MUSIC PREFERENCE LEARNING WITH PARTIAL INFORMATION

Yvonne Moh, Peter Orbanz and Joachim M. Buhmann

Institute of Computational Science
ETH Zurich

ABSTRACT

We consider the problem of online learning in a changing environment under sparse user feedback. Specifically, we address the classification of music types according to a user's preferences for a hearing aid application. The classifier, operating under limited computational resources, must be capable of adjusting to types of data not represented in the training set, and to changing user demands. The user provides feedback only occasionally, prompting the classifier to change its state. We propose an online learning algorithm capable of incorporating information from unlabeled data by a semi-supervised strategy, and demonstrate that the use of unlabeled examples significantly improves classification performance if the ratio of labeled points is small.

Index Terms— Online Learning, Semi-Supervised Learning, Music Classification

1. INTRODUCTION

Sound classification in changing acoustic environments poses a challenging statistics or machine learning problem that requires sophisticated online learning strategies. We address the sound classification problem in the context of a hearing aid application. The long-term goal is to design "smart" hearing aids, which incorporate an adaptive amplification unit and an environment sensitive controller. The controller changes the amplification program according to the current properties of the input data and user preferences. A number of authors have considered application of machine learning algorithms to classify sounds for hearing aids [1], but classifiers are usually assumed to be factory-trained. An emerging technology is the incorporation of feedback provided by the user. We consider a sub-task of such problems, the classification of music [2] into two classes (such as like-dislike) according to user preference. Algorithms should satisfy a number of requirements:

1. Online adaptation: The classifier may come with a factory setting, but has to adapt to the preferences of an individual user, preference changes and new types of music.
2. Sparse feedback: A user cannot be expected to provide a constant stream of labels.
3. Passivity: The user can provide feedback to express discontent with current performance. Hence, unless at least some feedback is received, the classifier should remain unchanged.
4. Efficiency: Feature extraction, training and data classification have to be performed online by a portable device.

To address the adaptation and online problems, we propose a classification algorithm based on additive expert ensembles [3]. Predictions of a fixed number of classifiers are combined by weighted majority. The weights are updated at each iteration, such that well-performing classifiers make large contributions. To cope with the sparse feedback problem, we show how the online learning algorithm can be combined with a label propagation algorithm for semi-supervised learning [4]. Music data are well-suited for semi-supervised methods, which attempt to improve classification performance by incorporating unlabeled data into the training process. The data distribution has to fulfill regularity assumptions for a successful transfer of label information from labeled to unlabeled points which holds for music data with similar types of instrumentation. Training a classifier to separate preferred from non-preferred classes results in a preference structure of considerably finer resolution than the common genre classifications. Experimental results show that the proposed classifier meets the requirements: It can adjust to both new music and changes in preference. Moreover, incorporating unlabeled data by label propagation significantly improves prediction performance when labels are sparse.

2. BACKGROUND

Online Learning. Most supervised learning algorithms operate under a batch assumption: A complete, static set of training data is assumed to be available prior to prediction. Additionally, at least for theoretical analysis, training data is assumed to be i.i.d., conditional on the class. Online learning [5] generalizes this scenario by assuming data points to be available one at a time, with each observation serving first as test, and then as training point. For a new data value, a prediction is made. After prediction, a label is obtained, and the observation is included in the training set. These methods only assume that the complete data sequence is generated by the same instance of the generative process – if the process is restarted, the classifier has to be trained anew. The data is not required to be i.i.d. On the theoretical side, well-known concentration-of-measure bounds of standard supervised learning are replaced by guarantees on the algorithm's performance relative to an optimal adversary, operating under identical conditions. In an i.i.d. batch scenario, online learning algorithms must be expected to perform worse than a well-chosen batch learner, but they are capable of dealing with both incrementally available data and data distributions that change over time.

Semi-supervised learning. In semi-supervised learning [4], the system is presented with both labeled data, denoted X_L, and unlabeled data X_U. The unlabeled data can provide valuable information for the training process. The risk (expected error) of a classifier in a given region of feature space is proportional to the local data density (under the commonly used, spatially uniform loss functions). To achieve low overall risk, a classifier should be most accurate in regions with high data density. Class density estimates obtained from unlabeled data can be used to inform training algorithms on where to focus. Unlabeled data is commonly exploited in either of two ways: Directly, e.g. by nonparametric density estimates used for risk estimation, or indirectly, by transferring labels from labeled to unlabeled

data. Both approaches are based on the notion that points sufficiently "close" to each other are likely to belong to the same class, which implies regularity assumptions on the class distributions: One is that the individual class densities are sufficiently smooth. The other is that classes are well-separated, that is, the density in overlap regions is small (and hence has small risk contribution). If these are not satisfied, unlabeled data should be used with care, as it may be detrimental to system performance.

