MULTIPLE-INSTANCE LEARNING FOR MUSIC INFORMATION RETRIEVAL

Michael I. Mandel
LabROSA, Dept. Elec. Eng.,
Columbia University, NY, NY
mim@ee.columbia.edu

Daniel P. W. Ellis
LabROSA, Dept. Elec. Eng.,
Columbia University, NY, NY
dpwe@ee.columbia.edu

ABSTRACT

Multiple-instance learning algorithms train classifiers from lightly supervised data, i.e. labeled collections of items, rather than labeled items. We compare the multiple-instance learners mi-SVM and MILES on the task of classifying 10-second song clips. These classifiers are trained on tags at the track, album, and artist levels, or granularities, that have been derived from tags at the clip granularity, allowing us to test the effectiveness of the learners at recovering the clip labeling in the training set and predicting the clip labeling for a held-out test set. We find that mi-SVM is better than a control at the recovery task on training clips, with an average classification accuracy as high as 87% over 43 tags; on test clips, it is comparable to the control with an average classification accuracy of up to 68%. MILES performed adequately on the recovery task, but poorly on the test clips.

1 INTRODUCTION

There are many high quality sources of metadata about musical material such as Last.fm, the All Music Guide, Pandora.com, etc. Typically, however, each source provides metadata only at certain granularities, i.e. describes the music only at certain scales. For example, the All Music Guide provides metadata about many artists and albums, but few tracks. Similarly, Last.fm users have described a large proportion of artists, a smaller proportion of albums, and an even smaller proportion of tracks. Furthermore, there are no publicly accessible, large-scale sources of metadata describing parts of tracks known as *clips*, here taken to be 10-second excerpts. This paper describes the use of clip-level classifiers to refine descriptions from one granularity to finer granularities, e.g. using audio classifiers trained on descriptions of artists to infer descriptions of albums, tracks, or clips.

Many descriptions of music apply at multiple granularities, like *rap*, or *saxophone*, although certain descriptions are valid only at specific granularities like *seen live* or *albums I own*. Descriptions valid at one granularity, however, might only apply to certain elements at a finer granularity. For example, at the artist level, the Beatles could very reasonably be tagged *psychedelic*. This tag would certainly apply to an album like *Sgt. Pepper's Lonely Hearts Club Band* but would not apply to one like *Meet the Beatles*. Similarly, the John Coltrane track "Giant Steps" could very reasonably be tagged *saxophone*. While valid for most clips in the track, it most notably is not valid during the piano solo. This paper describes systems capable of deriving, from feature similarity and a list of *psychedelic* artists or *saxophone* tracks, the clips for which these tags are most appropriate.

By considering tags one at a time, as either being present or absent from a clip, we pose the automatic tagging (autotagging) problem as clip classification. In this framework, the task of metadata refinement is known as *multiple instance learning* (MIL) [6]. In MIL, classifiers are trained on labels that are only applied to collections of *instances*, known as *bags*. Positive bags contain one or more positive instances, while negative bags contain no positive instances. Labeled bags provide less information to the learner than labeled instances, but are still effective at training classifiers. For the purposes of this paper, clips are the instances to be classified, and artists, albums, and tracks, in turn, are the bags.

There are two problems addressed in the multiple-instances learning (MIL) literature, the classification of bags and the classification of instances. As we are interested in refining musical metadata from bags to the instances within them, we only concern ourselves with multiple-instance learners that are capable of classifying instances. A related problem that we also examine is the training of a general instance-level classifier from bag-level labels. This task is slightly different in that the instances to be labeled are not in the training bags, and are unseen at training time.

To evaluate the applicability of MIL to music, we use the data from our MajorMiner game [10]. The game has collected approximately 12,000 clip-level descriptions of approximately 2,200 clips from many different tracks, albums, and artists. The most popular descriptions have been applied to hundreds of clips, and there are 43 tags that have been applied to at least 35 clips. Previous authors have generally used datasets that were labeled at the bag level, making it difficult to evaluate instance-level classification. Sometimes a subset of the data was laboriously annotated to allow the evaluation of instance-level classification. In the MajorMiner dataset, however, tags are applied directly to clips, making it possible to test instance-level classification in both the training set and a separate test set. By deriving bag tags from clip tags in the training set, we can directly test the ability of multiple instance learners to recover metadata at the instance level from the bag level. This derivation adheres to the MIL formulation, labeling a given bag positive as a positive example of a tag if any of its clips have been labeled with that tag. In addition, a held-out test set allows us to evaluate the generalization of these classifiers to instances outside the training set.

