USING REGRESSION TO COMBINE DATA SOURCES FOR SEMANTIC MUSIC DISCOVERY

Brian Tomasik, Joon Hee Kim, Margaret Ladlow, Malcolm Augat, Derek Tingle, Richard Wicentowski, Douglas Turnbull
Department of Computer Science, Swarthmore College, Swarthmore PA 19081
{btomasi1, joonhee.kim, mladlow1, maugat1}@alum.swarthmore.edu
{dt, richardw, turnbull}@cs.swarthmore.edu

ABSTRACT

In the process of automatically annotating songs with descriptive labels, multiple types of input information can be used. These include keyword appearances in web documents, acoustic features of the song's audio content, and similarity with other tagged songs. Given these individual data sources, we explore the question of how to aggregate them. We find that fixed-combination approaches like sum and max perform well but that trained linear regression models work better. Retrieval performance improves with more data sources. On the other hand, for large numbers of training songs, Bayesian hierarchical models that aim to share information across individual tag regressions offer no advantage.

1. INTRODUCTION

We are interested in developing a *semantic music discovery engine* in which users enter text queries and receive a ranked list of relevant songs. This task requires a *semantic music index*, i.e., a mapping between songs and associated tags. A *tag*, such as "afro-cuban roots," "heavy metal," or "steel-string guitar," is a short text token which describes some meaningful aspect of the music (e.g., genre, instrumentation, emotion, geographical origins). In this paper, our goal will be to compute a real-valued score $\widehat{y}_{st}$ that expresses how strongly tag t applies to song s.

There are a number of ways to collect semantic annotations of music. [1] compare five such approaches: surveys, social tagging, games, web documents, and audio content. Each of these data sources offers a different perspective, and each has its own strengths and weaknesses (e.g., scalability, popularity bias, accuracy), so we may wish to collect information from several of them. The question then becomes how to combine that information into a single score for use in our semantic index.

In Section 2, we describe three sources of music information that we have collected: text mining web documents, content-based audio analysis, and collaborative filtering. Section 3 describes various approaches for combining these sources, including simple fixed rules, as well as a trained *regression* model in which combination weights depend on the quality and sparsity of the input data. We explore both ordinary linear and logistic regression, as well as Bayesian hierarchical models that aim to share information across tags. Section 4 describes our experimental setup, which includes a ground-truth corpus of 10,870 songs for two vocabularies (71 Genre tags and 151 Acoustic tags) collected from Pandora's Music Genome Project. [1] Section 6 concludes.

2. MUSIC INFORMATION SOURCES

We collect semantic-annotation information from three sources: web documents (WD), content-based audio analysis (CB), and collaborative filtering (CF). For each song s and tag t, we use these sources to generate scores—denoted x_{st}^{WD}, x_{st}^{CB}, and x_{st}^{CF}, respectively—indicating how well t describes s.

2.1 Web Documents

Tags that appropriately describe a song will tend to appear in association with the song's name in natural-language text documents. We exploit this fact by downloading from the web pages that describe the song and counting how often the proposed tag appears within them.

Given a song s, we generate a database D_s of documents by querying Google for "song name" "artist name" in lower-case (e.g., "enjoy the silence" "depeche mode"). We download all hits in the top 10 and clean the HTML files into raw text. This was done for a total of 9,359 songs. Then, for each tag t, we compute

$$x_{st}^{\text{WD}} = \sum_{d \in D_s} \frac{n_{td}}{N_{td}},$$

where n_{td} is a measure that roughly expresses how many times t actually appeared in document d, and N_{td} is the number of times t *could have* appeared in d. N_{td} is just

[1] See http://www.pandora.com/mgp.shtml

$|d| \,/\, |t|$, the number of words in d divided by the number of words in t. n_{td} is a bit more complicated. For long tags, such as "call and answer vocal harmony (antiphony)," positional searches for the entire phrase would not work well. On the other hand, searching for the appearance of any of the words in t would yield too many hits. We compromise by computing n_{td} as the minimum number of hits for any word, taken over all words in t. In the case when the words in t appear in d only in the correct order, n_{td} will in fact be equal to the number of occurrences of the full phrase t.

2.2 Content-Based Audio Analysis

A second potential source of semantic information about a song is the audio content itself. For this purpose we use the supervised multiclass labeling (SML) model recently proposed by [2].

