COMBINING FEATURE KERNELS FOR SEMANTIC MUSIC RETRIEVAL

Luke Barrington **Mehrdad Yazdani** **Douglas Turnbull*** **Gert Lanckriet**
Electrical & Computer Engineering, *Computer Science & Engineering
University of California, San Diego
lbarrington@ucsd.edu myazdani@ucsd.edu dturnbul@cs.ucsd.edu gert@ece.ucsd.edu

ABSTRACT

We apply a new machine learning tool, *kernel combination*, to the task of semantic music retrieval. We use 4 different types of *acoustic content* and *social context* feature sets to describe a large music corpus and derive 4 individual kernel matrices from these feature sets. Each kernel is used to train a support vector machine (SVM) classifier for each semantic tag (e.g., 'aggressive', 'classic rock', 'distorted electric guitar') in a large tag vocabulary. We examine the individual performance of each feature kernel and then show how to learn an optimal linear combination of these kernels using convex optimization. We find that the retrieval performance of the SVMs trained using the combined kernel is superior to SVMs trained using the best individual kernel for a large number of tags. In addition, the weights placed on individual kernels in the linear combination reflect the relative importance of each feature set when predicting a tag.

1 INTRODUCTION

A significant amount of music information retrieval (MIR) research has focused on signal processing techniques that extract features from *audio content*. Often, these feature sets are designed to reflect different aspects of music such as timbre, harmony, melody and rhythm [15, 9]. In addition, researchers use data from online sources that places music in a *social context* [6, 16]. While individual sets of audio content and social context features have been shown to be useful for various MIR tasks (e.g., classification, similarity, recommendation), it is unclear how to combine multiple feature sets for these tasks.

This paper presents a principled approach to combining multiple feature sets by applying a useful machine learning technique: *kernel combination* [7]. A *kernel function* for each feature set measures the similarity between pairs of songs. Kernel functions can be designed to accommodate heterogeneous feature representations such as vectors [8], sets of vectors [4], distributions [10], or graphs [1]. We use the kernel function to compute a *kernel matrix* that encodes the similarity between all pairs of songs. From multiple feature sets, we compute multiple kernel matrices for a set of songs. We find the optimal linear combination of the kernel matrices using quadratically-constrained quadratic-programming (i.e., convex optimization). The result is a single kernel matrix that is used by a support vector machine (SVM) to produce a decision boundary (i.e., a classifier).

In this paper, we use kernel combination to integrate music information from two audio content and two social context feature sets. For each semantic tag (e.g., "aggressive", "classic rock", "distorted electric guitar") in our vocabulary, we learn the linear combination of the four kernel matrices that is optimal for SVM classification and rank-order songs based on their relevance to the tag. The weights that are used to produce the combined kernel reflect the importance of each feature set when producing an optimal classifier.

In the following subsection, we discuss related work on music feature representations, kernel methods in MIR research, and the origins of kernel combination. We continue in Section 2 with an overview of kernel methods and the kernel combination algorithm. Section 3 introduces a number of features that capture information about musical content and context and describes how to build song-similarity kernels from these features. Section 4 describes our experiments using individual and combined heterogenous feature kernels for semantic music retrieval. Section 5 concludes with a discussion of our findings.

1.1 Related Work

McKinney and Breebaart [9] use 4 different feature sets to represent audio content and evaluate their individual performance on general audio classification and 7-way genre classification. They determine that features based on the temporal modulation envelope of low-level spectral features are most useful for these tasks but suggest that intelligent combinations of features might improve performance. Tzanetakis and Cook [15] present a number of content-based features and concatenate them to produce a single vector to represent each song for use in genre classification.

Knees et al. [6] use semantic data mined from the results of web-searches for songs, albums and artists to generate a *contextual* description of the music based on large-scale, social input, rather than features that describe the audio *content*. Using this context data alone achieves retrieval results comparable to the best content-based methods. Whitman and Ellis [16] also leverage web-mined record reviews to develop an unbiased music annotation system.