1.1 Previous work

A number of authors have explored the link between music and text. Whitman and Ellis [14] trained a system for associating music with noun phrases and adjectives using a collection of reviews from the All Music Guide and Pitchfork Media. This work was based on the earlier work described in [15]. More recently, [12] used a naive Bayes classifier to both annotate and retrieve music based on an association between the music and text. Eck et al. [7] used boosted classifiers to identify the top k tags describing a particular track, training the classifiers on tags that the users of Last.fm had entered for the track's artist.

The MIL problem was first formulated for the task of digit recognition [8], in which a neural network was trained with the information of whether a given digit was present, but not where it was present. In this case, the bags were regions in which the digit was known to be present and the instances were shifted and windowed sub-regions. Another early application of MIL was to the problem of drug discovery [6], in which the bags were molecules and the instances were conformations of those molecules.

MIL has also been applied to object detection in images, in which the bags were images and the instances were either automatically segmented image regions [5] or automatically identified interest points [3]. It has been applied to video classification to match names and faces [16], in which the instances were (name, face) pairs, the bags were scenes, and the task was to determine whether a face had any names associated with it or not. And it has been applied to text classification [1], in which the bags were documents and the instances were sentences or paragraphs.

Many learning frameworks have been applied to MIL, including boosting [13], Markov chain Monte Carlo [3], and both 1-norm [4] and 2-norm support vector machines [1]. In this work, we compare the performance of two SVM-based MIL methods, mi-SVM [1] and MILES [4], on the task of autotagging 10-second song clips. These two algorithms are explained in greater detail in the next section.

2 MULTIPLE-INSTANCE LEARNING

The following notation is common to both mi-SVM and MILES. We denote the ith bag as B_i, of size ℓ_i, and the jth instance in that bag as x_{ij} where $j \in 1 \dots \ell_i$. The label for bag i is $Y_i \in \{-1, 1\}$ and the label for instance x_{ij} is y_{ij}. The set of positive bag indices is defined as $I^+ \equiv \{i : Y_i = 1\}$ and similarly the set of negative bag indices $I^- \equiv \{i : Y_i = -1\}$. All instances in a negative bag are negative and a single positive instance in a bag forces the bag to be positive, meaning that

$$Y_i = \max_j y_{ij}. \tag{1}$$

2.1 The mi-SVM algorithm

The mi-SVM algorithm is the instance-level MIL support vector machine classifier presented in [1]. Support vector machines generally maximize the margin around a hyperplane separating positive from negative examples. In the MIL setting, however, the optimal labeling of the points in positive bags must be computed as well, creating the following mixed integer quadratic program (QP)

$$\begin{aligned}
\min_{\{y_{ij}\},\mathbf{w},b,\xi} \quad & \tfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{ij}\xi_{ij} \qquad (2)\\
\text{subject to} \quad & \forall i : y_{ij}(\langle \mathbf{w}, \mathbf{x}_{ij}\rangle + b) \geq 1 - \xi_{ij}\\
& y_{ij} \in \{-1, 1\}\\
& \xi_{ij} \geq 0\\
& \forall i \in I^+ : \sum_{j=1}^{\ell_i} \tfrac{1}{2}(y_{ij} + 1) \geq 1\\
& \forall i \in I^- : y_{ij} = -1.
\end{aligned}$$

Unfortunately, it is difficult to solve this integer program directly and [1] presents a heuristic means of approximating it that converges to a local optimum. This heuristic solves the standard SVM quadratic program with the labels fixed, and then uses the solution of the QP to impute the missing labels for instances in positive bags. This alternation continues until the labels no longer change. The instance labels are initialized from the bag labels, so that all instances in positive bags are initially labeled positive and all instances in negative bags are initially (and subsequently) labeled negative.