The audio track of a song is represented as a bag of feature vectors $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$, where each $\mathbf{x}_i$ is a feature vector that represents a short-time segment of audio, and T depends on the length of the song. We use the expectation maximization (EM) algorithm to learn a *song-specific* Gaussian mixture model (GMM) distribution over each $\mathcal{X}$. Then, for each tag in our vocabulary, we learn a *tag-specific* GMM using the Mixture Hierarchies EM algorithm [3]. This algorithm combines the set of song-specific GMMs for all the songs that have been associated with the tag. Given a novel song s, we compute the likelihood that its bag of feature vectors $\mathcal{X}_s$ would have been generated by each of the tag GMMs. Normalizing these likelihoods using the technique described in [2] yields our set of scores x_{st}^{CB}, which can be interpreted as the parameters of a multinomial distribution over the vocabulary of tags.

We use the popular Mel frequency cepstral coefficients (MFCCs) as our audio feature representation since it was incorporated into all of the top performing autotagging systems in the 2008 MIREX tag classification task [2, 4–6]. MFCCs are loosely associated with the musical notion of timbre ("color") of the music because they are a low-dimensional representation of the frequency spectrum of a a short-time audio sample. For each monaural song in the data set, sampled at 22,050 Hz, we compute the first 13 MFCCs for each half-overlapping short-time ($\sim$23 msec) window from 6 five-second clips spaced at uniform intervals over the length of the song. Over the time series of audio segments, we calculate the first and second instantaneous derivatives (referred to as *deltas*) for 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 a 4-component GMM. We model each tag with an 8-component GMM.

2.3 Collaborative Filtering

One additional source of semantic information is user playlists: If two songs appear together in a large number of listener collections, one possible reason is that the songs share certain attributes (say, "punk influences") that the listeners enjoy. This suggests the idea of *tag propagation*: Find songs that tend to co-occur in playlists, and transfer tags from one of them to the other. A more robust approach is to find the collection of k songs ($k = 32$ here) that have the strongest co-occurence score with a given song s. For each tag t, we take the association x_{st}^{CF} of s with t to be the fraction of those 32 songs to which t applies. We set this number to 0 if the fraction is below a threshold of 0.3. The reasons for these choices, as well as further details on the entire data-collection process and choice of tag sets, appear in [7].

Our data consist of 400,000 user music libraries from last.fm, where a *library* is taken to be the set of items that a user listens to at least 1% of the time. It turns out that data at the song level is too sparse to generate meaningful co-occurence statistics, so we instead work at the artist level. We say that a tag applies to an artist if the tag applies to any of that artist's songs. At the end of the propagation process, we transfer an artist's score for a tag to each of its songs. We find the 32 closest artists using the following similarity score. Between artists i and j, we take

$$\text{sim}(i,j) = \frac{p(i,j)}{\sqrt{p(i)p(j)}},$$

where $p(i,j)$ is the fraction of all artist co-occurrences represented by artists i and j, and $p(i)$ is the fraction of all co-occurrences containing artist i.

3. COMBINING METHODS

Given the data sources described in Section 2, how can we aggregate them? This general question has been well studied and is known variously as *combining expert judgments* (e.g., [8, 9]), *multi-sensor data fusion* (e.g., [10]), *information fusion* (e.g., [11]), or *combining classifiers* (e.g., [12, 13]). Rather than reviewing the entire body of literature on the subject, we focus on two of the most basic approaches: Fixed-combination rules and trained combiners, specifically regression.

3.1 Fixed Combiners

Fixed combining rules take the output score $\widehat{y}_{st}$ to be a simple function of the input scores: e.g., max, min, median, sum, or product [14, sec. 3]. Usually the input scores x_{st}^i, with $i \in \{\text{WD}, \text{CB}, \text{CF}\}$, are calibrated so that they correspond to confidences or probabilities p_{st}^i that t applies to s given the source. This can be done, for instance, by standardizing the input scores to have mean 0 and variance 1 and then taking

$$p_{st}^i = \frac{1}{1 + \exp\left(-\alpha x_{st}^i\right)}$$

for some α [14, sec. 4.1]. We use $\alpha = 1$ in this paper.

