AUDIO TAGGING WITH CBA

Matthew D. Hoffman
Dept. of Computer Science
Princeton University
mdhoffma at cs.princeton.edu

David M. Blei
Dept. of Computer Science
Princeton University
blei at cs.princeton.edu

Perry R. Cook
Dept. of Computer Science
Dept. of Music
Princeton University
prc at cs.princeton.edu

ABSTRACT

This submission to the MIREX audio tag prediction task is an implementation of the Codeword Bernoulli Average (CBA) model presented at this year's ISMIR. CBA is a probabilistic model that learns to predict the probability that a word applies to a song from audio. Our model is simple to implement, fast to train, and predicts tags for new songs quickly.

1. INTRODUCTION

It has been said that talking about music is like dancing about architecture, but people nonetheless use words to describe music. In this paper we will present a simple system that addresses tag prediction from audio—the problem of predicting what words people would be likely to use to describe a song.

Two direct applications of tag prediction are semantic annotation and retrieval. If we have an estimate of the probability that a tag applies to a song, then we can say what words in our vocabulary of tags best describe a given song (automatically annotating it) and what songs in our database a given word best describes (allowing us to retrieve songs from a text query).

To address this problem, we use the Codeword Bernoulli Average (CBA) model, a probabilistic model that attempts to predict the probability that a tag applies to a song based on a vector-quantized (VQ) representation of that song's audio. Our CBA-based approach to tag prediction

- Is easy to implement using a simple EM algorithm.
- Is fast to train.
- Makes predictions efficiently on unseen data.

2. DATA REPRESENTATION

2.1 A vector-quantized representation

We begin by extracting a sequence of 13-dimensional MFCC vectors from each song, and appending to each feature vector the first and second derivatives ("delta" and "delta-delta") of each dimension, yielding a 39-dimensional MFCC-Delta feature representation. Rather than work directly with the MFCC-Delta feature representation, we first vector quantize all of the feature vectors in the corpus, ignoring for the moment what feature vectors came from what songs. We:

1. Normalize the feature vectors so that they have mean 0 and standard deviation 1 in each dimension.
2. Run the k-means algorithm [1] on a subset of randomly selected feature vectors to find a set of K cluster centroids.
3. For each normalized feature vector f_{ji} in song j, assign that feature vector to the cluster k_{ji} with the smallest squared Euclidean distance to f_{ji}.

This vector quantization procedure allows us to represent each song j as a vector $\boldsymbol{n}_j$ of counts of a discrete set of codewords:

$$n_{jk} = \sum_{i=1}^{N_j} 1(k_{ji} = k) \tag{1}$$

where n_{jk} is the number of feature vectors assigned to codeword k, N_j is the total number of feature vectors in song j, and $1(a = b)$ is a function returning 1 if $a = b$ and 0 if $a \neq b$.

This discrete "bag-of-codewords" representation is less rich than the original continuous feature vector representation. However, it is effective. Such VQ codebook representations have produced state-of-the-art performance in image annotation and retrieval systems [2], as well as in systems for estimating timbral similarity between songs [3,4].

3. THE CODEWORD BERNOULLI AVERAGE MODEL

In order to predict what tags will apply to a song and what songs are characterized by a tag, we developed the Codeword Bernoulli Average model (CBA). CBA models the conditional probability of a tag w appearing in a song j conditioned on the empirical distribution $\boldsymbol{n}_j$ of codewords extracted from that song. One we have estimated CBA's hidden parameters from our training data, we will be able to quickly estimate this conditional probability for new songs.

3.1 Generative process

CBA assumes a collection of binary random variables $\boldsymbol{y}$, with $y_{jw} \in \{0, 1\}$ determining whether or not tag w applies to song j. These variables are generated in two steps. First, a codeword $z_{jw} \in \{1, \ldots, K\}$ is selected with probability proportional to the number of times n_{jk} that that codeword appears in song j's feature data:

$$p(z_{jw} = k|\boldsymbol{n}_j, N_j) = \frac{n_{jk}}{N_j} \tag{2}$$

Then a value for y_{jw} is chosen from a Bernoulli distribution with parameter β_{kw}:

$$\begin{aligned} p(y_{jw} = 1|z_{jw}, \boldsymbol{\beta}) &= \beta_{z_{jw}w} \\ p(y_{jw} = 0|z_{jw}, \boldsymbol{\beta}) &= 1 - \beta_{z_{jw}w} \end{aligned} \tag{3}$$

The full joint distribution over $\boldsymbol{z}$ and $\boldsymbol{y}$ conditioned on the observed counts of codewords $\boldsymbol{n}$ is:

$$p(\boldsymbol{z}, \boldsymbol{y}|\boldsymbol{n}) = \prod_w \prod_j \frac{n_{jz_{jw}}}{N_j} \beta_{z_{jw}w} \tag{4}$$

3.2 Inference using expectation-maximization

We fit CBA with maximum-likelihood (ML) estimation. Our goal is to estimate a set of values for our Bernoulli parameters $\boldsymbol{\beta}$ that will maximize the likelihood $p(\boldsymbol{y}|\boldsymbol{n}, \boldsymbol{\beta})$ of the observed tags $\boldsymbol{y}$ conditioned on the VQ codeword counts $\boldsymbol{n}$ and the parameters $\boldsymbol{\beta}$. Analytic ML estimates for $\boldsymbol{\beta}$ are not available because of the latent variables $\boldsymbol{z}$. We use the Expectation-Maximization (EM) algorithm, a widely used coordinate ascent algorithm for maximum-likelihood estimation in the presence of latent variables [5].