The number of iterations required for convergence depended on the number of instances in each bag, but was generally on the order of 10-20. We use the dual domain formulation of the standard SVM QP so that a nonlinear kernel could be used to define the similarity between instances; in particular, we used the radial basis function (RBF) kernel. In this dual domain, the classifier is a linear combination of a hopefully sparse subset of the training instances, the support vectors.

2.2 The MILES algorithm

Another SVM-based multiple-instance learner is Multiple-Instance Learning via Embedded Instance Selection (MILES)

[4]. While MILES is mainly a bag classifier, it is also able to derive classifications for instances. Classification with MILES proceeds in three steps. In the first step, bags are projected into a large feature space by computing the similarity between each bag and all of the training instances. The similarity between a bag and an instance is defined as the maximum similarity between any of the instances in that bag and the instance in question,

$$K(B_i, \mathbf{x}) \equiv \max_{k \in 1 \ldots \ell_i} K(\mathbf{x}_{ik}, \mathbf{x}). \tag{3}$$

The feature vector for one bag is then

$$\mathbf{m}_i \equiv \left[K(B_i, \mathbf{x}_{11}) \ldots K(B_i, \mathbf{x}_{N\ell_N})\right]^T . \tag{4}$$

In the second step, a 1-norm support vector machine [2] simultaneously learns a classifier and selects discriminative "features," i.e. instances. The 1-norm SVM solves the following linear program

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \|\mathbf{w}\|_1 + C\mu \sum_{i \in I^+} \xi_i + C(1-\mu) \sum_{i \in I^-} \xi_i \qquad (5) \\ \text{subject to} \quad & \forall i : Y_i(\langle \mathbf{w}, \mathbf{m}_i \rangle + b) \geq 1 - \xi_i \\ & \xi_i \geq 0. \end{aligned}$$

This optimization can be solved as written in the primal domain by representing $\mathbf{w}$ as the difference of two non-negative vectors. The ℓ_1-norm in the objective function encourages sparsity in the elements of $\mathbf{w}$ forcing many of them to 0. Since $\mathbf{w}$ multiplies instance-similarities, only the instances corresponding to nonzero elements of $\mathbf{w}$ affect the classification of new bags. These instances can be considered the support vectors in this case. Thus, the ℓ_1 SVM is also sparse in its selection of instances.

In the third step, classifications of some of the instances are derived from the bag classifications. There is one instance in every bag selected by (3) as the closest to each support vector. A bag's classification is not affected by instances that are not the closest to any support vector, and those instances can be ignored. Any remaining instance that contributes more than a threshold amount to the classification of the bag is considered to be that class itself. We treat unclassified instances as negatively classified even though [4] allows instances to remain unclassified.

While MILES has a number of parameters that need to be tuned, we found its performance quite consistent across a wide range of parameter settings. The μ parameter controls the penalty for misclassifying an instance from a positive bag versus misclassifying an instance from a negative bag. We found it to have little effect in our experiments except when within 10^{-4} of 1, so we kept it at 0.5. The C parameter controls the trade-off between increasing the margin and violating class constraints. We found that it could affect performance when it was small and determined that a value of 10 worked well. The σ parameters is the standard deviation of the RBF kernel. With feature vectors of unit norm, as described in the next section, $\sigma = 1$ worked well. Similarly, mi-SVM has C and σ parameters for which $C = 1$ and $\sigma = 1$ worked well. All of these parameters were tuned on a subset of the tags using different train/test breakdowns of the artists from our main experiments.

3 MUSICAL FEATURES

We use two types of features to describe musical audio. The spectral features come from our earlier work [9] and capture timbral aspects of the music related to instrumentation and production quality. The temporal features are novel, and summarize the beat, tempo, and rhythmic complexity of the music in four different frequency bands. All of these features are calculated on 10-second long clips of songs.

The temporal features are similar to those described in [11]. They are calculated on the magnitude of the Mel spectrogram, including frequencies from 50 Hz to 10,000 Hz, using a window size of 25 ms and a hop size of 10 ms. The mel bands are combined into four large bands at low, low-mid, high-mid, and high frequencies giving the total magnitude in each band over time. The bands are windowed and their Fourier transforms are taken, from which the magnitude of the 0-10 Hz modulation frequencies are kept. The DCT of these magnitudes is then taken and the bottom 50 coefficients of this *envelope cepstrum* are kept for each band. The four bands' vectors are then stacked to form the final, 200-dimensional feature vector.

