2014 13th International Conference on Machine Learning and Applications

Speeding Learning of Personalized Audio Equalization

Bongjun Kim and Bryan Pardo
Electrical Engineering and Computer science
Northwestern University
Evanston, USA
bongjunkim2013@u.northwestern.edu, pardo@northwestern.edu

***Abstract*— Audio equalizers (EQs) are perhaps the most commonly used tools used in audio production. The SocialEQ project is a web-based personalized audio equalization system that uses an alternative interface paradigm to the standard approach. Here, the user names a desired effect (e.g. make the sound "warm") and teaches the tool (e.g. an equalizer) what settings make the sound embody the term. SocialEQ typically requires 25 ratings to properly personalize the equalization settings. In this paper, we present three methods to improve the speed of generating personalized items (audio settings) so users can be provided personalized EQ curves after rating a much smaller number of examples. These methods can be adapted to any situation where collaborative filtering is desirable, the end products created for users are unique and comparable to each other, but prior users did not rate the same set of examples as the current user. Methods are tested on a data set of 1635 user sessions.**

Keywords-audio equalizer; transfer learning; collaborative filtering; personalized item.

I. Introduction

Media production tools, such as audio equalizers are widely used in music production, video production, radio production, etc. In the past, these tools were typically used by expert professional engineers. Today, there is a paradigm shift and everyone is producing media for web distribution and sharing (e.g. Youtube, Soundcloud, Bandcamp). Therefore, the need for media production and manipulation software that even a non-expert can easily use has increased.

Audio equalizers (EQs) are commonly used tools used in audio production. They selectively boost or cut restricted portions of the frequency spectrum, and in doing so alter the timbre of a sound. Fig. 1 shows a typical parametric equalizer. This tool has on the order of 24 knobs, 15 buttons and one wheel. The large number of controls makes it difficult for non-experts to use effectively.

Sabin et.al [1] developed an alternative interface paradigm where the user names a desired effect (e.g. make the sound "warm") and teaches the tool (e.g. an equalizer) what settings should be applied to make the sound embody the term. The combination of term ("warm") and settings (the boost/cut at each frequency band) is a user-concept (e.g. Bob's warm).

The *SocialEQ* [2] project is a web-based personalized audio equalization system using the method in [1]. Since it has been released on the web, over 3000 user-concepts have been taught to the system. Each user-concept in the dataset is learned by asking the user to rate a randomly-chosen 25 audio examples out of a set of 50 examples used for training the system. Even though the rating method is good approach to build a personalized audio object, it requires too many ratings (e.g. 25) from each user to achieve accurate results.

Figure 1. A parametric audio equalizer

To reduce the number of questions users need to answer, transfer learning was applied in [3]. The idea was to use prior knowledge to predict user ratings to unrated audio examples. In other words, after the current user rates a small number of examples, the system predicts the user's preference to the rest of the unrated examples by using prior user data. The method in [3] requires all users to rate the exact same set of examples so that distance between lists of user ratings can be directly measured. *SocialEQ* asks users to rate a randomly-selected 25 examples out of a set of 50 examples, so most users only overlap on a portion of their rated examples. Therefore the method must be modified so similarity between user-concepts can be measured when users have not rated the same examples.

In this paper, we present three ways to improve the speed of generating personalized items (audio settings). The first improves the learning algorithm in [1]. The next two methods speed personalization using prior user data, overcoming the limitation in [3]. One allows comparison between user-concepts without direct reference to the user ratings. The other is a new imputation method that fills in missing ratings so that user-concepts can be compared in terms of user ratings of items. All three methods can be adapted to any situation where collaborative filtering is desirable, the end products created for users are unique and comparable to each other, but prior users did not rate the same set of examples as the current user.

