0.195
Daft Punk	'Da Funk'	•ELECTRONICA	0.465
		•STYLISH	0.262
		•PLAYFUL	0.259
		•CLUB/DANCE	0.205
		QUIRKY	0.188
Moby	'Find My Baby'	STYLISH	0.247
		•ELECTRONICA	0.247
		•CLUB/DANCE	0.207
		•BROODING	0.154
		ALTERNATIVE POP/ROCK	0.129
Moby	'Porcelain'	ELECTRONICA	0.377
		•STYLISH	0.252
		•CLUB/DANCE	0.207
		•BROODING	0.175
		ALTERNATIVE POP/ROCK	0.132
Leonard Cohen	'I'm Your Man'	PLAYFUL	0.252
		•REFLECTIVE	0.231
		•DETACHED	0.231
		STYLISH	0.210
		ALTERNATIVE POP/ROCK	0.199
John Lennon	'Instant Karma'	STYLISH	0.234
		DETACHED	0.227
		ALTERNATIVE POP/ROCK	0.205
		•REFLECTIVE	0.190
		QUIRKY	0.188

Figure 5.4: *Five highest-ranked labels for various songs using the Neural Network outputs, scaled by the frequency of the label in the training set. Correct labellings are marked with bullets.*

Portishead	'Wandering Star'	EARNEST	0.707
		•REFLECTIVE	0.686
		•WISTFUL	0.684
		•AUTUMNAL	0.672
		ADULT ALTERNATIVE	0.658
Daft Punk	'Da Funk'	•BOISTEROUS	0.922
		•ENERGETIC	0.916
		•PLAYFUL	0.879
		•CLUB/DANCE	0.766
		ROLLICKING	0.621
Moby	'Find My Baby'	•ELECTRONICA	0.735
		PLAYFUL	0.472
		SOMBER	0.410
		CYNICAL/SARCASTIC	0.396
		AGGRESSIVE	0.342
Moby	'Porcelain'	•ELECTRONICA	0.903
		•CLUB/DANCE	0.703
		•TECHNO	0.674
		SOMBER	0.596
		CALM/PEACEFUL	0.557
Leonard Cohen	'I'm Your Man'	AUTUMNAL	0.701
		•REFLECTIVE	0.670
		WISTFUL	0.642
		CATHARTIC	0.629
		ALTERNATIVE POP/ROCK	0.615
John Lennon	'Instant Karma'	IRONIC	0.909
		WITTY	0.907
		PROG-ROCK/ART-ROCK	0.882
		•CYNICAL/SARCASTIC	0.851
		WRY	0.833

Figure 5.5: *Five highest-ranked labels for various songs under Bayes net ranking using the Neural Network outputs of 6 related labels as priors. Correct labellings are marked with bullets.*

same labels are selected repeatedly: all six songs are labelled STYLISH, and five of the six are labelled ALTERNATIVE POP/ROCK, due simply to the preponderance of those labels in the training set. However, we no longer see such remarkably counterintuitive labels as we did in the unbiased outputs, perhaps because the incorrect labels are ones that occur quite frequently in the database, and hence are broad enough to seem fairly reasonable for a wide class of songs.

Figure 5.5 shows the results of basing each label on the observations of six similar labels. While interpretation is subjective, of course, it seems that the loss of accuracy over the NN alone makes for much more satisfying results. None of the labels seem out of place, whether they agree with the ground truth or not. The two very different Moby songs, which both appear on *Play* (and therefor have the same ground truth labelling), are labelled quite appropriately: the bluesy 'Find My Baby' is PLAYFUL and CYNICAL/SARCASTIC, while the far more relaxed electronica piece 'Porcelain' is CALM/PEACEFUL. While user studies will be necessary to confirm these results, I was pleased to observe that in may cases the posterior network actually seemed to improve on the ground truth labelling.

5.4 MILQDemo

In order to explore the results and application possibilities, I have implemented a solution visualization demo. As this is still a work under construction, and it is likely to remain so for some time, interested readers are advised to view the project web page, `http://www.cs.ubc.ca/ ebrochu/milq` for the latest news, screen shots and movies.

As of this writing, MILQDemo exists as an application that takes a trained MILQ model, and when given a new song, computes the prior and posterior probabilities and allows various means of visualization.

It is written in C++ using OpenGL. It was designed as a visualization application that interfaces with Apple's iTunes on Mac OS X, or XMMS on Linux. When a song is started on either player, information about the file is sent to the MILQ demo, which is used to look up the feature vector for the song (currently computed offline). This is given as input to the model stored in the demo, and the model outputs are used for visualization. Examples are shown in Figures 1.1, 5.6 and 5.7.