The spectral features consist of the mean and unwrapped covariance of a clip's mel-frequency cepstral coefficients (MFCCs). The MFCCs are calculated from the mel spectrogram used in the temporal features above. Since the on-diagonal variance terms are strictly positive, their log is taken to make their distribution more Gaussian. We use 18-dimensional MFCCs, and only keep the unique elements of the covariance matrix, for a total of 189 dimensions.

The 0-th cepstral coefficient, the DC modulation frequency, and the on-diagonal variance terms tend to be much larger than other features. We therefore scale each feature to be zero-mean and unit-variance across the set of all instances to make the features more comparable to each other. This scaling improves classification because those dimensions that are biggest are generally not the most discriminative. After the feature dimensions are scaled, the temporal features are multiplied by $\frac{1}{\sqrt{2}}$, a value that was determined empirically to balance the spectral and temporal features well. Finally, the feature vector for each clip is normalized to have unit length to avoid problems with degenerate clips.

4 EXPERIMENTS

The data used in our experiments come from our MajorMiner game [10]. In this game, players label 10-second clips with arbitrary textual descriptions called *tags*, scoring points when others describe the same clips with the same tags. The rules of the game encourage players to use original, yet relevant tags. In our experiments, we only include tags that have been verified by at least two different players on at least 35 clips, ensuring that the concept is relevant overall and to the individual clips. There are 43 such tags and they have been verified approximately 9000 times in total on approximately 2200 clips selected at random from 3900 tracks.

Note that these data do not include strict negative labels. While many clips are tagged *rock*, none are tagged *not rock*. Frequently, however, a clip will be tagged many times without being tagged *rock*. We take this as an indication that *rock* does not apply to that clip. More specifically, a negative example of a particular tag is a clip on which another tag has been verified, but the tag in question has not.

Certain data-handling issues required extra consideration. Before using the tags in our experiments, we performed a number of normalization steps on them, eliminating variations in punctuation, spelling, and suffixing. Around 45 silent clips were culled from the dataset and the last clip in each track was also discarded, as the irregular length of these clips can cause problems and they are generally silent. Finally, a number of tracks came from albums where the artist was indicated as “Various Artists” and “Original Soundtrack.” We excluded these clips from any bags used in our experiments, but allowed them in the instance-level test set, where bags were ignored anyway.

4.1 Procedure

The experiment was five repetitions of a two-fold cross-validation procedure, where artists were assigned to one fold or the other to make sure tags were learned independent of artists. Each tag was evaluated independently as a binary classification task. To create the training set for a particular tag the positive bags with the most instances of that tag were selected from the training fold until either they were all selected or 400 instances had been selected. The negative bags with the most instances were then selected until there were at most as many negative instances as positive instances. On average, the selected track, album, and artist bags contained 2.47, 4.44, and 8.17 instances, respectively. In the training set, bag labels Y_i were calculated from instance labels y_{ij} according to (1).

As an illustration, to create the training and testing datasets for the tag *saxophone* at the artist granularity, all of the artists with any clips tagged *saxophone* are considered positive bags. So Sonny Rollins would be one positive bag, John Coltrane another, etc. All artists without any clips tagged *saxophone* are considered negative bags. To create the training set, the positive artists with the most labeled clips are selected, followed by the negative artists with the most labeled clips.

In addition to mi-SVM and MILES, two control algorithms were included in the evaluation. The first, referred to as the naïve approach, assumes that bag labels apply to all of the instances in a bag and trains an instance-labeling SVM on those labels directly. The second control, referred to as the cheating approach, trains on instance labels directly and serves as a ceiling on performance for instance classification in the training set. Since there are many negative instances in positive bags and this algorithm needs to train on an equal number of positive and negative examples, it only selects a subset of the negative instances to train on. This selection causes its slightly imperfect performance.