Flexer et al. [3] combine information from two feature sources: tempo and MFCCs. They use a nearest-neighbor classifier on each feature space to determine an independent class-conditional probability for each genre, given each feature set. Using a naïve Bayesian combination, they multiply these two probabilities and find that the resulting probability improves 8-way genre classification of dance music.

Lanckriet et al. [7] propose a more sophisticated method for combining information from multiple feature sources. They use a kernel matrix for *each* feature set to summarize similarity between data points and demonstrate that it is possible to learn a linear combination of these kernels that optimizes performance on a discriminative classification task. They show that, for a protein classification task, an optimal combination of heterogenous feature kernels performs better than any individual feature kernel.

Discriminative SVM classifiers have been successfully applied to many MIR tasks. Mandel and Ellis [8] use patterns in the mean and co-variance of a song's MFCC features to detect artists. Meng and Shawe-Taylor [10] use a multivariate autoregressive model to describe songs. Using Jebara et al.'s probability product kernels [5], they kernelize the information contained in these generative models and then use an SVM to classify songs into 11 genres.

2 KERNEL METHODS

A simple solution for binary classification problems is to find a linear discriminant function that separates the two classes in the data vector space. In practice, the data may not lie in a valid vector space or may not be linearly separable. A linear discriminant may still be learned using a kernel function to map the data items into a new kernel-induced vector (Hilbert) space, called the feature space *F*.

If we consider mapping a pair of data items $\mathbf{x}_i, \mathbf{x}_j \in \Omega$ with $\Phi : \Omega \rightarrow F$, kernel methods only require the inner product of the data features $\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \in F$. The kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$ completely specifies this inner product relationship. Thus, a data set $\{\mathbf{x}_i, i \in 1, ..., n\}$ can be specified in the feature space with the kernel matrix $\mathbf{K}$ where each element (i, j) is obtained by the kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$. See [12] for details.

If we use a valid kernel function, the resulting kernel matrix $\mathbf{K}$ must be positive semi-definite. Henceforth, kernel matrix and kernel function will be referred to as "kernel" with the meaning being clear from the context. Many different kernels are used in practice and we have experimented with several of them. A natural question that arises is which kernel to use. Our work and the work of Lanckriet et al. [7] suggests that settling for a single kernel may not be optimal and that a combination of kernels can improve results.

2.1 Kernelizing Feature Data

2.1.1 Radial Basis Function

The Radial Basis Function (RBF) kernel is one of the most commonly used kernels. It is defined as:

$$\mathbf{K}_{i,j} = \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}).$$

The hyper-parameter σ is learned by cross validation [12].

2.1.2 Probability Product Kernel

The Probability Product Kernel (PPK) is a natural candidate for data items that have been modeled with probability distributions [5]. We have found experimentally that the PPK tends to outperform other distribution-capturing kernels (such as the Kullback-Leibler divergence kernel space[11]), and is easier and more elegant to compute. We model each data item $\mathbf{x}_i$ with a distribution $p(\mathbf{x}; \theta_i)$ specified by parameters θ_i and compute the PPK as:

$$\mathbf{K}_{i,j} = \int p(\mathbf{x}; \theta_i)^\rho p(\mathbf{x}; \theta_j)^\rho d\mathbf{x},$$

where $\rho > 0$. When $\rho = 1/2$, the PPK corresponds to the Bhattacharyya distance between two distributions. This case has a geometric interpretation similar to the inner-product between two vectors. That is, the distance measures the cosine of the angle between the two distributions. Another advantage of the PPK is that closed-form solutions for Gaussian mixture models (GMM's) are available [5], whereas this is not the case for the Kullback-Leibler divergence.

2.1.3 Normalized Kernels

It can also be useful to "normalize" the kernel matrix. This projects each entry in the kernel matrix onto the unit sphere and entries can be geometrically interpreted as the cosine of the angle between the two data vectors. Furthermore, when combining multiple kernels, normalized kernels will become "equally" important. We normalize $\mathbf{K}$ by setting:

$$\mathbf{K}_{i,j}^{norm} = \frac{\mathbf{K}_{i,j}}{\sqrt{\mathbf{K}_{i,i}\mathbf{K}_{j,j}}}.$$

The resulting kernel matrix has all ones on its diagonal, helping ensure numerical stability in the combination algorithm below. Henceforth all kernels are assumed to be normalized.