II. Related Work

One way we speed learning in this paper is related to memory-based collaborative filtering in recommender systems. It makes predictions of a user's unknown preference to items by analyzing the known preferences of other users [4]. One of the important problems in memory-based collaborative filtering is how to deal with a sparse user-item matrix to calculate accurate similarity between users. Yongli et al.[5] proposed a method to select the most informative missing data to impute for memory-based (neighbor-based) collaborative filtering. Hao et al.[6] presented a missing data prediction algorithm that exploits the information both from users and items. The work argues that if an item is very popular, the new user will be likely to give the item a good rating, so it uses both user correlation and item correlation to predict missing ratings. However, the argument may not apply in our case because items (i.e. audio examples) in this paper were generated on purpose to probe user's preference in relation to an audio concept and the same user may rate two presenations of the same example quite differently in light of two different audio concepts (e.g. "Is this tinny?" vs. "Is this dark").

Jeong et al.[7] defined user credit and converted each set of user ratings into the user credit. This makes it possible to measure similarity between users even with incomplete rating data. We take a similar approach in one of our methods to speed learning for equalization curves. Note that we are dealing with a situation distinct from those in general recommender system (e.g. movies or music recommendation). We do not just recommend an exiting item from the set of rated items (sound files manipulated by EQ), but rather create a new personalized item (an EQ setting) that embodies a user's audio concept.

III. Methods

In this section, we explain how an individualized EQ curve is built based on user ratings in the *SocialEQ* data using the original method in [1]. Then we describe an improvement to this learning method and two methods to use prior user data to speed learning, one which relies on imputation of missing user data and one which does not.

A. The SocialEQ data set

The *SocialEQ* data set has 3369 sessions. A single session contains 25 ratings of known examples from a single user. In each session, the user selects a concept word (e.g. "tinny") and rates examples on a scale from -1 to 1. A user rating of 1 means the audio example perfectly matches the sound that the user wants (e.g. "very tinny") and a user rating of -1 means the example has the opposite sense ("very NOT tinny").

An audio example is a sound file that has been modified by equalization setting that specifies a boost or cut of the amplitude at each of 40 frequencies. This setting is represented as a curve (an EQ curve) with 40 data points (*gains, frequency bands*) representing log-spaced frequencies from 20 Hz to 19682 Hz (Fig. 2). When a user gives a rating r_1 for an audio example, we get a rating value for each of the 40 gains at these 40 frequencies. If a user rates, for instance, 25 audio examples, 25 (*rating, gain*) data points for every frequency band are generated. Within a session all examples use EQ curves applied to the same audio file. All EQ curves for every session were drawn from an underlying set of 50 EQ curves.

We filtered sessions with the inclusion criteria used in [2] so that only session from people who put in effort and rated examples consistently were used in our study. We removed all sessions where the participant took less than 60 seconds to complete the task. We also removed all sessions where the participant gave the default rating for more than 5 out of the 40 examples. We also removed any session where the participant responded "no" to the survey question: "Was the listening environment quiet?" Finally, 15 of the 40 examples each participant rated were repeat examples. This let us test for consistency of user responses. We measured consistency using Pearson correlation between each participant's rating of the first presentation of an example with that of the second presentation of the example. A participant was excluded if their consistency fell more than one standard deviation below the mean across all participants. After filtering out the low quality data we have 1635 prior sessions. All results in this paper use this set of 1635 sessions.

