Abstract—Music tags provide different types of semantic
information about music. Recently, automatic music tagging has
generated a great deal of interest among researchers in the field
of music information retrieval. In this paper, we propose a
posterior weighted Bernoulli mixture model (PWBMM) that
automatically annotates a song with tags, or retrieves relevant
songs given a semantic tag. The PWBMM approach improves
the codeword Bernoulli average (CBA) approach in two ways.
First, it uses a Gaussian mixture model-based posterior
representation of music feature vectors, instead of a vector-
quantized representation; second, it incorporates the counts of
tags assigned to training songs when training the Bernoulli
mixture model. The results of music tag annotation and retrieval
experiments on the MajorMiner and CAL-500 datasets
demonstrate that the PWBMM approach outperforms the CBA
approach in terms of different evaluation metrics.

I. INTRODUCTION
The goal of automatic music tagging is to model the relations
between text (tags) and music. There are two types of tasks:
(1) annotating an unseen song with tags; and (2) retrieving
songs from a music database by inputting a pre-defined tag.

In recent years, automatic music tagging has generated a
great deal of interest among researchers in the field of music
information retrieval. It has been one of the evaluation tasks
at the annual Music Information Retrieval Evaluation
eXchange (MIREX) since 2008\(^1\). The system that won the
MIREX 2008 competition [1] models the feature distribution
of each tag with a Gaussian mixture model (GMM) and
estimates the model's parameters with a weighted mixture
hierarchies expectation maximization algorithm. In contrast to
using probability models, Eck et al. [2] use AdaBoost to
automatically generate audio tags for music recommendation.
In addition, Hoffman et al. [3] propose the codeword
Bernoulli average (CBA) model, which is a probabilistic
model that attempts to predict the probability that a tag
applies to a song based on a vector-quantized (VQ)
representation of the song's audio. The codewords perform
well in terms of data representation if there are sufficient
clusters. However, vector quantization based on the K-means
derived codebook reveals hard-assignment of a vector in a test
song to its closest codeword; hence, the differentiation and
unseen-coverage ability is not good enough.

In this paper, we try to improve the CBA approach in two
ways. First, we replace the K-means derived codebook with a
GMM. As a result, the sequence of frame-based feature
vectors of a song is represented by a fixed-dimensional vector
of posterior probabilities of the Gaussian components in the
GMM, instead of the normalized counts of the codewords in
the K-means codebook. Second, the count of a tag assigned to
a song is taken into account when training the Bernoulli
mixture model (BMM). The results of music annotation and
retrieval experiments on two music datasets demonstrate that
the new approach outperforms the CBA approach.

The remainder of this paper is organized as follows. In
Section 2, we describe the music features used in this work
and how we exploit GMM for music data representation. In
Section 3, we introduce the Bernoulli mixture model and
explain how to apply it in music tagging. The experiments
and results are detailed in Section 4. Finally, we summarize
our conclusions and discuss our future work in Section 5.

II. MUSIC DATA REPRESENTATION
In this section, we describe the music features used in this
work, and explain how we exploit GMM posterior
probabilities to transform frame-based feature vectors into
song-level data representations.

A. Music Feature Extraction
To apply our model to predict a song’s tags, we must first
index the features of the song as a sequence of fixed-
dimensional vectors. We use low-level audio features because
they contain musical information that can be measured by
processing the audio signal directly. The features are
computed within a fixed-size frame of the signal, and the
values of different features are bundled as a high dimensional
vector. We do not consider high-level music descriptions,
such as pitch/melody, tempo/beat, chord/bass-line, key/harmonic,
and structure, as they are related to musical
concepts that are difficult to define with exact values, or the
corresponding extractors may lead to errors and uncertainties.
For example, the musical structure cannot be described by a
value, while the pitch extractor might output double pitches.
A detailed discussion of high-level music descriptions and
low-level audio features can be found in [4].