3. ONLINE LEARNING WITH RANDOM PARTIAL INFORMATION

The learning problem described in the introduction is formalized as follows: We start with a baseline classifier (factory setting). New data values x_t (sound features) are provided sequentially. Some of these observations are labeled by the user as $y_t \in \{-1, +1\}$. The feedback label y_t is assumed to be available between observations x_t and x_{t+1}. If no feedback is provided, then $y_t = 0$. Changes in the input data distribution may occur, representing two cases:

- New concept: Data with a distribution not previously used in training is introduced.
- Concept change: Labels are contradictory to previous ones.

The online aspect of the learning problem is addressed by means of an additive expert ensemble [3]. The overall classifier is a weighted ensemble of up to $K_{\max}$ experts (component classifiers), denoted $\eta_{t,k}$ for time step t and component k. The experts are combined as a linear combination with non-negative weights. Given a new, labeled observation (x_{t+1}, y_{t+1}), the algorithm adjusts the classifier weights according to current error rates of the experts. Components performing well on the current data set receive large weights. Additionally, new experts are introduced, and poor performing experts are discarded to bound the total number K_t of components by $K_{\max}$. As the application scenario requires a bounded memory footprint, previously observed data cannot be stored indefinitely. We therefore window the learning algorithm, that is, updates in each round are performed on a moving window of constant size. Knowledge obtained from observations in previous rounds is stored only implicitly in the state of the classifier, until new, contradictory information votes against it.

Standard online learning algorithms adapt the classifier after each sample. We assume that feedback is provided only to change the state of the classifier. While the system is performing to the user's satisfaction, no feedback should be required. The learning algorithm therefore incorporates a passive update scheme: If no feedback is received, the classifier remains unchanged. The learning algorithm only acts if the current data point x_t is labeled by the user. In this case, observations in the current window up to x_t are used to change the classifier.

To integrate unlabeled data into the learning process, the online learning algorithm is combined with a semi-supervised approach. The method we employ is a graph-based approach for label transfer, a choice motivated in particular by the window-based online method. Since the window size limits the amount of data available at once, direct density estimation is not applicable. Graph-based methods are known for good performance on reasonably regular data. Their principal drawback, quadratic scaling with the number of observations, is eliminated by the constant window size. The particular method used here is learning with local and global consistency (LLGC) [6]. Data points are regarded as nodes of a fully connected graph. Edges are encode pairwise similarities (exponential of the negative Euclidean distance). In large-sample scenarios, the computational burden for fully connected graphs is often prohibitive, but in combination with the (windowed) online algorithm, the graph size is constantly bounded. LLGC spreads label information from labeled to unlabeled points by a discrete diffusion process along the graph edges. The diffusion operator in Euclidean space is discretized according to the graph's notion of distance by the normalized graph Laplacian L. The latter is computed from the graph's affinity matrix W and diagonal degree matrix D. The entries of W are pairwise similarities, and D is computed as $D_{ii} := \sum_i W_{ij}$. The normalized graph Laplacian is then defined as $L := D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$.

For each sample x_t, the algorithm executes a prediction step, then possibly obtains a label either as user feedback or by LLGC, and finally executes a learning step. It takes three scalar input parameters: A trade-off parameter $\alpha \in [0, 1)$ controls how rapidly label information is transferred along the edges during the propagation step. For the learning step, $\beta \in [0, 1]$ and $\gamma \in \mathbb{R}_+$ control the decrease of expert weights and the coefficients of new experts, respectively. The prediction step for x_t is

1. Get expert predictions $\eta_{t,1}, ..., \eta_{t,N_t} \in \{-1, +1\}$.
2. Output prediction: $\hat{y}_t = \arg\max_{c \in Y} \sum_{i=1}^{N_t} w_{t,i} \mathbb{I}[c = \eta_{t,i}]$

Learning step (only if $y_t \neq 0$): The algorithm first propagates labels to unlabeled points, and then updates the classifier ensemble. The graph Laplacian L_t is updated for current window index t, for which the corresponding labels are $Y_t = (y_{t-\tau+1}, ..., y_t)'$.