The maximum number of positive instances, 400, was chosen to give a reasonable running time for all algorithms. With 400 labeled instances in positive bags, the largest data sets would have 800 examples. Training an individual 1-norm or 2-norm SVM on this many examples takes only a few seconds, but there were about 5000 SVMs to be trained over 10 repetitions, on 43 tags, with 3 different bag sizes for each of 4 algorithms. The slowest algorithm was mi-SVM, which took 6.7 hours to run on one Intel Xeon 1.89 MHz CPU. Second was the naïve approach, which took 67 minutes, followed by MILES at 42 minutes and the cheating approach at 24 minutes.

We evaluate the classifiers on two different tasks. The first, is the recovery of instance labels in the training set. Since the classifier is trained only on bag labels and not directly on instance labels, this task is not trivial. To fix a performance baseline of 0.5, accuracy is evaluated on an equal number of positive and negative instances from the training bags. To increase the precision of the measurement, the maximum number of examples are selected while still maintaining balance, although this causes the variance of the estimates to differ between classes.

While recovering instance labels is one useful application of MIL, it does not allow the classifiers' performance on novel instances to be measured accurately, because of the partial supervision. We instead measure this using instances from the held-out test set, ignoring any notion of bags. To facilitate comparison between the accuracy of classifying each tag, the same number of test examples are selected for each tag, namely 11 positive and 11 negative (setting the baseline accuracy again to 0.5). When combined across two cross validation folds and five repetitions of the experiment, each class is tested on 220 examples. Since MILES classifies bags before classifying instances in those bags, each test point is presented to MILES in its own bag.

5 RESULTS

The overall classification accuracy of the various algorithms can be seen in Table 1. On the training instances, the cheating algorithm was the most accurate, followed by mi-SVM, the naïve algorithm, and MILES. Since the cheating algorithm is an upper bound on training set accuracy, mi-SVM is the most accurate realistic classifier, if only by a small margin.

	Training set			Test set		
	Trk	Alb	Art	Trk	Alb	Art
Cheat	85.9	83.9	82.6	64.7	65.7	66.2
mi-SVM	87.0	80.9	78.0	66.8	67.8	65.4
Naïve	85.0	79.5	77.3	67.4	67.7	66.0
MILES	81.2	73.2	70.0	51.7	50.9	50.6

Table 1. Overall classification accuracy percentages on labeled points in the train and test sets at track, album, and artist granularities.

The mi-SVM algorithm outperformed the naïve algorithm because the naïve algorithm generally returned the bag labels for the instances, while mi-SVM was able to identify some of the negative instances in positive bags.

On the test data, mi-SVM and the naïve algorithm performed equally well, followed by the cheating algorithm, and MILES. The inversion of the cheating and naïve algorithms on the test set was unexpected. It could possibly be caused by the naïve algorithm's ability to use more of the training examples presented to it, or it could indicate that the instance labels do contain some amount of noise that the bag labels smooth over. Such noise could be due to the lack of unambiguous negative labels in the MajorMiner data.

MILES was not very accurate on the test set. It was reasonably accurate on tags with many examples and small training bags, but otherwise it classified all test instances identically, resulting in an accuracy of 0.5. This might be due to a mismatch between the size of the bags in the training and test sets, since the test "bags" were all single instances. In experiments where MILES was tested on bags of a similar size to those it was trained on, its test accuracy more closely followed its training accuracy. In these experiments, MILES seemed to be able to rank test bags well, even when they differed in size from the training set, possibly indicating that the bias, b, was learned incorrectly. In the training set, its bag classifications were almost always correct, while its instance classifications suffered slightly from over-sparsity. In particular, as bag size increased, a smaller proportion of instances in the bag contributed to its classification, leaving all of the others classified as negative by default.

As expected, classification was more accurate on instances in training sets than in test sets. This indicates that even though training instances might not be labeled individually, bag labels still convey useful information. Also as expected, accuracy generally decreases for coarser-granularity bags, more so for training instances than for testing instances. This indicates that larger bags, while still allowing the training of classifiers, constrain the labels of training instances less.