2.2 Support Vector Machine (SVM)

The SVM learns the separating hyperplane that maximizes the margin between two classes. We present the SVM here to establish notation and clarify the formulation of kernel combination. The algorithm considered is the 1-norm soft margin SVM which can be cast as the following optimization problem:

$$\min_{\mathbf{w},b,\zeta\succeq 0} \mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\zeta_i$$
$$\text{subject to: } y_i(\mathbf{w}^T\Phi(\mathbf{x}_i)+b) \geq 1-\zeta_i, \forall i, \tag{1}$$

where $\mathbf{x}_i$ is the data point we wish to classify and $y_i \in \{+1,-1\}$ is its corresponding label. $\Phi(\mathbf{x}_i)$ is the data point mapped to the feature space. $\mathbf{w}$ is the normal vector to the hyperplane that defines the discriminant boundary in the feature space and b is the hyperplane offset. C is a regularization parameter that penalizes misclassifications or within-margin points and ζ_i are slack variables that "relax" the hard-margin constraints of the SVM.

The the dual formulation of 1-norm soft margin optimization in Equation 1 is:

$$\max_{0\leq\alpha\leq C,\alpha^T\mathbf{y}=0} 2\alpha^T\mathbf{e} - \alpha^T\text{diag}(\mathbf{y})\mathbf{K}\text{diag}(\mathbf{y})\alpha, \tag{2}$$

where $\mathbf{e}$ is an n-vector of ones and $\alpha \in \mathcal{R}^n$ is the vector of Lagrange multipliers. The solution to this dual problem is a function of K and results from the point-wise maximum of a series of affine functions in alpha. Because the point-wise maximum of affine functions is convex, the solution to the dual problem is convex in K.

The SVM classifies a new, unlabeled data point $\mathbf{x}_{new}$ based on the sign of the linear decision function:

$$f(\mathbf{x}_{new}) = \mathbf{w}^T\Phi(\mathbf{x}_i) + b = \sum_{i=1}^{n}\alpha_i\mathbf{K}(\mathbf{x}_i,\mathbf{x}_{new}) + b. \tag{3}$$

For example, the classification task might be to determine whether a given song represents the tag "guitar" or not. It is common to associate positive examples (e.g. those classified as "guitar") with the label "+1" and negative examples with "-1." This labeling is achieved by taking the sign of the decision function $sign(f(\mathbf{x}))$.

2.3 Kernel Combination SVM

Lanckriet et al. [7] propose a linear combination of m different kernels, each encoding different features of the data:

$$\mathbf{K} = \sum_{i=1}^{m}\mu_i\mathbf{K}_i,$$

where $\mathbf{K}_i$ are the individual kernels formulated via feature extraction methods described in Section 3. A useful property of kernel matrices is that, since they are positive semi-definite, a positive linear combination of a set of kernels will also be a positive semi-definite kernel [12]. That is,

$$\mathbf{K} = \sum_i \mu_i\mathbf{K}_i,\ \mu_i > 0,\ \mathbf{K}_i \succeq 0 \quad \forall i \quad \Rightarrow \quad \mathbf{K}\succeq 0.$$

The kernel combination problem reduces to learning the set of weights, μ, that combine the feature kernels, $\mathbf{K}_i$, into the "optimum" kernel. To ensure the positive semi-definiteness of $\mathbf{K}$, the weights are restricted to be positive, $\mu_i \geq 0$.

We wish to ensure that the weights μ sum to one. This prevents any weight from growing too large, forces the kernel matrices to be combined convexly and gives interpretability to the relative importance of each kernel. We can achieve this with the constraint $\mu^T\mathbf{e} = 1$. The optimum value of the dual problem in Equation 2 is inversely proportional to the margin and is convex in $\mathbf{K}$. Thus, the optimum $\mathbf{K}$ can be learned by minimizing the function that optimizes the dual (thereby maximizing the margin) with respect to the kernel weights, μ.

$$\min_{\mu}\left\{\max_{0\leq\alpha\leq C,\alpha^T\mathbf{y}=0} 2\alpha^T\mathbf{e} - \alpha^T\text{diag}(\mathbf{y})\mathbf{K}\text{diag}(\mathbf{y})\alpha\right\}$$
$$\text{subject to: } \mu^T\mathbf{e} = 1$$
$$\mu_i \geq 0 \quad \forall i = 1,...,m, \tag{4}$$

where $\mathbf{K} = \sum_{i=1}^{m}\mu_i\mathbf{K}_i$.