\(^1\) http://www.music-ir.org/mirex/2008/
To extract music features, we use MIRToolbox 1.2.5\(^2\), a free software that comprises approximately 50 audio/music feature extractors and statistical descriptors [5]. As shown in Table I, we consider four categories of features in this work, namely, dynamic, spectral, timbre, and tonal features. All the features are extracted in the same fixed-size short-time frame to ensure alignment and prevent the mismatch of different features in a vector. Therefore, we do not use some dynamic frame-length features (e.g., onset, event-density, pulse-clarity, attack-slop, and attack-time features), or long-term features (e.g., fluctuations and lower-energy features). A sequence of 70-dimensional feature vectors is extracted from a 16-bit, 22kHz, mono music waveform with a 50ms frame size and 0.5 hop shift.

<table>
<thead>
<tr>
<th>Category</th>
<th>Feature Description</th>
<th>Dim</th>
</tr>
</thead>
<tbody>
<tr>
<td>dynamic</td>
<td>rms</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>centroid: mean</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>spread: standard deviation</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>skewness</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>kurtosis</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>entropy</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>flatness</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>rolloff at 85%</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>rolloff at 95%</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>brightness: high frequency energy</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>roughness: sensory dissonance</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>irregularity: the degree of variation of successive peaks</td>
<td>1</td>
</tr>
<tr>
<td>spectral</td>
<td>zero crossing rate</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>spectral flux</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>MFCC</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>delta MFCC</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>delta-delta MFCC</td>
<td>13</td>
</tr>
<tr>
<td>timbre</td>
<td>key clarity</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>key mode possibility</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>HCDF</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>chroma peak</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>chroma centroid</td>
<td>1</td>
</tr>
<tr>
<td>tonal</td>
<td>chroma</td>
<td>12</td>
</tr>
</tbody>
</table>

**B. GMM Posterior Representation**

This step converts the frame-based feature vectors of a song, which can be of different lengths, into a fixed-dimensional vector representation of the song. In the CBA approach, this is achieved by assigning each feature vector to the closest cluster prototype (codeword), and then representing the song by a vector of the counts of pre-trained codewords, i.e., a bag of codewords representation. Our goal is to find a more accurate prototype for each cluster and estimate a better generative model to represent the observed music features from the real-valued space. The Gaussian mixture model (GMM) fits this requirement very well.

Hereafter, we assume that a collection of universal music feature vectors is modeled by a large GMM, called the Universal Music Features Model (UMFM). The likelihood of a music feature vector \( \mathbf{x} \) is defined as

\[
p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k),
\]

where \( \pi_k \), \( \mu_k \), and \( \Sigma_k \) are, respectively, the mixture weight, mean vector, and covariance matrix of the \( k \)-th Gaussian mixture component. By considering a “latent music class” as a latent variable \( z_k \in \{z_1, z_2, \ldots, z_K\} \) that corresponds to the \( k \)-th Gaussian mixture component, we obtain the conditional probability as follows:

\[
p(\mathbf{x} | z_k = 1) = \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k).
\]

To train the UMFM, we first normalize the feature vectors in each dimension to mean 0 and standard deviation 1. Then, the UMFM is fitted by using the expectation and maximization (EM) algorithm to maximize the total likelihood of the feature vectors generated by the model with respect to the mixture weights, the mean vectors, and the covariance matrices.

With the UMFM, we can estimate the likelihood that a given feature vector belongs to a latent music class \( z_k \) by the posterior probability of the class:

\[
p(z_k = 1 | \mathbf{x}) = \frac{\pi_k p(\mathbf{x} | z_k = 1)}{\sum_{k=1}^{K} \pi_k p(\mathbf{x} | z_k = 1)}. \tag{3}
\]

By assuming that each frame of a given song, \( s_j \), contributes equally to the song, the posterior probability of a certain latent music class can be computed by

\[
p(z_k = 1 | s_j) = \frac{1}{N_j} \sum_{i=1}^{N_j} p(z_k = 1 | \mathbf{x}_{i}), \tag{4}
\]

where \( \mathbf{x}_{i} \) is the feature vector of the \( i \)-th frame of \( s_j \) and \( N_j \) is the number of frames in \( s_j \). The posterior representation of a song in Eq. (4) can be modeled by the generative process of the Bernoulli mixture model, as will be described later.