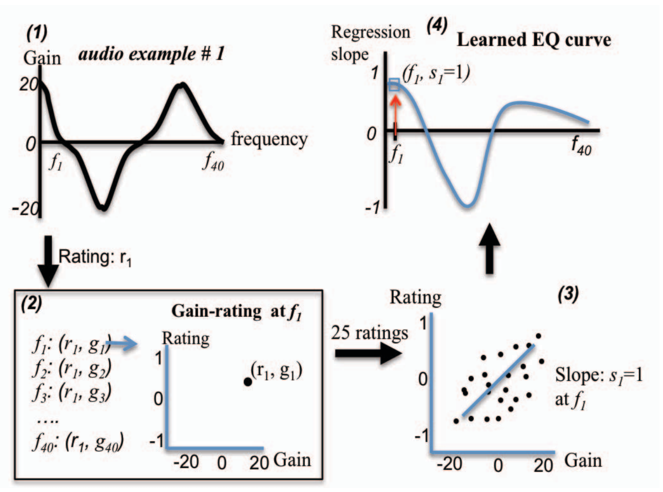

Figure 2. An overview of the process of the baseline learning method

B. The Baseline Learning Method

A user-concept is the ideal EQ curve that would modify sound to embody the desired word (i.e. the perfect "tinny" EQ). In this work, we assume that the baseline learning method from [1] perfectly learns the user-concept when given 25 rated examples in a session from the *SocialEQ* data.

Fig. 2 shows an overview of the *baseline* learning method from [1]. The input is the set of 25 rated 40-band EQ curves in a session. For each frequency band we calculate a regression slope from the 25 ratings. The regression slope at each frequency band represents relative gain of EQ at the frequency that the user would prefer. This results in a 40-point EQ curve that we call the *user-concept.* For a more detailed explanation about the EQ building process, see [1].

C. *Speed Learning with Reestimation*

An estimate of the user-concept built with the baseline method can be unreliable if learned from only a few rated examples (e.g. 5 examples, instead of 25). An estimate learned from fewer ratings can, however, be used as the input to predict ratings on examples the user has not yet rated. We can then use the combination of real and estimated ratings to reestimate the user-concept. The method is as follow: we obtain the EQ curve genearted from n user ratings (e.g. 5 ratings), using the baseline method. Next, we predict ratings to the rest of unrated examples (the remaining 20 unrated examples) by calculating the Pearson correlation between the estimated user-concept EQ curve and the EQ curve for each unrated audio example. We use this correlation coefficient as an estimated rating. We then reestimate the user-concept EQ curve by building a new curve using both the estimated (e.g. 20 estimates) and actual (e.g. 5) ratings back into the baseline method. We call this method *reestimation*.

D. *Transfer learning and active learning*

Transfer learning [8] is an effective learning method to speed concept learning through using prior knowledge from previously learned tasks. We can further improve our estimate of the output EQ curve by using prior user data (user-concepts) to augment the information gained from the current user's ratings. The idea is that if two user-concepts are close to each other in terms of either the ratings users gave to examples or in terms of the EQ curve learned from the partial set of ratings (even after only a few ratings), the resulting user-concept EQ curves learned after all 25 ratings should also be similar. When a user rates n examples, we measure similarities between the current user-concept and prior user-concepts in n-dimensional rating space or 40-dimensional EQ space. Once we have the similarity between the current user-concept and all prior user-concepts, we can estimate the current user's ratings of currently-unrated examples by using a K-nearest neighbor linear combination of prior data, weighted by similarity to the current user-concept. As a similarity measure, in this work, Pearson correlation is used because it accounts for scaling differences between users (i.e. user A rates everything in the range 1 to -1 and user B rates things in the range 0.01 to -0.01). We select the 64 closest prior user-concepts because a pilot study showed this number was optimal for generating good estimates.

Active Learning refers to the case where the learner selects the examples to learn from, rather than passively receiving examples chosen by the teacher. We apply the active learning method used in [9], which is to present examples to the user ordered by the variance in how those examples were rated across all prior users. This lets us quickly differentiate between prior user-concepts and locate the current user's concept in the space of prior user-concepts. This approach was selected for simplicity, effectiveness and consistency with the prior work we build upon. A comparison to other approaches is outside the scope of this paper. We now describe two methods of applying K-nearest neighbor estimation.