A disadvantage of this technique, however, is that each source is treated on equal footing, when in fact, one of our sources may be far more trustworthy or better informed [14, sec. 1]. One method that overcomes this limitation is Bayesian Model Averaging (BMA) (e.g., [15]), which assumes that one of the data sources is the "correct" source

and takes the final probability to be a weighted combination of the input probabilities:

$$p_{st}^{\text{all sources}} = \sum_{i \in \{\text{WD,CB,CF}\}} p_{st}^i p_i,$$

where p_i is the probability that source i is correct. As [16, sec. 1] point out, this assumption is often unrealistic, as the truth about whether a tag applies to a song needn't be captured by exactly one of our data sources. Still, the idea of taking our final score $\widehat{y}_{st}$ to be a weighted combination of the input scores—

$$\widehat{y}_{st} = \sum_{i \in \{\text{WD,CB,CF}\}} \beta_t^i x_{st}^i \quad (1)$$

for some weights β_t^i—does seem like a natural way to account for the differential predictive value of different inputs. The question is how to determine the weights.

3.2 Trained Combiners

If we have training data for a subset of songs, [2] the obvious answer is to use supervised learning. This is the *trained combiners* approach advocated in [14]. Indeed, (1) has the form of a linear-regression model, and we can determine the weights of the sources just by treating them as input features and computing their regression coefficients.

We try both linear and logistic regression, predicting the ground truth $y_{st} \in \{0, 1\}$ by the individual scores x_{st}^i, as well as an intercept and possibly other features of interest (see Section 3.4). We take our predicted values $\widehat{y}_{st}$ to be real-valued so that we can more finely rank-order songs than with 0/1 labels. Regression is a convenient combination approach because it potentially allows us to use a number of standard statistical tools: p-values for the significance of regression coefficients, prediction intervals for our output scores, model selection based on residual sum of squares, and many more advanced techniques.

3.3 Hierarchical Regression Models

One such technique is borrowing of information across tags. Each tag has its own regression model, but we might suspect that these models share significant structure: For instance, if collaborative filtering tends to be a highly predictive source, we would expect its coefficient to be consistently large. And the linear combination of sources that best predicts the tag "traditional country" is probably similar to the one that best predicts "contemporary country."

One way to capture this intuition is with a Bayesian hierarchical linear model (e.g., [17]). We'll illustrate this concept in the case of a single regression coefficient β_t for a single data source x_{st} without an intercept, but similar equations apply in the multivariate setting. Independent regression across the T tags assumes

$$y_{st} = \beta_t x_{st} + \epsilon_{st}, \;\; \epsilon_{st} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_t^2), \;\; t = 1, \ldots, T \quad (2)$$

for some variances σ_t^2, with no relationship among the β_t values. We call this the *Independent Linear* model. The *Independent Logistic* model is the same, except that y_{st} is replaced by the log-odds $\ln\left(\frac{p_{st}}{1-p_{st}}\right)$, where p_{st} is the probability that $y_{st} = 1$.

In a hierarchical model, we assume in addition to (2) that the β_t's share a common structure:

$$\beta_t = \overline{\beta} + v_t, \quad v_t \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2), \quad t = 1, \ldots, T. \quad (3)$$

For instance, if we had three tags with independent regression coefficients of 0.1, 0.2, and 0.3, it might be reasonable to suppose that $\overline{\beta} \approx 0.2$ with $\sigma \approx 0.1$. We can further assume a prior over $\overline{\beta}$ and perform Bayesian inference to estimate the parameters. The multivariate version of this model we call *Hierarchical Linear*, and the corresponding version in which y_{st} is replaced by the log-odds that $y_{st} = 1$ we call *Hierarchical Logistic*.

We might also assume that v_t in (3), rather than being normally distributed, is drawn from a mixture of normal distributions. For instance, perhaps the web-document source is much better at predicting genre labels than acoustic ones, so that its β_t values for genre tags cluster around 0.2, say, while its β_t values for acoustic tags cluster around 0.05. In that case, β_t could be modeled by taking $\overline{\beta} = 0.05$, with v_t having peaks at 0 and 0.15. We call this model *Mixture Linear*$_k$, where k is the number of centers. [3]

3.4 Regression Models

Equation (1) suggests the basic regression model to use, although in practice we include an intercept, which we find always to be highly statistically significant. We can also regress on just one or two of the main sources at a time.

A nice aspect of using regression is that we can include extra features in our model (assuming we expect they'll contribute useful information rather than meaningless noise that will lead us to overfit). In particular, we include *scrobble counts* from last.fm as a measure of the popularity of the artist who wrote the given song. If we suspected that more popular songs had more nonzero y_{st} values in our ground-truth, we would expect this popularity term to have a high positive regression coefficient. Including the term could be seen as a way of controlling for popularity bias if we omit the popularity feature when we predict $\widehat{y}_{st}$ for novel songs. We can also include terms for the interaction of data sources with popularity. A positive interaction coefficient would indicate that the data source gives a more confident prediction that a tag applies to a song when the song's artist is popular.