**III. BERNOULLI MIXTURE MODEL**

The Bernoulli mixture model (BMM), also known as latent class analysis [6], is described in detail in [7]. Here, we explain how to apply BMM to the music tagging problem through a generative process and model inference. We also describe how to incorporate the counts of tags assigned to the songs in BMM training. Assume that we have a music corpus with \( J \) songs, each denoted by \( s_j \), \( j=1, \ldots, J \), and let each song’s tag count \( c_{jw}, \ w=1, \ldots, W \), be a positive integer representing the number of times that tag \( t_w \) has been assigned to song \( s_j \). The binary random variable \( y_{jw} \), with \( y_{jw} \in \{0, 1\} \), determines whether or not tag \( t_w \) applies to song \( s_j \).
A. Generative Process

The generative process of BMM involves two steps. First, the probability of the latent class \( z_{kw} \), \( k=1,\ldots,K \), is chosen from song \( s_j \)'s class weight vector \( \theta_j \):

\[
p(z_{kw} = 1 | \theta_j) = \theta_{jk},
\]

where \( \theta_{jk} \) is the weight of the \( k \)-th latent class. Then, a case of the discrete variable \( y_{jw} \) is selected based on the following conditional probabilities:

\[
\begin{align*}
p(y_{jw} = 1 | z_{kw} = 1, \beta) &= \beta_{kw} \\
p(y_{jw} = 0 | z_{kw} = 1, \beta) &= 1 - \beta_{kw},
\end{align*}
\]

where \( \beta \) is a \( K \)-by-\( W \) parameter matrix recording \( \beta_{kw} \) indexed in the \( k \)-th row and the \( w \)-th column. The conditional probability, which models the probability that song \( s_j \) is annotated with tag \( t_{kw} \), is a Bernoulli distribution with input discrete variable \( y_{jw} \) and parameter \( \beta \) for the \( k \)-th class \( z_{kw} \).

The complete joint distribution over \( y \) and \( z \) is described by the model parameter \( \beta \) and weight matrix \( \Theta \) whose row vector is \( \theta_j \) of song \( s_j \):

\[
p(y, z | \beta, \Theta) = \prod_{j=1}^{J} p(y, z_{kw} = 1 | \beta, \theta_j) = \prod_{j=1}^{J} \prod_{w=1}^{W} \theta_{jk}^{y_{jw}} \beta_{kw}.
\]

The marginal log likelihood of the corpus can be expressed as:

\[
\log p(y | \beta, \Theta) = \sum_{j=1}^{J} \sum_{w=1}^{W} \log \left( \sum_{k=1}^{K} \theta_{jk}^{y_{jw}} \beta_{kw} \right).
\]

B. Model Inference by the EM Algorithm

The Bernoulli mixture model can be fitted with respect to parameter \( \beta \) and weight matrix \( \Theta \) by maximum-likelihood (ML) estimation. In this work, we assume that the latent class of BMM corresponds to the latent music class of UMFM described in Sec. II. In other words, the latent classes have been determined by GMM fitting in the music data representation stage. Each song that has been transformed into the posterior weights \( \theta_j = p(z_{kw} = 1 | s_j) \) by Eq. (4) provides observation information to BMM; therefore, we only need to estimate \( \beta \), which corresponds to the probability of a latent music class. We apply the EM algorithm to maximize the log-likelihood in Eq. (8) in the presence of a latent variable \( z \).