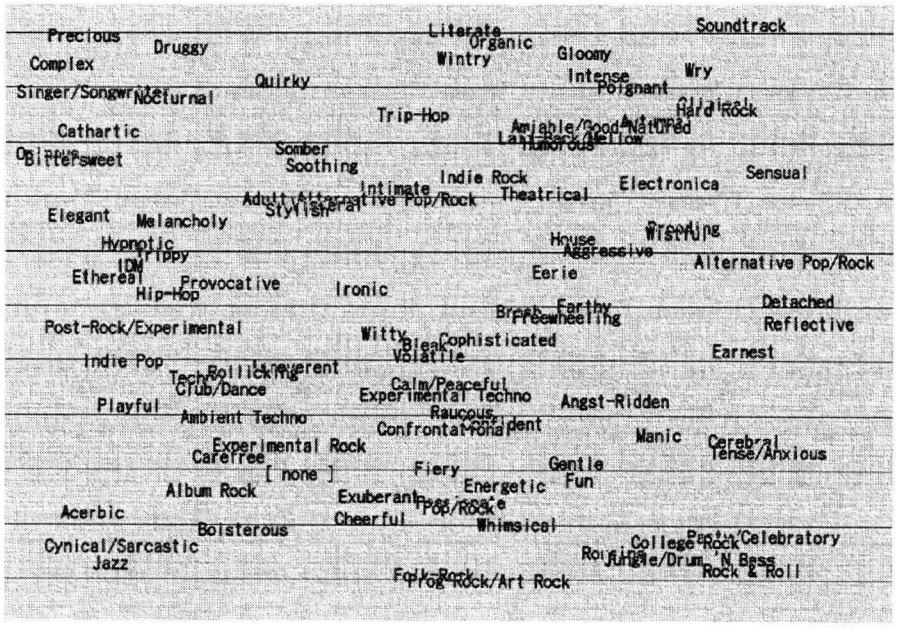

Figure 5.6: *Complete posterior feature set for the song 'Wandering Star' by Portishead (also used in Figure 1.1). Higher-probability features are positioned higher in the image. From a screen capture of the MILQDemo software.*

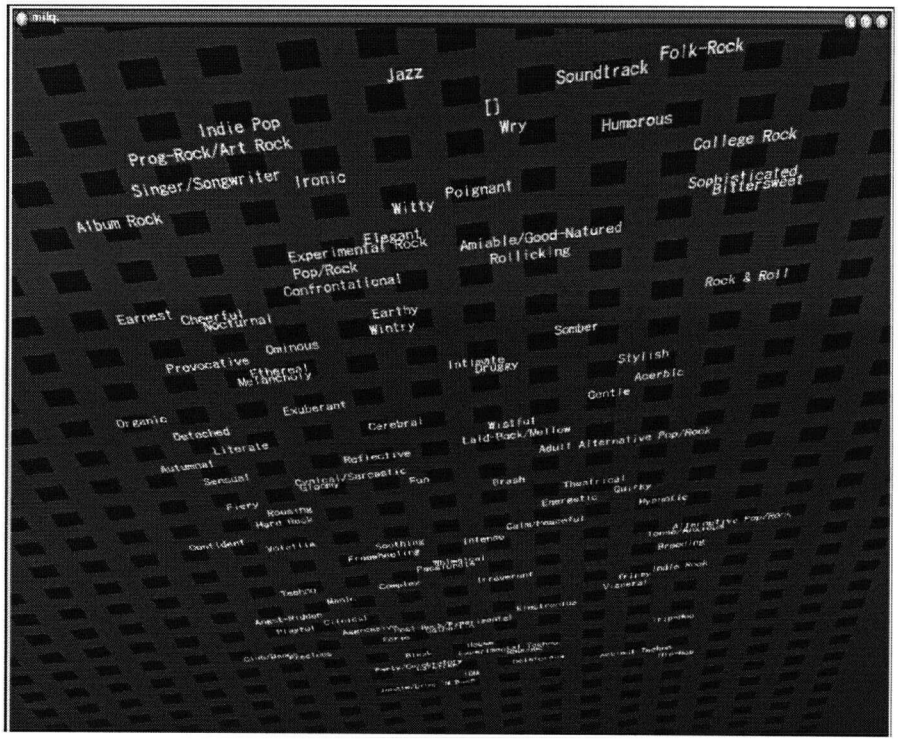

Figure 5.7: MILQ *screen capture for Leonard Cohen's 'I'm Your Man'. Labels assigned the highest probabilities are closest. Note that the 'ground truth' binary labelling consists of the labels* FOLK-ROCK, WRY, IRONIC, LITERATE *and* SINGER/SONGWRITER. *Nevertheless, by utilizing the entire model and training set, a reasonable ranking is given to all 100 labels in the model. More screen shots, examples and current project status can be found at* `http://www.cs.ubc.ca/~ebrochu/milq`.