1. **Propagation:**
   (a) Initialize estimate vector as $\hat{Y}_t^{(0)} = Y_t$
   (b) Iterate $\hat{Y}_t^{(j+1)} = \alpha L_t \hat{Y}_t^{(j)} + (1 - \alpha)\hat{Y}_t^{(0)}$
   (c) Assign each x_i the label given by sign($\hat{y}_i^{\text{final}}$)
2. **Learning:**
   (a) Update expert weights: $w_{t+1,i} = w_{t,i} \beta^{[y_t \neq \eta_{t,i}]}$
   (b) If $\hat{y}_t \neq y_t$, then add a new expert N_{t+1} (and eliminate expert with lowest weight), where the new expert is trained on the current window of data: $w_{t+1,N_{t+1}} = \gamma \sum_{i=1}^{N_t} w_{t,i}$
   (c) Update each expert on example x_t,y_t

Due to the limited window size, LLGC is efficient, and run until equilibration. The first step interpolates the label of each unlabeled point from all other nodes. Due to similarity-weighted edges, only points close in feature space have a significant effect. Further steps correspond to longer-range correlations, i.e. affecting nodes over paths of length 2, 3 etc. Allowing the graph to equilibrate therefore improves the quality of results for uneven distribution of labels in feature space. Once the propagation step terminates, class assignments for the unlabeled input points are determined by the polarity of their accumulated mass. The resulting hypothesized labels are presented to the classifier ensemble as "true" labels.

4. EXPERIMENTS

For evaluation, we built a music database of 2000 files. The bulk of the database are "classical music": opera (Händel, Mozart, Verdi and Wagner), orchestral music (Beethoven, Haydn, Mahler, Mozart, Shostakovitch) and chamber music (piano, violin sonatas, and string quartets). A small set of pop music was also included to serve as "dissimilar" music.

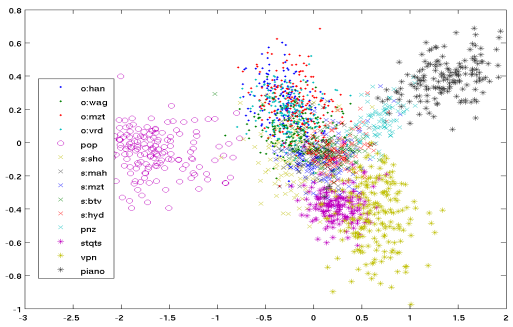

Fig. 1. Visualization of data onto two-dimensional space using Fisher LDA. We observe substructures in the data and smooth transitions, e.g. from piano to piano concertos (pnz) to symphonies (s:*). The categories pop, opera (o:*) and chamber music are identifiable.

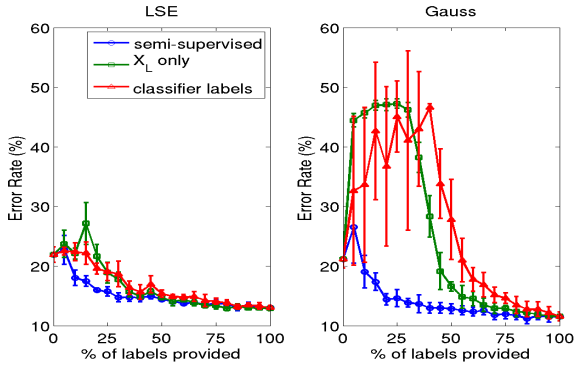

Fig. 2. Cumulative Errors on learning concept changes versus ratio (percentage) of available labels, shown for LSE (left) and Gaussian (right) experts. Results are obtained by five-fold cross validation. Error rates at 0% correspond to the initial static classifier. The peak in error rates is discussed in Sec. 4.2.

Features are computed from 20480 Hz mono channel raw sources. We compute means of 12 MFCC components [7] and their first derivatives, as well as means and variances of zero crossing, spectral center of gravity, spectral rolloff, and spectral flux. In total we obtain a 32-dimensional feature vector per file. Fig. 1 shows a 2-D Fisher linear discriminant analysis (LDA) projection of features averaged over each song or track (i.e. one point per track in the plot). Since the current study focuses on the classification algorithm, we do not consider higher-level features [8].