[2] Another possible scenario is that, rather than having ground-truth labels for a subset of our songs, we have data that applies to all of our songs but is weakly labeled, i.e., not every song that applies for a given tag is labeled as such. If our input data sources are less sparse, we can use them to "fill in zeros" in the ground truth while preserving the labels that the ground truth had already.

[3] See Chapters 3 and 5 of [18] for details on each of these three hierarchical models in a more general setting.

4. EXPERIMENTAL SETUP

4.1 Data Set

Our data set consists of 10,870 songs representing 19 top-level genres (e.g., rock, classical, electronic) and 180 subgenres (e.g., grunge, romantic period opera, trance). We have approximately 60 songs per subgenre. Each song is associated with one or more genres and one or more subgenres. For each song, we also attempt to collect between 2 and 10 acoustic tags from Pandora's Music Genome Project vocabulary. This vocabulary consists of over 1,000 unique tags like "dominant bass riff," "gravelly male vocalist," and "acoustic sonority." These acoustic tags can be thought to be *objective* in that two trained experts can annotate a song using the same tags with high probability [19].

4.2 Cross-Validation Setup

We evaluate the retrieval performance of our combined scores using five-fold cross-validation on the Pandora data set. Ordinarily, this would involve training our regression model on 4/5 of the data and testing on the remaining 1/5. However, we need to be careful here, because our content-based data source also trains on the Pandora data set. The danger is that the content-based system may overfit the training data, and because our regression model would be using the same training data, the model might overweight the content-based source. [14, sec. 5] notes this problem and suggests that it be addressed by dividing the training set into two parts, which we do as follows.

We divide the songs into five partitions, each with roughly 2,000 songs. We apply an *artist filter* to the partitions, with all of the songs by an artist appearing in a single fold, to avoid overfitting our model to the particular artists that appear in our training set. On three of the partitions we train the content-based system, using it to then obtain predictions for the songs in the remaining two. We use one of those partitions (roughly 2,000 songs) to train our regression model, which then makes its predictions on the final partition. We then cycle this process five times. The reason for the uneven split between the two training sets is that the content-based system needs to learn many more parameters than our regression model, which typically has at most five coefficients.

4.3 Tag Pruning

Some tags are labeled with too few songs to be useful for training when we divide the songs into five partitions, so we prune them. In particular, the content-based training considers only tags that have at least 20 positive instances in the ground truth over each possible set of three partitions on which to train. In addition, our regression model requires that each single partition have at least one positive ground-truth song (since it would be trivial to train a model when the y_{st}'s are all 0) and at least one positive song in each of the three main data sources. After pruning we are left with 71 Genre tags and 151 Acoustic tags.

4.4 Implementation Details

Regression works best when the features are roughly normally distributed, so we transform some of the input scores for this purpose. For popularity counts, which range anywhere from 1 to over 15 million, we apply a log transformation. For the web-document source, which is based on count data, we apply a square-root transformation [20, p. 84]. We then standardize each data source by subtracting the mean and dividing by the standard deviation for a given tag. The x_{st}^i's referred to in Section 3.1 are these standardized values.

For a small number of tags, β_t^i was estimated as negative for one or two of the input data sources. Because we believe that our main three data sources, while potentially unhelpful, should not be anti-predictive of the ground truth, we eliminate negative coefficients by setting them to 0 when they occur. (Making this adjustment results in a small but statistically significant improvement in mean average precision and area under the ROC curve for both Genre and Acoustic tags.) We do allow popularity to have a negative coefficient, and we remove this restriction entirely when considering models with interaction terms.

4.5 Regression Types

We implement Independent Linear and Independent Logistic regression using the basic `lm` and `glm` functions of the `R` language. For the hierarchical regressions, we use the `bayesm` package [21], specifically the `rhierLinearModel`, `rhierBinLogit`, and `rhierLinearMixture` functions for Hierarchical Linear, Hierarchical Logistic, and Mixture Linear$_k$, respectively, with all optional parameters set to their default values. These methods use Markov chain Monte Carlo to sample the entire posterior distribution for the β_t^i's given the data, but we simply take our β_t^i estimate to be the average of these draws. Performance is good with as few as a few hundred samples, but we find that area under the ROC curve does not level off completely until 5,000 to 10,000 draws. For the results in this paper, we sample 15,000 draws, which takes on the order of 30 minutes with roughly 100 tags and 2,000 songs. A parameter sweep of the number k of means in the Gaussian-mixture prior showed no appreciable differences over the range 2 to 50, so we use $k = 2$ as the default.