E. *Using Prior Data with Missing Values without need for imputation of missing values*

In our case, the equalization curves learned for any two users can be directly compared, even though they were generated from different sets of rated objects. Therefore, instead of filling in estimates for "missing" ratings so that all prior users can be compared, we create an EQ curve from the current user's set of ratings (even if they have only rated a few examples) and compare that curve to the EQ curves learned for each prior user. Note, prior user EQ curves are learned with the baseline method from all 25 examples in the session. This lets us apply data from prior users, even if they rated completely different sets of example EQ settings.

From the current user's ratings, we build an EQ curve from n ratings using the baseline method and measure the similarity of this (admittedly bad) EQ curve, to each of the EQ curves learned from previous users. This is done with Pearson correlation. We then create a composite EQ curve for the current user from the 64 closest EQ curves from prior users. The weight of each prior user's user-concept EQ curve is proportional to its similarity to the current user's curve.

F. *Imputation of Missing Values in Rating Data Set*

In this method, we compare users based on their ratings of examples, rather than based on the final EQ curve estimated for each user-concept. Users in the *SocialEQ* data set have each rated 25 randomly-selected examples out of a set of 50 from which the examples were drawn for all sessions. Therefore, to make prior user-concepts comparable, we must fill in (impute) missing ratings. Once we impute ratings for all unrated examples, we directly compare user-concepts in rating space. The objective is to estimate user ratings to the 25 unrated examples, so an EQ curve learned from the imputed rating set (including actual 25 and estimated 25 ratings) is identical to the EQ curve learned from just the actual 25 user ratings.

Suppose we predict a user rating r to the example i. To find a missing rating that does not change a previously learned EQ curve, we use the slopes of linear regression lines that can be derived from existing ratings. Since we already know gains over frequencies of all audio examples, we can calculate the estimate of a missing rating to an example using (1).

$$\hat{r} = \frac{1}{n}\sum_{i=1}^{n} s_i g_i + b_i \qquad (1)$$

,where s_i and b_i are the slope and intercept of regression line at frequency band f_i of the original EQ curve, g_i is the gain at the frequency, and n is the number of frequency bands. We perform imputation for all unrated examples and all prior sessions so that every session has ratings for all 50 possible examples.

To calculate the weight prior sessions have on learning a new concept we measure similarity between the current user's ratings and prior user ratings. Let $s(u, v)$ be similarity (Pearson correlation) between the new user-concept u and each prior user-concept v. The weight of the prior user-concept v can be derived by (2).

$$w(v) = \frac{s(u,v)+1}{\sum_{v\in U}(s(u,v)+1)}, u \neq v, \qquad (2)$$

where U is a set of prior user-concepts. In this work, we limit U to be the 64 user-concepts whose ratings of examples are most similar to the current user's ratings.

$$\widehat{r_u}(q) = \sum_{v\in U} w(v) \cdot r_k(q) \qquad (3)$$

Based on the weights, the current user's future ratings to the audio example q are calculated by (3) which is the weighted sum of prior ratings.

IV. The Experiment

The purpose of this work is to reduce the number of rated examples needed to estimate a user's desired audio concept. To do that, we first evaluate the imputation technique described in Section III-F by comparing other common imputation methods. We then measure the learning speed of each method described in Section III.

A. Data imputation evaluation

We compared the imputation method in Section III-*F.* (labeled *Reg*, for regression) to three widely-used techniques to impute missing values. These are: 1) Pearson correlation coefficient between a user's leanred EQ curve and unrated audio examples; (*Corr*) 2) mean value of rating to certain examples across all prior users (*Mean*); and 3) Matrix Factorization (*MF*) [10].

Fig. 3 shows the pairwise Pearson correlation between EQ curves learned from 25 actual ratings to EQ curves generated from 25 actual ratings + 25 imputed ratings on 1635 user-concepts from the SocialEQ data set. Similarity 1.0 means that EQ curve learned from data including imputed ratings is identical to original EQ curves learned from only the 25 actual ratings. The mean similarity values for the two methods that show the best performance (*Reg* and *Corr*) are 0.992 and 0.969, respectively. More importantly, the figure shows that our method results in more stable and accurate performance.