Since the objective function is linear in μ, and hence convex, we can write this min-max problem equivalently as a max-min problem [2]. We may recast this problem in epigraph form and deduce the following quadratically constrained quadratic program (QCQP)[7]:

$$\max_{\alpha,t} 2\alpha^T\mathbf{e} - t$$
$$\text{subject to: } t \leq \alpha^T\text{diag}(\mathbf{y})\mathbf{K}_i\text{diag}(\mathbf{y})\alpha, i = 1,\ldots,m \tag{5}$$
$$0 \leq \alpha \leq C, \alpha^T\mathbf{y} = 0.$$

The solution to this QCQP will return the optimum convex combination of kernel matrices. For a more complete derivation of the kernel combination algorithm, see [7].

3 KERNEL CONSTRUCTION FROM AUDIO FEATURES

We experiment with a number of popular MIR feature sets which attempt to represent different aspects of music: timbre, harmony and social context.

3.1 Audio Content Features

Audio content features are extracted directly from the audio track. Each track is represented as a set of feature vectors, $\mathcal{X} = \{\mathbf{x_1},...,\mathbf{x_T}\}$, where each feature vector $\mathbf{x_t}$ represents an audio segment, and T depends on the length of the song. We integrate the set of feature vectors for a song into a single representation by estimating the parameters of a probability distribution, approximated with a Gaussian mixture model, over the audio feature space. Finally, we compute a kernel matrix using set of song GMMs with the probability product kernel technique [5] described in Section 2.1.2.

3.1.1 Mel-Frequency Cepstral Coefficients

Mel-frequency cepstral coefficients (MFCCs) are a popular feature for a number of music information retrieval tasks [8, 10, 3]. For a 22050Hz-sampled, monaural song, we compute the first 13 MFCCs for each half-overlapping short-time ($\sim$23 msec) segment and append the first and second instantaneous derivatives of each MFCC. This results in about 5,000 39-dimensional **MFCC+delta** feature vectors per 30 seconds of audio content.

We summarize an entire song by modeling the distribution of its MFCC+delta features with an 8-component GMM. We sample 5,000 feature vectors from random times throughout a song, learn a MFCC+delta GMM for each song and convert the GMM parameters of every song into a probability product kernel matrix. For our semantic music retrieval task, we find that using GMMs with diagonal covariances results in better performance than using single, full-covariance Gaussian distributions.

3.1.2 Chroma

Chroma features [4] represent of the harmonic content of a short-time window of audio by computing the spectral energy present at frequencies that correspond to each of the 12 notes in a standard chromatic scale. We extract a 12-dimensional chroma feature every $\frac{1}{4}$ second and, as with the MFCCs above, model the distribution of a song's chroma features with a GMM and create a probability product kernel matrix. We also investigated features that describe a song's key (estimated as the mean chroma vector) and tempo (in beats per minute) but found that these features performed poorly for semantic music retrieval.

3.2 Social Context Features

We can also summarize each song in our dataset with a *semantic* feature vector. Each element of this vector indicates the relative strength of association between the song and a semantic tag. We propose two methods for collecting this semantic information: social tags and web-mined tags. Given these semantic feature vectors, we compute an RBF kernel for each data source as described in Section 2.1.1. The semantic feature vectors tend to be *sparse* as most songs are annotated with only a few tags. When tags are missing for a song, we set that dimension of the semantic feature to zero. In addition to being sparse, many semantic context features are *noisy* and do not always reflect an accurate semantic relationship between a song and a tag. Songs which have not been annotated with any tags are assigned the average semantic feature vector (i.e., the estimated prior probabilities of the tags in the vocabulary).