Chapter 6

Conclusion

What I Learned.

I set out on this project with the goal of seeing whether computers could learn the moods of music. It seems, to a large degree, that they can, given a suitably generous definition of what a 'mood' is.

Using a large data set (Section 3.1), a selected set of labels (Section 3.4; Appendix A) and good feature extraction (Section 3.2), I was able to tailor a set of Neural Networks that could predict not only labels like genre, but also styles and even adjective keywords which corresponded to the tone, or mood of the music (Section 4.1). Unsurprisingly, however, the Neural Networks performed poorly on labels that required a lot of cultural context to appreciate.

Fortunately, the set of labels is large enough that a certain amount of cultural context can be introduced by basing the probability of each label on the observations (the prior network) made on related labels (Section 4.2). Determining whether a piece of music is FUN is difficult for a Neural Network, but by looking at the scores for easier labels like BOISTEROUS and MANIC, a better estimate can be made. Labelling like this comes at some cost of the precision and recall of individual labels (Section 5.1), but improves the correlation of the labels (Section 5.2).

6.1 Applications

In addition to the 'milq demo' project, there are a number of applications of this technology that would be interesting to explore.

- The labels could be used for music visualizers, like those packaged with iTunes, XMMS, WinAmp or Windows Media Player. For the most part, these programs use random numbers and beat detection to generate visualizations. But by finding a mapping from the mood-based labels to sets of visualizations that matched that mood, visuals could be created that matched the emotional content of the music being played. I plan to implement a system in 2004, tentatively called HONI to do just that.

- The vector could be used to locate songs in a 'mood-space'. From this, we could do clustering, browsing or retrieval, for example. A query-by-example could return songs with similar moods, or query-by-label could generate playlists constrained by their labels, returning songs that are FUN and ELECTRONICA but not SILLY, for example.

- The model need not be restricted to music. By changing the input and training on appropriately-labelled documents, the same model could be used to apply culturally-dependent labels on books, or images, or video.

6.2 Future work

In addition to the possible applications of this technology, there are numerous refinements that could, perhaps *should*, be made to make MILQ a viable technology.

- Probably the most important net step would be to run a user study. Since the model is so closely related to trying to anticipate user expectation, a user study could be very enlightening at this stage. Such a study would probably involve the user listening to music while being presented with labels generated from various models (random labelling, prior outputs only, posterior outputs, etc), without knowing which one they were being given. The user would be asked to evaluate the labellings and the results could be examined.

- The model was very intentionally designed to be fairly agnostic as far as feature spaces on the input are concerned. Additional feature-extraction methods could be added, or used to replace existing ones. More sophisticated audio features could very significantly improve the quality of the results.

- From a practical point of view, it will also be necessary to implement *faster* feature extraction. The average time for full feature extraction on the methods I use is over a minute per song on a 2.5 GHz Xeon processor. This is simply not acceptable if users are applying the model to novel music and expecting timely results.

- The training labels could stand to be greatly improved. The 'ground truth' labelling set is inconsistent, incomplete, and is applied on a per-album level. What is needed is expert labelling of the individual songs in a consistent manner. Perhaps the best way to do this is through some kind of distributed system where users submit labels for songs as they listen to them. Of course, the users would have to have the sense that they were getting something in return, and there would be a threshold of participants before the system would work. This is the model used by MoodLogic. It would be very interesting to explore this topic.

Appendix A

Labels

label	type	frequency
Acerbic	tone	424
Adult Alternative	style	764
Aggressive	tone	729
Album Rock	style	700
Alternative Pop/Rock	style	2940
Ambient Techno	style	565
Amiable/Good-Natured	tone	594
Angry	tone	307
Angst-Ridden	tone	588
Autumnal	tone	417
Bittersweet	tone	1029
Bleak	tone	407
Brash	tone	583
Brooding	tone	1163
Calm/Peaceful	tone	545
Cathartic	tone	967
Cerebral	tone	997
Cheerful	tone	296
Clinical	tone	638
Club/Dance	style	1402
College Rock	style	542
Complex	tone	796
Confident	tone	687
Confrontational	tone	494
Cynical/Sarcastic	tone	580
Detached	tone	1337