In Figure 1, results on the test set are broken down by tag for each algorithm and bag size. This figure reveals the differences between the tags. Some tags perform better using track bags, while others perform better using artist or album bags. These differences could indicate the granularities at which each tag is appropriate. In particular, one can compare the performance of the naïve algorithm or mi-SVM trained on different bag granularities to determine if one granularity is significantly different from another.

The naïve algorithm trained on artist bags make the assumption that a tag that is appropriate for some of an artist's clips is also appropriate for the rest of them. The accuracy of the resulting classifier, relative to other granularities, should indicate how well this assumption holds. Since mi-SVM is initialized with the same labels as the naïve algorithm, a similar property might hold. We have found some evidence of this phenomenon in our results, but not enough to conclusively prove its existence. For example, the tags *saxophone*, *synth*, *piano*, *soft*, and *vocal* should be most relevant at the track level, and indeed, they train mi-SVM classifiers that are much better for track bags than for artist bags. A counterexample is provided by *trumpet*, however, which is better classified by mi-SVM trained on artist bags.

6 CONCLUSION

We have formulated a number of music information related multiple-instance learning tasks and evaluated the mi-SVM and MILES algorithms on them. By using clip-level tags to derive tags at the track, album, and artist granularities, we have created three different ground truth datasets. These datasets are suitable for testing the learners' ability both to recover the original tags for clips in the training set and to tag clips in a held-out test set. We found that mi-SVM was the most accurate in recovering tags in the training set, followed by the naïve approach of assuming bag tags applied to all instances, and then MILES. In predicting tags for test clips, we found that mi-SVM and the naïve approach were quite comparable, and both were much more accurate than MILES. While these results are promising, many multiple-instance learners have been formulated and it is possible that another one is more appropriate to the task of predicting tags.

6.1 Future work

The most straightforward extension of this work is to larger datasets with more tags and more labeled examples. It should be possible to evaluate the refinement of tags from the artist and album level to the track level, so data from sources like Last.fm, the All Music Guide, and Pandora could be used in the evaluation of MIL. It would also be interesting to qualitatively examine the results of refining labels from these data sources down to the clip level. Many other tasks in music information retrieval could also benefit from the decreased cost of collecting training data within the MIL framework, including polyphonic transcription, singing voice detection, structure finding, and instrument identification.