3.2.1 Social Tags

For each song in our dataset, we attempt to collect two lists of social (raw-text) tags from the Last.fm website (www.last.fm). The first list relates the *song* to a set of tags where each tag has a score that ranges from 0 (low) to 100 (high) as a function of both the number and diversity of users who have annotated that song with the tag. The second list associates the *artist* with tags and aggregates the tag scores for all the songs by that artist. We find the scores for all relevant song and artist tags as well as their synonyms (determined by the authors). For example, a song is considered to be annotated with "down tempo" if it has instead been annotated with "slow beat". We also allow wildcard matches for tags so that, for example, "blues" matches with "delta electric blues" and "rhythm & blues". To create the Last.fm semantic feature vector, we add the song and artist tag scores into a single vector.

3.2.2 Web-Mined Documents

We extract tags from a corpus of web documents using the relevance scoring (RS) algorithm [6]. To generate tags for a set of songs, RS works by first repeatedly querying a search engine with each song title, artist name and album title to obtain a large corpus of relevant web-documents. We restrict the search to a set of musically-relevant sites (see [13] for details). From these queries, we retain the (many-to-many) mappings between the songs and the documents. Then we use each tag as a query string to find all the relevant documents from our corpus, each with an associated relevance weight. By summing the relevance weights for the documents associated with a song, we calculate a score for each song-tag pair. The semantic feature vector for a song is then the vector of song-tag scores for all tags in our vocabulary.

4 SEMANTIC MUSIC RETRIEVAL EXPERIMENTS

We experiment on the CAL-500 data set [14]: 500 songs by 500 unique artists, each annotated by a minimum of 3 individuals using a 174-tag vocabulary representing genres, instruments, emotions and other musically relevant concepts. The kernel combination algorithm requires at least 50 data points (and often more) from each class to learn a reliable classifier so, for the experiments reported here, we require that each tag be associated with at least 50 songs and remove some redundant tags, reducing the vocabulary to 61 tags [1].

Given a tag (e.g., "jazz"), the goal is to rank all songs by their relevance to this query (e.g. jazz songs at the top). We learn a decision boundary for each tag (e.g., a boundary between jazz / not jazz) and rank all test songs by their distance (positive or negative) from this boundary, calculated using Equation 3. The songs which most strongly embody the query tag should have a large positive distance from

[1] For the list of tags used, see http://cosmal.ucsd.edu/cal

the boundary. Conversely, less semantically relevant songs should have a small or negative distance from the boundary.

We compare the SVM results to the human-annotated labels provided in the CAL-500 dataset where a song is positively associated with a tag if 80% of test subjects agree that the tag is relevant. We quantify the ranking returned by the discriminative SVM classifier using the area under the receiver operating characteristic (ROC) curve. The ROC compares the rate of correct detections to false alarms at each point in the ranking. A perfect ranking (i.e., all the relevant songs at the top) results in an ROC area equal to one. Ranking songs randomly, we expect the ROC area to be 0.5. We also compute mean average precision (MAP) by moving down the ranked list of test songs and averaging precisions at every point where we correctly identify a new song.

One benefit of using ROC area as a metric to evaluate rankings is that it is immune to differences in the tags' prior probabilities. The tag frequencies in the data set roughly follow an exponential distribution with most terms having far more negative than positive examples. The average prior probability over all tags is 16.9%. While measuring binary classification performance would have to overcome a bias towards the negative class, evaluating ranking performance (using ROC area) is a better measure of what's important for real MIR applications and does not suffer from this imbalance. For example, 339 of the 500 songs were annotated as having "Male Lead Vocals" while only 56 songs were judged to be from the "Electronica" genre. In the latter case, a classifier could achieve 88% accuracy by never labeling any songs as "Electronica" while its average ROC area would be 0.5 (random).