In the E-step, given the song-level weight matrix \( \Theta \) and the model parameter \( \beta \), the posterior probability of each latent variable \( z_{kw} \) can be computed by

\[
\begin{align*}
\gamma_j(z_{kw}) &= p(z_{kw} = 1 | \beta, \Theta, y) \\
p(y_{jw} | z_{kw} = 1, \beta) p(z_{kw} = 1 | \theta_j) &= p(y_{jw} | \beta, \theta_j)
\end{align*}
\]

By setting the derivative to zero, we obtain the update rule for \( \beta_{kw} \) as follows,

\[
\beta_{kw} \leftarrow \frac{\sum_{j=1}^{J} \gamma_j(z_{kw}) y_{jw}}{\sum_{j=1}^{J} \gamma_j(z_{kw})}.
\]

C. Predicting Tags of New Songs

An untagged song \( s \) is first transformed into a sequence of music feature vectors, after which the posterior weight vector \( \theta \) is computed from the pre-trained UMFM. Finally, the score for song \( s \) to be annotated with tag \( t_w \) is computed as the
conditional probability of $y_{sw}=1$ given $\theta$ and $\beta$, which is the model parameter of BMM:

$$p(y_{sw}=1|\beta, \theta) = \sum_{k=1}^{K} \theta_k p(y_{sw}=1|\beta_k) = \sum_{k=1}^{K} \theta_k \beta_{sw}. \quad (14)$$

The scores associated with the pre-defined tags provide a reference for annotating a new song by following the ranking of the conditional probabilities of all tags, or by retrieving songs in decreasing order of the conditional probabilities of the songs associated with a particular tag.

For the annotation task, the $M$ most likely tags are selected by ranking the tag scores for a new song, where $M$ is rounded off based on the average number of tags for a song in the training corpus. In the retrieval task, the relations between the scores and the binary classes (either relevant or irrelevant to a tag) are expressed by a quadratic decision function with the Mahalanobis metric [8]. The corresponding coefficients can be determined by performing an inside validation test on the training music corpus. However, we found that, if a song has several tags, the scores of the top tags are usually lower than those of a song with a few tags. Therefore, the scores of a particular tag across different songs might not be comparable. To solve this problem, we normalize the scores of different tags for a song by subtracting $d$ times their mean, where $d$ denotes the “diversity factor” [3]. However, the normalization process may yield negative values for some tags. In such cases, at the end of the normalization process, we set the negative values to positive values smaller than the smallest positive value without changing their order. Then, it is straightforward to set a pruning threshold on the normalized scores for a specific tag across different songs in the retrieval process. Since this method maintains the ranking order of tags for a song, it does not affect the tagging result of a song.

Although the GMM posterior representation of a song may lose some information about the song, the entire approach is very efficient. The computational cost only depends on the number of latent classes $K$ and the length of the given song. If we ignore the feature extraction part, the tag-prediction procedure is almost performed on the fly. Therefore, our approach provides an effective solution for an on-line music tagging system.

IV. EXPERIMENTS

A. Datasets

We evaluate the proposed approach on the MajorMiner and CAL-500 datasets.

The MajorMiner dataset is taken from the MajorMiner website\(^3\), which uses a game to gather informative free text labels for music. Each player labels randomly given music clips (each about 10 seconds long) by listening to them without any meta-information. If two players assign the same text label to a particular music clip, the label is adopted by the system. Hence, each music clip’s tag count is at least 2. Following to the audio tag classification task in the MIREX competition, 45 popular tags are considered in the tag set. We download all the music clips associated with those 45 tags from the MajorMiner website. The resulting dataset contains 2,441 music clips. In the dataset, the count of a tag given to a music clip is at most 12. The 45 tags in the MajorMiner dataset are listed in TABLE II.