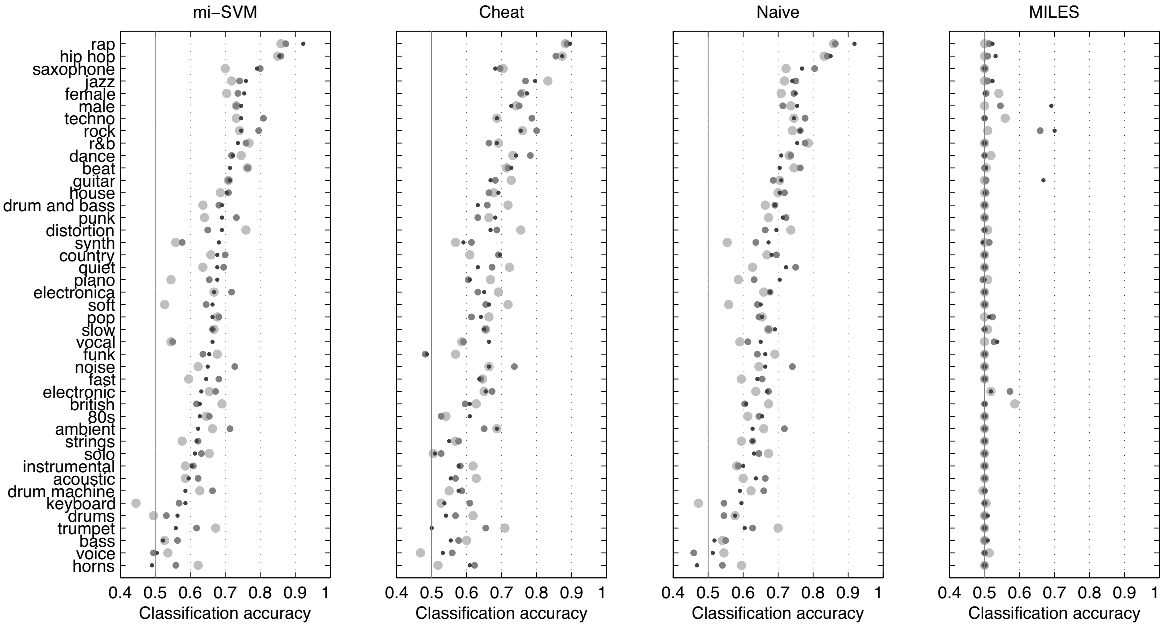

Figure 1. Classification accuracy on the test set, broken down by tag. Each pane is an algorithm, and each style of dot is a different bag granularity. Dots get larger and lighter for coarser granularities: track, album, artist. The tags are ordered by the accuracy of mi-SVM with track bags. For every dot, N=220, meaning that a difference of around 0.06 is statistically significant.

Acknowledgements

The authors would like to thank Stuart Andrews for his advice and useful discussions. This work was supported by the Fu Foundation School of Engineering and Applied Science via a Presidential Fellowship, by the Columbia Academic Quality Fund, and by the National Science Foundation (NSF) under Grants IIS-0238301 and IIS-0713334. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

7 REFERENCES

[1] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In S. Thrun and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 561–568. MIT Press, Cambridge, MA, 2003.

[2] J. Bi, K. Bennett, M. Embrechts, C. Breneman, and M. Song. Dimensionality reduction via sparse support vector machines. *Journal of Machine Learning Research*, 3:1229–1243, 2003.

[3] P. Carbonetto, G. Dorkó, C. Schmid, H. Kück, and N. de Freitas. Learning to recognize objects with little supervision. *International Journal of Computer Vision*, 77(1):219–237, May 2008.

[4] Y. Chen, J. Bi, and J. Z. Wang. MILES: Multiple-instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1931–1947, 2006.

[5] Y. Chen and J. Z. Wang. Image categorization by learning and reasoning with regions. *Journal of Machine Learning Research*, 5:913–939, 2004.

[6] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, January 1997.

[7] D. Eck, P. Lamere, T. B. Mahieux, and S. Green. Automatic generation of social tags for music recommendation. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*. MIT Press, 2008.

[8] J. D. Keeler, D. E. Rumelhart, and W. K. Leow. Integrated segmentation and recognition of hand-printed numerals. In *Advances in Neural Information Processing Systems 3*, pages 557–563, San Francisco, CA, 1990. Morgan Kaufmann Publishers, Inc.

[9] M. I. Mandel and D. P. W. Ellis. Song-level features and support vector machines for music classification. In J. D. Reiss and G. A. Wiggins, editors, *Proc. Intl. Symp. Music Information Retrieval*, pages 594–599, September 2005.

[10] M. I. Mandel and D. P. W. Ellis. A web-based game for collecting music metadata. In S. Dixon, D. Bainbridge, and R. Typke, editors, *Proc. Intl. Symp. Music Information Retrieval*, pages 365–366, 2007.

[11] A. Rauber, E. Pampalk, and D. Merkl. Using psycho-acoustic models and self-organizing maps to create a hie rarchical structuring of music by sound similarity. In M. Fingerhut, editor, *Proc. Intl. Symp. Music Information Retrieval*, pages 71–80, 2002.

[12] D. Turnbull, L. Barrington, and G. Lanckriet. Modeling music and words using a multi-class naive bayes approach. In *Proc. Intl. Symp. Music Information Retrieval*, October 2006.

[13] P. Viola, J. Platt, and C. Zhang. Multiple instance boosting for object detection. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1417–1424. MIT Press, Cambridge, MA, 2006.

[14] B. Whitman and D. Ellis. Automatic record reviews. In *Proc. Intl. Symp. Music Information Retrieval*, pages 470–477, 2004.

[15] B. Whitman and R. Rifkin. Musical query-by-description as a multiclass learning problem. In *IEEE Workshop on Multimedia Signal Processing*, pages 153–156, 2002.

[16] J. Yang, R. Yan, and A. G. Hauptmann. Multiple instance learning for labeling faces in broadcasting news video. In *Proc. ACM Intl. Conf. on Multimedia*, pages 31–40, New York, NY, USA, 2005. ACM.