5. RESULTS AND DISCUSSION

We assess performance using the four standard information-retrieval metrics listed in Table 1 (see [22, sec. 8.4] for explanation of each). We have also made available [4] a list of the top 5 predicted songs for each tag for purposes of qualitative evaluation.

[4] See `http://www.sccs.swarthmore.edu/users/09/btomasi1/combiner/`

Table 1. Area under the ROC curve, mean average precision, R-precision, and 10-precision for various settings described further in the text. Rows are ordered by average AUC for Genre tags. Means and standard errors are taken over the tags, applied to the averages of five-fold cross-validation. (To compute standard errors with respect to each individual CV fold, divide the reported standard errors by a further $\sqrt{5}$.) The data-source abbreviations are web documents (WD), collaborative filtering (CF), content-based analysis (CB), popularity (P), all three main sources in the model (All3), and interactions with each of the three main sources (I).

Regression Model								
	71 Genre Tags				151 Acoustic Tags			
	AUC	MAP	R-Prec	10-Prec	AUC	MAP	R-Prec	10-Prec
Random	0.502±0.003	0.09±0.01	0.08±0.01	0.08±0.02	0.508±0.003	0.032±0.003	0.030±0.003	0.03±0.00
WD	0.666±0.010	0.25±0.02	0.29±0.02	0.47±0.03	0.616±0.006	0.135±0.007	0.181±0.008	0.29±0.02
CF	0.732±0.010	0.45±0.02	0.45±0.02	0.72±0.04	0.641±0.008	0.154±0.010	0.213±0.011	0.25±0.02
CB	0.781±0.014	0.23±0.02	0.25±0.02	0.38±0.03	0.836±0.008	0.141±0.007	0.161±0.008	0.19±0.01
WD&CF	0.789±0.010	0.50±0.02	0.50±0.02	**0.74±0.04**	0.724±0.007	0.231±0.010	0.280±0.011	0.40±0.02
CB&WD	0.819±0.010	0.32±0.02	0.34±0.02	0.53±0.03	0.870±0.006	0.220±0.009	0.246±0.009	0.36±0.02
CB&CF	0.853±0.009	0.49±0.02	0.48±0.02	0.73±0.04	0.861±0.007	0.213±0.010	0.244±0.010	0.29±0.01
All3&P&I	0.856±0.007	**0.52±0.02**	0.50±0.02	**0.74±0.04**	0.860±0.006	0.262±0.010	0.288±0.010	0.40±0.02
All3	0.871±0.007	**0.52±0.02**	0.50±0.02	**0.74±0.04**	**0.888±0.006**	0.276±0.010	0.298±0.010	**0.42±0.02**
All3&P	**0.876±0.007**	**0.52±0.02**	**0.51±0.02**	**0.74±0.04**	0.887±0.006	**0.277±0.010**	**0.299±0.010**	**0.42±0.02**
Combination Method								
	71 Genre Tags				151 Acoustic Tags			
	AUC	MAP	R-Prec	10-Prec	AUC	MAP	R-Prec	10-Prec
Min	0.658±0.015	0.27±0.02	0.27±0.02	0.60±0.04	0.654±0.009	0.121±0.006	0.161±0.008	0.26±0.01
Product	0.826±0.009	0.42±0.03	0.41±0.02	0.67±0.04	0.814±0.006	0.197±0.008	0.232±0.009	0.32±0.01
Median	0.826±0.009	0.43±0.02	0.43±0.02	0.68±0.04	0.820±0.006	0.219±0.009	0.261±0.009	0.35±0.02
Sum	0.851±0.007	0.44±0.03	0.44±0.02	0.69±0.04	0.847±0.006	0.220±0.009	0.252±0.009	0.34±0.01
Max	0.856±0.007	0.46±0.02	0.48±0.02	0.59±0.03	0.859±0.006	0.239±0.009	0.274±0.009	0.34±0.01
Ind Log	0.866±0.006	0.51±0.03	0.50±0.02	0.72±0.04	0.875±0.005	0.266±0.010	0.293±0.010	0.40±0.02
Hier Log	0.872±0.006	0.51±0.03	0.50±0.02	0.73±0.04	0.883±0.006	0.272±0.010	0.296±0.010	0.40±0.02
Hier Mix	**0.876±0.007**	**0.52±0.02**	**0.51±0.02**	**0.74±0.04**	**0.887±0.006**	**0.277±0.010**	**0.299±0.010**	**0.42±0.02**
Hier Lin	**0.876±0.007**	**0.52±0.02**	**0.51±0.02**	**0.74±0.04**	**0.887±0.006**	**0.277±0.010**	**0.299±0.010**	**0.42±0.02**
Ind Lin	**0.876±0.007**	**0.52±0.02**	**0.51±0.02**	**0.74±0.04**	**0.887±0.006**	**0.277±0.010**	**0.299±0.010**	**0.42±0.02**