### TABLE II

| THE 45 TAGS IN THE MAJORMINER DATASET |

<table>
<thead>
<tr>
<th>metal</th>
<th>instrumental</th>
<th>horns</th>
<th>piano</th>
<th>guitar</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambient</td>
<td>saxophone</td>
<td>house</td>
<td>loud</td>
<td>bass</td>
</tr>
<tr>
<td>fast</td>
<td>keyboard</td>
<td>electronic</td>
<td>noise</td>
<td>british</td>
</tr>
<tr>
<td>solo</td>
<td>electronica</td>
<td>beat</td>
<td>80s</td>
<td>dance</td>
</tr>
<tr>
<td>jazz</td>
<td>drum machine</td>
<td>strings</td>
<td>pop</td>
<td>r&amp;b</td>
</tr>
<tr>
<td>female</td>
<td>rock</td>
<td>voice</td>
<td>rap</td>
<td>male</td>
</tr>
<tr>
<td>slow</td>
<td>vocal</td>
<td>quiet</td>
<td>techno</td>
<td>drum</td>
</tr>
<tr>
<td>funk</td>
<td>acoustic</td>
<td>distortion</td>
<td>organ</td>
<td>soft</td>
</tr>
<tr>
<td>country</td>
<td>hip hop</td>
<td>synth</td>
<td>trumpet</td>
<td>punk</td>
</tr>
</tbody>
</table>

CAL-500 is a music corpus comprised of 500 clips of popular Western songs [1]. The length of the clips ranges from 3 seconds to more than 22 minutes. Each clip has been manually labeled by at least three humans based on 135 predefined musically-relevant concepts, resulting in 174 distinct text labels. In this work, the “soft-assignment” annotation labels with a multi-level degree in the range [0, 1] are transformed into positive integer counts ranging from 1 to 14. We extract our own music features from the clips instead of using the MFCC features provided in the corpus.

B. Model Training

For each music clip in the training data, we extract a sequence of 70-dimensional feature vectors from a 16-bit, 22kHz, mono music waveform with a 50ms frame size and 0.5 hop shift. To train the UMFM used in the proposed approach and the codebook used in the CBA approach, we randomly select 25% of the feature vectors from each clip. There are approximately 235,000 vectors for the MajorMiner dataset and 821,000 vectors for the CAL-500 dataset. The mixture number and the codebook size are set between $K=128$ and $K=2,048$ for the MajorMiner dataset and between $K=128$ and $K=1,024$ for the CAL-500 dataset. We do not consider the $K=2,048$ case for the CAL-500 dataset due to the high computational overhead. To train the 70-dimensional UMFM with 20 iterations on the $K$-means-derived seeds, we utilize the HTK 3.4 toolkit [9]. The codebook is trained by using the Matlab software with the stopping criterion that the total distance is reduced by less than 0.0001 compared to that of the previous iteration. With the UMFM and the codebook, we can estimate two kinds of music data representation for a song, namely GMM posterior representation and vector quantized representation.

Using the posterior weights estimated by Eq. (4) for each training clip, we train the BMM with an initial matrix $\beta$, in which all entries are set to 0.5. By iteratively updating $\beta_{sw}$ using Eq. (13), the training process stops when the increase

\(^{3}\) http://majorminer.org/
ratio of the log-likelihood in Eq. (8) is less than 0.0001. This method is denoted as PWBMM-II. We can apply the same posterior weights in the CBA approach. The BMM is trained by using Eq. (12) without considering the tag counts. This method is called PWBMM-I.

C. Experimental Setup

The experiments evaluate the proposed approach in terms of two tasks: annotation and retrieval. We perform three-fold evaluation on the MajorMiner dataset based on the MIREX audio tag classification evaluation setup, and ten-fold evaluation on the CAL-500 dataset following the work in [3]. In other words, there are three runs for the MajorMiner dataset, each using two folds for training and one fold for testing; and ten runs for the CAL-500 dataset, each using nine folds for training and one fold for testing. Each dataset is divided into folds randomly.

In the annotation task, the number of tags assigned to a clip is the same as the average number of tags assigned to the clip in the training data. Under our implementation, the system selects the top 5 tags for a clip in the MajorMiner dataset and the top 34 tags for a clip in the CAL-500 dataset. The performance is evaluated in terms of precision rate, recall rate, and F-measure, as well as the area under the ROC curve per clip (Clip AUC).