Results reported here use signatures of complete songs. A real-world application would, of course, have to use partial signatures, such that the system can react to new music without long delays. Reference experiments with a static classifier show that between 20 and 60 seconds of music are required to obtain a reliable classification for the current features. Strategies for allowing the classifier to act early are beyond the scope of this article; cf Sec. 5 for a brief discussion.

4.1. Classifier Setting

The additive expert is based on an ensemble of simple component classifiers. Two types components were used in the experiments: A least mean-squared error (LSE) classifier, and a full covariance Gaussian model (GM). The decision surfaces of the individual components are hyperplanes in the LSE case, and quadratic hypersurfaces for the GM. (Using a Gaussian mixture instead of an individual Gaussian for each class proved not to be useful in preliminary experiments.) The two principal differences between the two classifiers are the fact that the GM constitutes a generative model, whereas the LSE model does not, and that the GM is more powerful. The set of hyperplanes expressible in terms of LSE is included in the GM as a special case. Higher expressive power comes at the price of higher model complexity (in d-dimensional space, the GM estimates $2 * (d + \frac{d(d+1)}{2})$ parameters, compared to $d + 1$ for the LSE).

A baseline model is first learned on an initial set of data. During the evaluation phase, the remaining data is presented to the classifier sequentially. When no labels are provided, the classifier does not update, such that values reported for 0% shows the performance of a static baseline classifier. When all labels are provided, we obtain the conventional, fully supervised online learning scenario. For both choices of experts, we compare the semi-supervised online algorithm to two other learning strategies. The three variants shown in each of the diagrams are:

1. X_U takes the label hypothesized by LLGC (semi-supervised).
2. X_U is ignored and not used for learning (X_L only).
3. X_U takes the label hypothesized by the current classifier (classifier labels).

Results report cumulative error on the evaluation data, i.e. if $\hat{y}_t$ denotes the label predicted by the classifier for x_t, the error is given as $\text{Err} = \frac{1}{T}\sum_{t=1}^{T}[\hat{y}_t \neq y_t]$.

4.2. Experimental Results

Results are presented separately for two mismatch scenarios: change of concepts, and introduction of new concepts. The experiments simulate behavior in adaptation phases. During normal operation, the user need not provide any labels. Since the classifier is passive, user action is required only in order to prompt the system to adapt.

Learning a changed concept. The baseline model is trained on 2 sets consisting of subclusters {o:*, pop} and {s:*, strqts,pno}. During the evaluation phase, subclusters s:mah, s:sho and pop are reassigned to the opposite classes. Fig. 2 shows the results for both GM and LSE models. When the proportion of label data is low, using the unseen labels via LLGC significantly improves system performance. In all experiments conducted, the semi-supervised algorithm consistently outperforms the other approaches until at least about 80% of labels are available. The error rate at 0% is the performance of the initial baseline system. Initially, for very small numbers of labels, overfitting to the labeled subset decreases prediction accuracy with respect to the baseline. Interestingly, for small label ratios, overfitting effects *increase* with the number of labels, until the error peaks and then decreases. More labeled points mean more adjustment steps, and therefore stronger overfitting if the available information is insufficient. Hence, the peaks in error rates are due a trade-off effect between the information provided by the labels and the number of learning steps they trigger. The decrease in performance is most notable for Gaussian experts, which are less robust than the LSE experts. In a real-world implementation, one would choose the

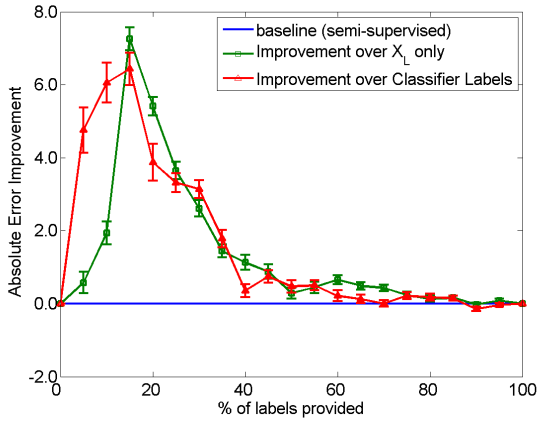

Fig. 3. Absolute performance improvement of semi-supervised system over comparison strategies (100 random runs).