Even in the worst case, correlation is above 0.95, while the other three methods have large variance in their correlation. This means that our imputation method is much less likely than existing methods to skew the outcome by creating bad data estimates. Therefore, it is the best method to let us use prior data in transfer learning.

B. Learning methods evaluation

The methods described in Section III were designed to let the system learn user-concepts with fewer ratings than required by the baseline system. We measure the ability of each approach to speed learning as follow: First, we select one of the 1635 prior sessions from the *SocialEQ* data. Each session has 25 rated examples. Then, we select n audio examples rated by the user in that session. We then generate an EQ curve to represent the user-concept using each of the methods from Section III. To measure the correctness of each estimated EQ curve, the estimated EQ curve is compared to the actual EQ generated from the user's full set of rated examples for that session. For the comparison, Pearson correlation was used and the correlation coefficient is called *machine-user correlation*. We perform the simulation for all 1635 prior sessions and summarized the 1635 the machine-user correlation values by taking the mean.

Fig. 4 shows the mean machine-correlation as a function of the number of rated examples n, comparing all learning methods. All three methods outperformed the baseline method. Using our imputation technique and measuring similarities to the imputed prior data in rating space was the best solution tested. For example, when we use the method, a correlation of 0.8 to the final EQ curve learned from 25 rated examples can be achieved using only 7 ratings. But the baseline method requires 13 ratings for the same level of correlation.

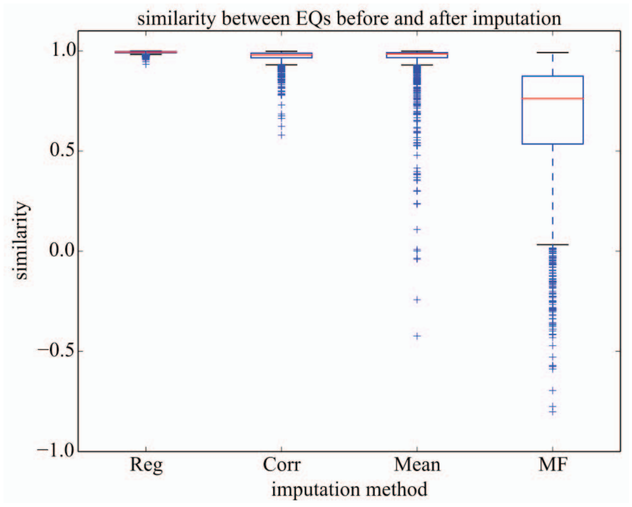

Figure 3. Simlarity between EQ curves before and after imputation

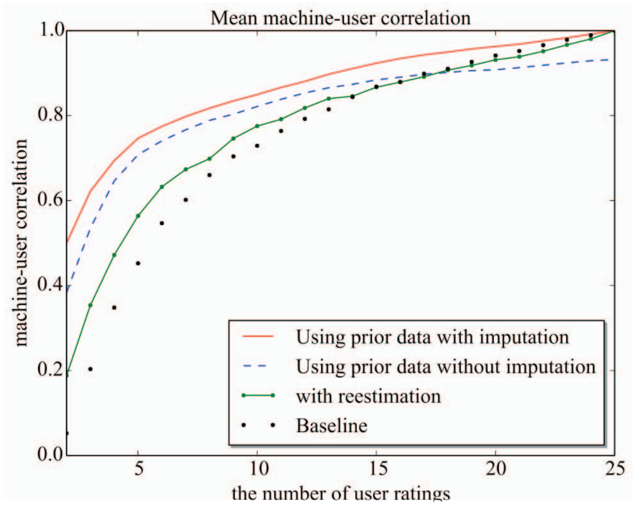

Figure 4. The mean machine-user correlation comparing all learning methods

V. Dynamic Data Environments

The prior analysis of algorithms had the implicit assumption that the database of prior users remains constant. We now revisit our algorithms and consider the case where the database of prior users is growing over time. Consider the case where the nth user's data (responses to examples and learned EQ curve) has just been added to the database. We now wish to find an EQ curve for user $n+1$. What is the time complexity to learn an EQ curve for user $n+1$?