label	type	frequency
Druggy	tone	478
Earnest	tone	835
Earthy	tone	623
Eerie	tone	833
Electronica	genre	3005
Elegant	tone	594
Energetic	tone	728
Ethereal	tone	829
Experimental Rock	style	486
Experimental Techno	style	374
Exuberent	tone	544
Fiery	tone	495
Folk-Rock	style	296
Freewheeling	tone	1025
Fun	tone	570
Gentle	tone	478
Gloomy	tone	421
Hard Rock	style	443
Heavy Metal	style	331
Hip-Hop	style	292
House	style	299
Humorous	tone	365
Hypnotic	tone	1134
IDM	style	609
Indie Pop	style	419
Indie Rock	style	1084
Intense	tone	999
Intimate	tone	1210
Ironic	tone	679
Irreverent	tone	633
Jazz	genre	426
Jungle/Drum 'N' Bass	style	340
Laid-Back/Mellow	tone	780
Literate	tone	992
Manic	tone	323
Melancholy	tone	1009
Nocturnal	tone	1059
Ominous	tone	453
Organic	tone	475
Party/Celebratory	tone	378

label	type	frequency
PASSIONATE	tone	697
PLAYFUL	tone	1611
POIGNANT	tone	868
POP/ROCK	style	898
POST-ROCK/EXPERIMENTAL	style	300
PROG-ROCK/ART-ROCK	style	332
PROVOCATIVE	tone	462
QUIRKY	tone	1255
RAUCOUS	tone	402
REFLECTIVE	tone	1507
ROCK	genre	5551
ROCK & ROLL	style	415
ROLLICKING	tone	380
ROUSING	tone	613
SENSUAL	tone	841
SINGER/SONGWRITER	style	1038
SOMBER	tone	613
SOOTHING	tone	660
SOPHISTICATED	tone	1128
SOUNDTRACK	style	307
STYLISH	tone	1566
TECHNO	style	381
TENSE/ANXIOUS	tone	426
THEATRICAL	tone	1166
TRIP-HOP	style	1041
TRIPPY	tone	1129
VISCERAL	tone	330
VOLATILE	tone	562
WHIMSICAL	tone	543
WINTRY	tone	437
WISTFUL	tone	757
WITTY	tone	611
WRY	tone	537

Table A.1: The 100 labels used for the experiments and applications discussed in this thesis. The three types are 'genre' – the general type of song – 'style', which can be seen as a kind of sub-genre (most styles co-occur with only one genre), and 'tone', describing the emotional qualities of the music. While the most novel problems this thesis deals with are in the tone qualities, the more easily learned genre and style labels can be used to assist labelling the more challenging tones. Also shown is the 'ground truth' frequency of each label in the database (out of 8556 songs).

Bibliography

[Aucouturier and Pachet, 2003] J-J Aucouturier and F Pachet. Representing musical genre: A state of art. *Journal of New Music Research*, 32(1), 2003.

[Bao, 2003] K Bao. On-line EM and quasi-Bayes, or: How I learned to stop worrying and love stochastic approximation. Master's thesis, University of British Columbia, Vancouver, 2003.

[Beran, 2004] J Beran. *Statistics in Musicology.* Chapman & Hall/CRC Press, Boca Raton, USA, 2004.

[Bishop, 1995] C M Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, New York, 1995.

[Bridle, 1990] J S Bridle. Probabilistic interpretation of feed forward networks with relationship to statistical pattern recognition. In F F Soulie and J Herault, editors, *Neurocomputing: Algorithms, Architectures and Applications*, New York, 1990. Springer-Verlag.

[Brochu and de Freitas, 2003] E Brochu and N de Freitas. "Name that Song!": A probabilistic approach to querying on music and text. In *Advances in Neural Information Processing Systems 15*, Cambridge, USA, 2003. MIT Press.

[Brochu *et al.*, 2003] E Brochu, N de Freitas, and K Bao. The sound of an album cover: Probabilistic multimedia and IR. In C M Bishop and B J Frey, editors, *Proceedings of Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, USA, 2003.

[de Freitas *et al.*, 2003] N de Freitas, E Brochu, K Barnard, P Duygulu, and D Forsyth. Bayesian models for massive multimedia databases: a new frontier. Technical Report TR-2003-005, Department of Computer Science, University of British Columbia, 2003.

[Foote and Cooper, 2001] J Foote and M Cooper. Visualizing musical structure and rhythm via self-similarity. In *Proceedings of the International Conference on Computer Music*, Havana, 2001.

[Foote and Uchihashi, 2001] J Foote and S Uchihashi. The beat spectrum: a new approach to rhythm analysis. In *Proceedings of the 2001 International Conference on Multimedia and Expo*, Tokyo, 2001.