4.1 Single Kernel Results

For each of the four features described in Section 3, we construct a kernel and train a one-vs-all SVM classifier for each tag where the negative examples are all songs not labeled with that tag. We train SVMs using 400 songs, optimize regularization parameter, C, using a validation set of 50 songs and use this final model to report results on a test set of 50 songs. The performance of each kernel, averaged over all tags and 10-fold cross validation (so that each song appears in the test set exactly once), is shown in Table 1. The rightmost column of Table 1 shows the number of tags for which each of the different features resulted in the best single kernel for ranking songs. The bottom row of Table 1 shows the theoretical upper bound that an 'oracle' could achieve by choosing the single best feature for each tag, based on test-set performance.

Table 1 demonstrates that the MFCC content features are most useful for about 60% of the tags while social context features are best for the other 40%. However the facts that all kernels individually perform well above random and that three of four perform best for many tags indicate that all features can be useful for semantically ranking songs.

Feature	ROC area	MAP	Best Kernel
MFCC	0.71	0.50	37
Chroma	0.60	0.40	0
Last.fm	0.68	0.49	16
Web Docs	0.67	0.48	8
Single Oracle	0.73	0.54	

Table 1. Individual feature kernel SVM retrieval results and the number of tags for which each kernel was the best.

4.2 Kernel Combination Results

The kernel combination SVM can integrate the four sources of feature information to enhance our music retrieval system. In practice, we find that some tags do not benefit from this integration since they have reached maximum performance with their individual best feature kernel. With appropriate setting of the regularization parameter, C, it is generally possible to get the combined kernel to correctly place all weight on this single best kernel and the result in these cases is the same as for the single best kernel. 31 of the 61 tags (51%) show an improvement in ROC area of 0.01 or more while 14 tags (23%) show no significant change. 16 tags (26%) get worse, we suspect due to insufficient data to train and optimize the parameters of the combined kernel SVM.

Table 2 shows the average retrieval ROC area and MAP results for the combined kernel as well as those that could be obtained if the single best feature were known in advance. We break down results by tag category and show how many tags from each category benefit from kernel combination. Despite the facts that some tags perform worse using kernel combination and that knowing which single feature is best requires looking at the test set, the combined kernel improvement is significant (paired t-test, $N = 61, \alpha = 0.05$).

Though all kernels are normalized, the absolute value of a weight does not give direct insight into the relevance of that feature to the classification task. If its weight is zero (as happens if we include a randomly-generated kernel), the feature is truly useless but even a small positive weight can allow a feature kernel to have a large impact. However, examining the relative feature weights in Figure 1 for the 45 words where the algorithm succeeds does allow us to make some interesting comparisons of how the features' utility varies for different tags. The simplex in Figure 1 shows that the MFCC kernel tends to get most weight, unsurprising since it is the single most reliable feature. We also see a preference for Last.fm over Web Docs and both get more weight than Chroma, in accordance with their individual performances in Table 1. The spectral MFCC features are most useful for genres like "Rock" as well as the ubiquitous "Drum Set" which most web users neglect to tag. The human-derived context information gets more weight for nebulous tags like "Powerful" as well as both "Male" & "Female Lead Vocals" which, while only represented in a few MFCC vectors, are easily identified by human listeners.

Tag Category	Single Oracle ROC	Single Oracle MAP	Kernel Combo ROC	Kernel Combo MAP	Improve / Total
All tags	0.73	0.54	0.74	0.54	31 / 61
Emotion	0.71	0.51	0.72	0.52	14 / 28
Genre	0.80	0.58	0.82	0.58	3 / 5
Instrument	0.74	0.55	0.76	0.56	8 / 10
Song	0.73	0.60	0.73	0.60	5 / 15
Usage	0.70	0.34	0.69	0.35	0 / 1
Vocals	0.69	0.37	0.69	0.39	1 / 2

Table 2. Music retrieval results for 61 CAL500 tags. "Single Oracle" picks the single best feature for each tag, based on test set performance. "Kernel Combo" learns the optimum combination of feature kernels from the training set.

It may be possible to enhance these results by adding even more feature kernels. In practice however, as the number of kernels grows, over-fitting can be a problem. Keeping C low ensures the optimum solution is well regularized. In the experiments reported here, optimum results for both single and combined kernel SVMs are found by optimizing a different value of C for the positive and negative class examples.