5.1 Regression Models

The top half of Table 1 reports the performance of the Independent Linear model on subsets of the data sources, as well as models that include popularity information. The Random method is a regression model in which all sources have coefficients of 0, so that the final ranking of songs is the same as the (randomized) order in which they were initially seen. Each source alone clearly performs better than random, and each addition of a new source results in a statistically significant improvement in AUC.[5] This is consistent with the fact that the data sources are relatively uncorrelated, having correlation coefficients typically less than 0.3 and often less than 0.1, depending on the tag.

According to the AUC measure, CB is the individually most predictive source, while according to precision, CF is. We suspect this reflects the fact that CB's input representation is dense, providing nonzero scores for 91.2% of songs for each tag, while CF's input contains mostly zeros, with scores for only an average across tags of 2.4% of songs. (WD falls in the middle, with nonzero scores for an across-tag average of 13.7% of songs.) When CF has a nonzero value, it really means something, so that CF's top results are very precise. Toward the later end of the ranked results list, however, CF is essentially random, while CB still provides useful information.

It is interesting to observe that CB's advantage over CF in terms of AUC is larger in the case of acoustic tags than genre tags, perhaps because acoustic tags are inherently more predictable by audio content alone.

Popularity data was not especially helpful. While its addition to the three main sources did result in a statistically significant AUC improvement for Genre tags (p-value 0.007), it did not for Acoustic tags (p-value 0.4), and the magnitude of difference was relatively small. In some sense, this is a welcome result, since it suggests that the Pandora labels are not biased very much by whether an artist is well-known. The interaction model contained too many features and tended to overfit, which is unsurprising given the modest usefulness of the main popularity term.

5.2 Coefficient Magnitudes

Our default regression model was Independent Linear with the three main data sources, popularity, and an intercept. Averaging the β_t^i's over all of the tags t gives the following prediction equation for Genre tags (the one for Acoustic tags is similar):

$$\widehat{y}_{st} = 0.08 + 0.02x_{st}^{\text{WD}} + 0.02x_{st}^{\text{CB}} + 0.09x_{st}^{\text{CF}} + 0.02x_{s}^{\text{pop}}.$$

[5] This is usually apparent from inspection of standard errors, but we verify it by checking that p-values are less than 0.05 for paired t-tests on the per-tag AUC values. In fact, the only pairs between which this fails to hold are (1) CB and WD&CF for Genre tags, (2) All3 and All3&P for Acoustic tags, and (3) CB&CF and All3&P&I for both tag types. If we apply a conservative Bonferroni correction for the $\frac{10 \cdot 9}{2}$ pairs of tests, a few more pairs become not significant, including the transition from CB&CF to All3 for Genre tags.

Because the x^i_{st}'s represent the transformed and standardized input values (see Section 4.4), the standard error for each β^i_{st} is roughly the same for a given tag, [6] so that the t-statistic of each coefficient is roughly proportional to the coefficient's magnitude. It's worth noting, though, that statistical significance of a coefficient as different from zero is not identical with usefulness as a data source. Indeed, we saw in Section 5.1 that CB was individually more predictive than CF, at least as measured by AUC, while CB's coefficient is 0.02 instead of 0.09. The reason may again be that CB provides a denser input representation than CF; CF can afford to have a large β^{CF}_{st} because in the rare cases when its values are nonzero, they're strongly informative.