Each iteration of EM operates in two steps. In the expectation ("E") step, we compute the posterior of the latent variables $\boldsymbol{z}$ given our current estimates for the parameters $\boldsymbol{\beta}$. We define a set of expectation variables h_{jwk} corresponding to the posterior $p(z_{jw} = k|\boldsymbol{n}, \boldsymbol{y}, \boldsymbol{\beta})$:

$$h_{jwk} = p(z_{jw} = k|\boldsymbol{n}, \boldsymbol{y}, \boldsymbol{\beta}) \tag{5}$$

$$= \frac{p(y_{jw}|z_{jw} = k, \boldsymbol{\beta}) p(z_{jw} = k|\boldsymbol{n})}{p(y_{jw}|\boldsymbol{n}, \boldsymbol{\beta})} \tag{6}$$

$$= \begin{cases} \frac{n_{jk}\beta_{kw}}{\sum_{i=1}^{K} n_{ji}\beta iw} & \text{if } y_{jw} = 1 \\ \frac{n_{jk}(1-\beta_{kw})}{\sum_{i=1}^{K} n_{ji}(1-\beta iw)} & \text{if } y_{jw} = 0 \end{cases} \tag{7}$$

In the maximization ("M") step, we find maximum-likelihood estimates of the parameters $\boldsymbol{\beta}$ given the expected posterior sufficient statistics:

$$\beta_{kw} \leftarrow \mathbb{E}[y_{jw}|z_{jw} = k, \boldsymbol{h}] \tag{8}$$

$$= \frac{\sum_j p(z_{jw} = k|\boldsymbol{h}) y_{jw}}{\sum_j p(z_{jw} = k|\boldsymbol{h})} \tag{9}$$

$$= \frac{\sum_j h_{jwk} y_{jw}}{\sum_j h_{jwk}} \tag{10}$$

By iterating between computing $\boldsymbol{h}$ (using equation 7) and updating $\boldsymbol{\beta}$ (using equation 10), we find a set of values for $\boldsymbol{\beta}$ under which our training data become more likely.

3.3 Generalizing to new songs

Once we have inferred a set of Bernoulli parameters $\boldsymbol{\beta}$ from our training dataset, we can use them to infer the probability that a tag w will apply to a previously unseen song j based on the counts $\boldsymbol{n}_j$ of codewords for that song:

$$\begin{aligned} p(y_{jw}|\boldsymbol{n}_j, \boldsymbol{\beta}) &= \sum_k p(z_{jw} = k|\boldsymbol{n}_j) p(y_{jw}|z_{jw} = k) \\ p(y_{jw} = 1|\boldsymbol{n}_j, \boldsymbol{\beta}) &= \frac{1}{N_j} \sum_k n_{jk}\beta_{kw} \end{aligned} \tag{11}$$

As a shorthand, we will refer to our inferred value of $p(y_{jw} = 1|\boldsymbol{n}_j, \boldsymbol{\beta})$ as s_{jw}.

Once we have inferred s_{jw} for all of our songs and tags, we can use these inferred probabilities both to retrieve the songs with the highest probability of having a particular tag and to annotate each song with a subset of our vocabulary of tags.

The cost of computing each s_{jw} using equation 11 is linear in the number of codewords K, and the cost of vector quantizing new songs' feature data using the previously computed centroids obtained using k-means is linear in the number of features, the number of codewords K, and the length of the song. For practical values of K, the total cost of estimating the probability that a tag applies to a song is comparable to the cost of feature extraction. Our approach can therefore tag new songs efficiently, an important feature for large commercial music databases.

4. REFERENCES

[1] J.B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob., Vol. 1*, 1966.

[2] C. Wang, D. Blei, and L. Fei-Fei. Simultaneous image classification and annotation. In *Proc. IEEE CVPR*, 2009.

[3] M. Hoffman, D. Blei, and P. Cook. Content-based musical similarity computation using the hierarchical Dirichlet process. In *Proc. International Conference on Music Information Retrieval*, 2008.

[4] K. Seyerlehner, A. Linz, G. Widmer, and P. Knees. Frame level audio similarity—a codebook approach. In *Proc. of the 11th Int. Conference on Digital Audio Effects (DAFx08), Espoo, Finland, September*, 2008.

[5] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.