5 DISCUSSION

This paper uses a novel machine learning tool to combine a number of existing MIR features in a manner that optimizes retrieval performance for a variety of semantic tags. The results show significant improvement in retrieval performance for 51% of the 61 tags considered. Although discriminative SVM classification may not be the best solution for semantic retrieval problems, the results reported here (ROC area of 0.71 for MFCC+delta, 0.73 for the single best feature, 0.74 for kernel combination) compare favorably to the generative framework in [14] which reports retrieval ROC area of 0.71 on the CAL-500 data using the MFCC+delta features (although that work used a larger vocabulary of 174 tags).

6 ACKNOWLEDGEMENTS

Thanks to Shlomo Dubnov, Lawrence Saul and our reviewers for helpful comments. LB and DT are supported by NSF IGERT fellowship DGE-0333451. We also acknowledge support from NSF grant DMS-MSPA 062540922.

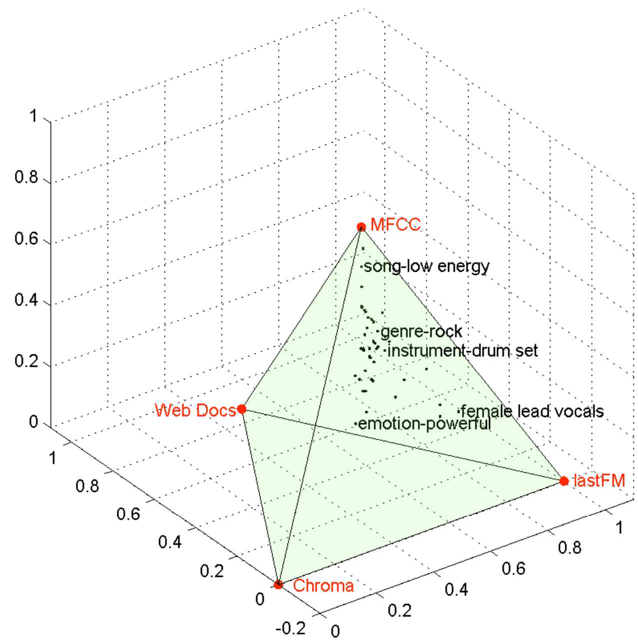

Figure 1. Optimal feature weights for 45 tags from the CAL-500 data set. Five tags are named for illustrative purposes.

7 REFERENCES

[1] A. Berenzweig, B. Logan, D. Ellis, and B. Whitman. A large-scale evalutation of acoustic and subjective music-similarity measures. *Computer Music Journal*, pages 63–76, 2004.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.

[3] A. Flexer, F. Gouyon, S. Dixon, and G. Widmer. Probabilistic combination of features for music classification. *ISMIR*, 2006.

[4] M. Goto. A chorus selection detection method for musical audio singals and its application to a music listening station. *IEEE TASLP*, 2006.

[5] T. Jebara, R. Kondor, and A. Howard. Probability product kernels. *Journal of Machine Learning Research*, 5:819–844, 2004.

[6] P. Knees, T. Pohle, M. Schedl, D. Schnitzer, and K. Seyerlehner. A Document-centered Approach to a Natural Language Music Search Engine. *European Conference on Information Retrieval*, 2008.

[7] G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[8] M.I. Mandel and D.P.W. Ellis. Song-level features and support vector machines for music classification. *ISMIR*, 2005.

[9] M.F. McKinney and J. Breebaart. Features for audio and music classification. *ISMIR*, 2003.

[10] A. Meng and J. Shawe-Taylor. An investigation of feature models for music genre classification using the support vector classifier. *ISMIR*, 2005.

[11] P.J. Moreno, P.P. Ho, and N. Vasconcelos. A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. *NIPS*, 2004.

[12] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, USA, 2004.

[13] D. Turnbull, L. Barrington, and G. Lanckriet. Five approaches to collecting tags for music. *ISMIR*, 2008.

[14] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE TASLP*, 16(2):467–476, February 2008.

[15] G. Tzanetakis and P. Cook. *IEEE Transactions on Speech and Audio Processing*, 10(5), 2002.

[16] B. Whitman and D. Ellis. Automatic record reviews. *ISMIR*, 2004.