5.3 Regression Types

The bottom half of Table 1 shows various combination techniques. The regression approaches use the model All3&P, while the fixed-combination approaches use just the three main sources. All trained regression models outperform all fixed-combining methods. [7] This result contrasts with the finding by [11] that the simple sum rule outperformed supervised linear-discriminant analysis (similar to logistic regression) and decision trees. Still, Sum and especially Max do not fare badly and would not be unreasonable choices for a simple combining system. That Max is close to Independent Logistic regression is perhaps unsurprising, because the fixed-combining methods apply the same sigmoid transformation to the input data that logistic regression uses.

While Hierarchical Logistic regression did slightly outperform Independent Logistic, the hierarchical and mixture models showed no apparent effect for linear regression. We suspect this is because the number of observations (songs) is so large (over 2,100 on average) that the Bayesian prior terms in those models wash out. To confirm this, we tried artificially restricting ourselves to 250 songs, and in that case, the hierarchical methods did slightly outperform their independent counterparts.

6. CONCLUSIONS

We have shown that combining different sources of song-tag annotation information improves retrieval performance. Fixed-combining methods like Sum and Max do a fine job for simple systems, but retrieval improves when we use a trained combining method like linear or logistic regression. In settings where large numbers of songs are available, basic Independent Linear regression on each tag separately gives results just as good as more sophisticated hierarchical models, while allowing for easier implementation, faster computation, and greater parallelizability.

[6] This is only "roughly" because of small inter-feature correlations.

[7] Paired t-tests on the AUC values for individual tags give p-values less than 0.05 for all pairs except between (1) Product and Median and (2) Sum and Max for Genre tags, and (3) all three of Hier Mix, Hier Lin, and Ind Lin for both tag types. For Genre tags, five more pairs fail to reject the null hypothesis if we apply a Bonferroni correction on the $\frac{10 \cdot 9}{2}$ pairs of tests, including Sum vs. Independent Logistic (p-value 0.01).

7. REFERENCES

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

[2] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE TASLP*, 2008.

[3] N. Vasconcelos. Image indexing with mixture hierarchies. *IEEE CVPR*, pages 3–10, 2001.

[4] S. J. Downie. The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, 2008.

[5] M. Mandel and D. Ellis. Multiple-instance learning for music information retrieval. In *ISMIR*, 2008.

[6] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic generation of social tags for music recommendation. In *NIPS*, 2007.

[7] J. Kim, B. Tomasik, and D. Turnbull. Using artist similarity to propagate semantic information. *ISMIR*, 2009.

[8] P. A. Morris. Combining expert judgments: A Bayesian approach. *Management Science*, pages 679–693, 1977.

[9] R. A. Jacobs. Methods for combining experts' probability assessments. *Neural computation*, 7(5):867–888, 1995.

[10] H. B. Mitchell. *Multi-Sensor Data Fusion: An Introduction*. Springer, 1 edition, September 2007.

[11] A. Ross and A. Jain. Information fusion in biometrics. *Pattern Recognition Letters*, 24(13):2115–2125, 2003.

[12] J. Kittler. Combining classifiers: A theoretical framework. *Pattern Analysis & Applications*, 1(1):18–27, 1998.

[13] C. De Stefano, C. D'Elia, A. Marcelli, and A. S. di Freca. Using bayesian network for combining classifiers. In *ICIAP*, pages 73–80, 2007.

[14] R. Duin. The combining classifier: To train or not to train? In *ICPR*, volume 16, pages 765–770, 2002.

[15] J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky. Bayesian model averaging: a tutorial. *Statistical science*, pages 382–401, 1999.

[16] Z. Ghahramani and H. C. Kim. Bayesian classifier combination. *Gatsby Computational Neuroscience Unit Tech Report*, 2003.

[17] D. V. Lindley and A. F. M. Smith. Bayes estimates for the linear model. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–41, 1972.

[18] P. E. Rossi and G. M. Allenby. Bayesian statistics and marketing. *Marketing Science*, pages 304–328, 2003.

[19] T. Westergren. Personal notes from Pandora get-together in San Diego, March 2007.

[20] J. J. Faraway. *Practical Regression and ANOVA using R*. July 2002. `http://cran.r-project.org/doc/contrib/Faraway-PRA.pdf`.

[21] P. E. Rossi and R. McCulloch. bayesm. R package ver. 2.2-2, 2008-06-09, `http://faculty.chicagobooth.edu/peter.rossi/research/bsm.html`.

[22] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.