In the retrieval task, given a tag, we first apply heuristic normalization (described in Sec. III) to the scores of each clip. Here, d is set at 1.5. Then, we apply the pre-trained quadratic decision function with the Mahalanobis metric on the normalized scores to retrieve the relevant clips. In the CBA method, given a tag \( t_n \), the test clips are ranked according to 
\[
 p( y_n = 1 \mid \beta, \theta )
\]
directly. The performance is evaluated in terms of F-measure and AUC per tag (Tag AUC). Note that, in the CAL-500 dataset, sometimes no clips should be retrieved for a tag according to the ground truth label (i.e., no clip in the test fold is associated with the tag) due to the large and diverse nature of the tag set. We exclude these tags because we cannot calculate the AUC for them.

D. Experimental Results and Discussions

We evaluate three approaches in the experiments: the CBA approach, the PWBMM-I approach and the PWBMM-II approach. The CBA approach is taken as the baseline. The results of experiments on the MajorMiner and CAL-500 datasets in terms of the mean and standard deviation (in parentheses) of the performance over multiple runs are summarized in Table III and Table IV, respectively. The values in bold font represent the best performance of each evaluation metric.

The experiment results suggest that the proposed PWBMM-based methods tend to perform better in annotation tasks than in retrieval tasks. This is because the methods were originally designed for annotation only. To apply them in the retrieval task, we adopt a simple normalization strategy to manipulate the scores and a simple decision function for thresholding. Although the two methods are not good enough, their performance is comparable to that of the methods evaluated on the MajorMiner dataset at MIREX 2009.

For the annotation task, the PWGMM-I approach, which uses the GMM posterior weights, outperforms the CBA approach, while PWGMM-II, which also considers the tag counts in training, outperforms PWGMM-I. The results verify that using GMM posterior representation and considering the tag counts of the training clips in PWBMM training improve the performance.

For the PWBMM-I and PWBMM-II approaches, increasing \( K \) does not yield a significant improvement, in contrast to the CBA approach. This is because GMM-based music data representation is more capable of representing the music feature space than codebook-based music data representation. As a result, the feature modeling ability of a 128-mixture UMFM is as good that of the 1,024-centroid codebook in terms of the Clip AUC metric.

The CBA approach outperforms the PWBMM-I approach in terms of tag precision on the MajorMiner dataset. A clip in the MajorMiner dataset (~10 seconds) is much shorter than a clip in the CAL-500 dataset (usually >1 minute). Since short clips can not express many scenarios and they are usually consistent in only a few musical concepts, their music features may show strong and obvious characteristics that really match the capability of vector quantization. In other words, sometimes a short clip can be well represented by only few codewords. However, the good performance of the PWBMM-I approach on the CAL-500 dataset indicates that a long clip prefers a more generative model. This is because a long clip contains abundant information that needs a softer representation to maintain its entropy within a fixed-dimensional vector. We have not observed the convergence of performance with respect to an increase in \( K \), which means the performance may continue to improve as \( K \) increases.

V. Conclusions

In this paper, we have proposed a new music tag annotation and retrieval approach based on the posterior weighted Bernoulli mixture model. The approach uses GMM posterior representation of music feature vectors and incorporates the tag counts of training songs when training the Bernoulli mixture model. The results of experiments on the MajorMiner and CAL-500 datasets demonstrate that our approach outperforms the CBA approach in terms of different evaluation metrics.

The current implementation gives equal weight to each frame in a song when constructing the song-level GMM posterior representation. We believe that short-time fixed-length frames do not contribute equally to a specific tag. For long music clips in particular, different segments, such as the intro, verse, chorus, and bridge, may provide different information to trigger an individual’s perception of the music. Through automatic music structure analysis and segmentation, we may find a better way to represent a song by giving higher weights to those music hooks. In our future work, we will also

---

consider more temporal music characteristics to improve the GMM posterior representation.

ACKNOWLEDGEMENTS

This work was partially supported by the Taiwan e-Learning and Digital Archives Program (TELDAP), which is sponsored by the National Science Council of Taiwan under Grant: NSC99-2631-H-001-020. The authors wish to thank the anonymous reviewers for their valuable comments and suggestions, which helped improve the quality of the paper.