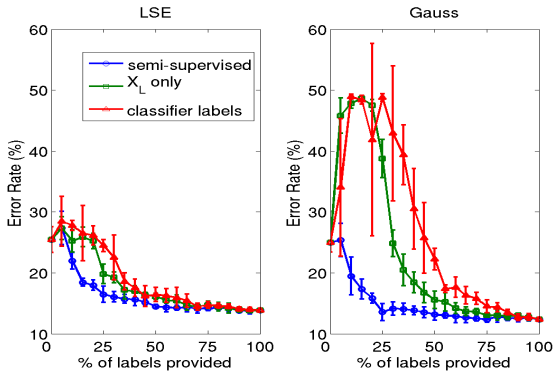

Fig. 4. Cumulative Errors on learning new concepts

baseline classifier until a minimum ratio of labels is available. While the semi-supervised approach requires about 10% of labels to start improving upon the baseline method, between 20% (LSE) and 40% (Gaussian) are required if the unlabeled data is neglected. At large label ratios, the Gaussian model slightly outperforms the LSE. The semi-supervised version of the model requires only about 40% of labels to reach optimal performance.

To evaluate the average behavior of the system when the change of concept is not hand-picked, we generated 100 random runs of groupings of the subclusters. For each case, four subclusters reverse their labels during evaluation phase. Fig. 3 plots the absolute improvement in error rates of the semi-supervised method over the two comparison classifiers, showing behavior consistent with the results in Fig. 2.

Learning a new concept. The second type of classifier adaptation is adjustment to previously unobserved music. Of particular interest is the classifiers behavior when the new concept substantially differs from those already incorporated in the baseline model. In this experiment, the baseline model is trained on opera, {o:*}, and classical orchestral/chamber music. During the evaluation phase, "modern" music (Mahler and piano) are assigned to the opera class, and pop music and Shostakovitch to the other class. Fig. 4 shows the results for the LSE classifier. As in the concept change case, the amount of feedback required by online learning with LLGC is substantially reduced with respect to the fully supervised method.

5. DISCUSSIONS AND CONCLUSIONS

We have presented an algorithm for music preference learning that combines an online approach to learning with a partial label scenario. The classifier is capable of tracking changes in class distributions and adapting to data that differs from previous observations, in reaction to user feedback. Due to the integration of unlabeled data in the learning process, only partial feedback is required for the classifier to achieve satisfactory performance. The algorithm remains passive unless user feedback triggers an adaptation step. A window-based design limits both computational costs and memory requirements in an economically feasible range.

A step towards applicability in a real-world scenario will require the incorporation of strategies that enable the algorithm to classify a new piece of music as early as possible. Acoustic features should be chosen accordingly. Adaptation speed has to be traded of against reliability, to prevent the device from oscillating back and forth due to initially unreliable estimates. Since different types of music are more or less quickly recognizable, one may consider estimating reliability scores for classification results to control changes in the current control program of the system.

Our algorithm design does not make any assumptions about the base learner. In principle, any classification algorithm may be used, e.g., the proposed algorithm may be extended by kernelization of the LSE base learner, which generalizes decision boundaries beyond the linear case. We expect our method to be a step towards adaptivity in the control of "smart" hearing devices.

Acknowledgement. This work is funded by KTI, Nr8539.2;2EPSS-ES.

6. REFERENCES

[1] M. Büchler, S. Allegro, S. Launer, and N. Dillier, "Sound classification in hearing aids inspired by auditory scene anlysis," *Journal of Applied Signal Processing*, vol. 18, pp. 2991–3002, 2005.

[2] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Trans. on Speech and Audio Processing*, vol. 10, no. 5, 2002.

[3] J. Z. Zolter and M. A. Maloof, "Using additive expert ensembles to cope with concept drift," in *Proceedings of the 22nd Intl Conference on Machine Learning*, 2005.

[4] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*, MIT Press, 2006.

[5] N. Cesa-Bianchi and G. Lugosi, *Prediction, learning and games*, Cambridge University Press, 2006.

[6] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in Neural Infomation Processing Systems*. 2004, vol. 16, pp. 321–328, MIT Press.

[7] D. P. W. Ellis, "PLP and RASTA (and MFCC, and inversion) in Matlab," 2005, online web resource.

[8] G. Tzanetakis and P. Cook, "Marsyas: A framework for audio analysis," *Organised Sound*, vol. 4, no. 3, 2000.