The baseline method described in III-B takes no prior user data into account and therefore its time complexity is constant with respect to the number of prior users n. The same applies to the reestimation method in III-C. When transfer learning is applied (Section III-D) a distance must be determined between the current user and all prior users. This is clearly $O(n)$. It can, however, be reduced to something approximating $O(\log(n))$ through the use of database organizational techniques, such as vantage point trees [11].

Our imputation method presented in Section III-F only needs data from one user-concept to impute missing values of that user's ratings. It does not depend on prior users' data and therefore takes constant time with respect to n. Therefore the algorithm itself does not need to be modified in the dynamic situation even though it requires more time to compute the result. The commonly-used matrix factorization technique, on the other hand, requires a refactorization of the full dataset matrix each time a new user is added.

VI. Conclusions

We presented three methods to reduce the number of ratings required for creating a personalized audio EQ curve and tested the methods with *SocialEQ* data set. Each approach showed improvement over the baseline learning method. When prior data is unavailable, then our reestimation method in Section III-C improves learning. If prior data is available and imputation of missing data is infeasible, Section III-E describes a method that improves learning even more. If missing ratings can be reliably estimated, then our approach in Section III-F further improves learning. What is more, for our data set, the imputation method in Section III-F is novel and is significantly better in estimating missing values than existing techniques, such as Matrix Factorization. This work can be adapted to any situation where a system generates a personalized item based on user-ratings to sample items.

Acknowledgment

This research was supported by grant 1116384 from the US National Science Foundation.

References

[1] A. T. Sabin, Z. Rafii, and B. Pardo, "Weighting-function–based rapid mapping of descriptors to audio processing parameters," *Journal of the Audio Engineering Society,* vol. 59, pp. 419-430, 2011.

[2] M. Cartwright and B. Pardo, "Social-eq: Crowdsourcing an equalization descriptor map," *Proc. of International Society for Music Information Retrieval (Curitiba, Brazil, 2013),* 2013.

[3] D. L. B. Pardo, and D. Gergle, "Towards speeding audio EQ interface building with transfer learning," presented at the Proceedings of NIME 2012 - 12th International Conference on New Interfaces for Musical Expression, 2012.

[4] M. Z. Dietmar Jannach, Alexander Felfernig, Gerhard Friedrich, *Recommender Systems: An Introduction*: Cambridge, 2010.

[5] Y. Ren, G. Li, J. Zhang, and W. Zhou, "The efficient imputation method for neighborhood-based collaborative filtering," in Proceedings *of the 21st ACM international conference on Information and knowledge management*, 2012, pp. 684-693.

[6] H. Ma, I. King, and M. R. Lyu, "Effective missing data prediction for collaborative filtering," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007, pp. 39-46.

[7] B. Jeong, J. Lee, and H. Cho, "User credit-based collaborative filtering," *Expert Systems with Applications,* vol. 36, pp. 7309-7312, 2009.

[8] S. J. Pan and Q. Yang, "A survey on transfer learning," *Knowledge* and *Data Engineering, IEEE Transactions on,* vol. 22, pp. 1345-1359, 2010.

[9] P. Bryan, L. David, and G. Darren, "Building a personalized audio equalizer interface with transfer learning and active learning," presented at the Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies, Nara, Japan, 2012.

[10] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer,* vol. 42, pp. 30-37, 2009.

[11] P. N. Yianilos, "Data structures and algorithms for nearest neighbor search in general metric spaces," in *Proceedings of the fourth* annual *ACM-SIAM Symposium on Discrete algorithms*, 1993, pp. 311-321