REFERENCES


### Table III
THE RESULTS OF EXPERIMENTS ON THE MAJORMINER DATASET.

<table>
<thead>
<tr>
<th>Model Name</th>
<th>$K$</th>
<th>Clip F-Measure</th>
<th>Precision</th>
<th>Recall</th>
<th>Clip AUC</th>
<th>Tag F-Measure</th>
<th>Tag AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBA</td>
<td>128</td>
<td>0.4415 (0.0052)</td>
<td>0.4391</td>
<td>0.4458</td>
<td>0.8476 (0.0005)</td>
<td>0.1450 (0.0023)</td>
<td>0.7452 (0.0043)</td>
</tr>
<tr>
<td></td>
<td>256</td>
<td>0.4516 (0.0113)</td>
<td>0.4498</td>
<td>0.4557</td>
<td>0.8532 (0.0038)</td>
<td>0.1586 (0.0066)</td>
<td>0.7491 (0.0143)</td>
</tr>
<tr>
<td></td>
<td>512</td>
<td>0.4596 (0.0159)</td>
<td>0.4555</td>
<td>0.4600</td>
<td>0.8532 (0.0070)</td>
<td>0.1741 (0.0088)</td>
<td>0.7558 (0.0065)</td>
</tr>
<tr>
<td></td>
<td>1,024</td>
<td>0.4660 (0.0116)</td>
<td>0.4646</td>
<td>0.4705</td>
<td>0.8597 (0.0017)</td>
<td>0.1734 (0.0051)</td>
<td>0.7555 (0.0061)</td>
</tr>
<tr>
<td></td>
<td>2,048</td>
<td>0.4686 (0.0072)</td>
<td>0.4708</td>
<td>0.4783</td>
<td>0.8606 (0.0039)</td>
<td>0.1806 (0.0098)</td>
<td>0.7536 (0.0033)</td>
</tr>
</tbody>
</table>

### Table IV
THE RESULTS OF EXPERIMENTS ON THE CAL-500 DATASET.

<table>
<thead>
<tr>
<th>Model Name</th>
<th>$K$</th>
<th>Clip F-Measure</th>
<th>Precision</th>
<th>Recall</th>
<th>Clip AUC</th>
<th>Tag F-Measure</th>
<th>Tag AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBA</td>
<td>128</td>
<td>0.4576 (0.0054)</td>
<td>0.4116</td>
<td>0.5174</td>
<td>0.8609 (0.0014)</td>
<td>0.2703 (0.0069)</td>
<td>0.7494 (0.0038)</td>
</tr>
<tr>
<td></td>
<td>256</td>
<td>0.4614 (0.0078)</td>
<td>0.4153</td>
<td>0.5210</td>
<td>0.8629 (0.0097)</td>
<td>0.2780 (0.0049)</td>
<td>0.7501 (0.0076)</td>
</tr>
<tr>
<td></td>
<td>512</td>
<td>0.4663 (0.0075)</td>
<td>0.4205</td>
<td>0.5271</td>
<td>0.8662 (0.0050)</td>
<td>0.2838 (0.0035)</td>
<td>0.7505 (0.0035)</td>
</tr>
<tr>
<td></td>
<td>1,024</td>
<td>0.4690 (0.0095)</td>
<td>0.4230</td>
<td>0.5294</td>
<td>0.8680 (0.0043)</td>
<td>0.2869 (0.0055)</td>
<td>0.7536 (0.0087)</td>
</tr>
<tr>
<td></td>
<td>2,048</td>
<td>0.4722 (0.0131)</td>
<td>0.4340</td>
<td>0.5265</td>
<td>0.8652 (0.0016)</td>
<td>0.2725 (0.0024)</td>
<td>0.7653 (0.0106)</td>
</tr>
</tbody>
</table>

consider more temporal music characteristics to improve the GMM posterior representation.