[Foote *et al.*, 2002] J Foote, M Cooper, and U Nam. Audio retrieval by rhythmic similarity. In *Proceedings of the 2002 International Symposium on Music Information Retrieval*, 2002.

[Golub, 2000] S Golub. Classifying recorded music. Master's thesis, University of Edinburgh, 2000.

[Hoos *et al.*, 2001a] H H Hoos, K A Hamel, K Renz, and J Kilian. Representing score-level music using the GUIDO music-notation format. *Computing in Musicology*, 12, 2001.

[Hoos *et al.*, 2001b] H H Hoos, K Renx, and M Gorg. GUIDO/MIR - an experimental musical information retrieval system based on GUIDO music notation. In *Proceedings of the International Symposium on Music Information Retrieval*, 2001.

[Jordan, 1995] M I Jordan. Why the logistic function? A tutorial discussion on probabilities and neural networks. Technical Report 9503, Massachusetts Institute of Technology, Computational Cognitive Science, 1995.

[LaLoudouana and Tarare, 2002] D LaLoudouana and M B Tarare. *Data Set Selection.* 2002. Presented at the Neural Information Processing Systems 2002 Workshop.

[Manning and Schütze, 1999] C D Manning and H Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge, USA, 1999.

[Miller, 1990] G A Miller. WordNet: an on-line lexical database. *International Journal of Lexicography*, 3(4), 1990.

[Moller, 1993] M Moller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4), 1993.

[Murphy *et al.*, 1999] K P Murphy, Y Weiss, and M I Jordan. Loopy belief propagation for approximate inference: An empirical study. In K B Laskey and H Prade, editors, *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, Stockholm, 1999.

[Pachet, 2003] F Pachet. Content management for electronic music distribution. *Communications of the ACM*, 46(4), 2003.

[Pampalk, 2001] E Pampalk. Islands of music: Analysis, organization, and visualization of music archives. Diplomarbeit, Institut für Softwaretechnik und Interaktive Systeme der Technischen Universität Wien, Vienna, 2001.

[Pearl, 1988] J Pearl. *Probabilistic Reasoning in Expert Systems.* Morgan Kaufman, 1988.

[Platt *et al.*, 2002] J C Platt, C J C Burges, S Swenson, C Weare, and A Zheng. Learning a Gaussian process prior for automatically generating music playlists. In *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[Rifkin, 2002] R M Rifkin. *Everything Old is New Again: a Fresh Look at Historical Approaches to Machine Learning.* PhD thesis, Massachusetts Institute of Technology, 2002.

[Rumelhard *et al.*, 1995] D E Rumelhard, R Durbin, R Golden, and Y Chauvin. Backpropogation: the theory. In Y Chauvin and D E Rumelhart, editors, *Backpropogation: Theory, Architectures and Applications*, pages 1–34, Hillsdale, 1995. Lawrence Erlbaum.

[Tzanetakis *et al.*, 2001] G Tzanetakis, G Essl, and P Cook. Automatic musical genre classification of audio signals. In *Proceedings of the 2001 International Symposium on Music Information Retrieval*, 2001.

[van Rijsbergen, 1979] C J van Rijsbergen. *Information Retrieval.* Buttersworth, London, 1979.

[Vapnik, 1998] V Vapnik. *Statistical Learning Theory.* John Wiley, New York, 1998.

[Vinet *et al.*, 2002] H Vinet, P Herrera, and F Pachet. The CUIDADO project. In *Proceedings of the 3rd International Symposium on Music Information Retrieval*, 2002.

[Wasserman, 1993] P D Wasserman. *Advanced Methods in Neural Computing.* Van Nostrand Reinhold, New York, 1993.

[Whitman and Rifkin, 2002] B Whitman and R Rifkin. Musical query-by-description as a multiclass learning problem. In *Proceedings of the IEEE Multimedia Signal Processing Conference*, St Thomas, USA, 2002.

[Whitman and Smaragdis, 2002] B Whitman and P Smaragdis. Combining musical and cultural features for intelligent style detection. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, Paris, 2002.

[Whitman *et al.*, 2003] B Whitman, D Roy, and B Vercoe. Learning word meanings and descriptive parameter spaces from music. In *Proceedings of the HLT-NAACL03 Workshop on Learning Word Meaning from Non-Linguistic Data*, Edmonton, Canada, 2003.

[Wold *et al.*, 1996] E Wold, T Blum, D Keislar, and J Wheaton. Content-based classification, search and retrieval of audio. *IEEE Multimedia*, 3(3), 1996.

[Zwicker and Fastl, 1999] J Zwicker and H Fastl. *Psychoacoustics: Facts and Models.* Springer, Berlin, Germany, 1999.