Automatic Group-Interactive Radio using Social-Networks of Musicians

Ben Fields, Christophe Rhodes and Mark d'Inverno
Department of Computing
Goldsmiths University of London
London, United Kingdom
contact: me@benfields.net

Abstract

Using request radio shows as a base interactive model, we present the Steerable Optimizing Self-Organized Radio (SoSoRadio) system as a prototypical music recommender system with robust automatic playlist generation. This work describes a web-based radio system that interacts with current listeners through the selection of periodic request songs from a pool of nominees.

1 Introduction

The way music is distributed and consumed has, in the last decade, undergone a revolution as significant as that which occurred around the first successful commercial distribution of recorded music. This change has been driven by the massive reduction in the cost of distribution of information brought about by the wide-spread adoption of the internet in general and the Web in particular. In the domain of music this has led to music consumers and listeners being given immediate access to tremendously large catalogues of music at any time, overtaking the previous model where music is purchased to build a personal collection, and personal listening choices are then drawn from this relatively small personal collection. This new environment of catalogues of 10^8 songs rather than personal collections of 10^3 songs presents new challenges for listeners. In this paper we present a new social media application which includes a novel interaction technique in the way ratings are interpreted.

We confront the issue of how to playback songs to best sustain user interest, exposing the underlying question: just how automatic can this task be?

Our hypothesis is that we can achieve this aim by encouraging and eliciting user interaction and through *controlled variation* (a middle ground between sets of very similar and maximally heterogenous objects) of retrieved material via musical content from one song to the next and by an awareness of social information connecting songs (e.g. artist, label, genre) To test this hypothesis we present a deployed system that elicits feedback from its audience, and uses that feedback to direct searching for and ordering of relevant songs based both on content and social data. We test the fitness of our *controlled variation* (using statistical analysis) and lay the foundations for subjective user analysis of these methods.

Our contributions are both domain specific and generally applicable and include:

- a radio station that uses both content and social data for the first time;
- a testing framework for measuring the fitness of controlled variance amongst a collection of playlists; and
- a proposed rating measurement scheme that is tied to paths (i.e. transitions between tracks) as opposed to individual tracks.

We detail a fully-automatic, interactive radio system, designed to put the lessons of the analysis first described in (Fields et al. 2011) into practice, within an interactive, user-centric, group-playback application.

2 Background

When considering the source of dataset for a music system, there are two distinct sources: user generated content (UGC) where there is a network around that content (e.g. Myspace, Soundcloud, Youtube) versus curated 'conventional' record-store type datasets, organised into flat taxonomies (e.g. genre) to maximize visibility (e.g. iTunes Music Store, 7digital, HMV, etc...). We require a dataset that encodes both music content and social data about the producers of that content, the necessary information to enable our testing. Thus a UGC network is required. For this work we use Myspace, which at the time the dataset was collected was the defacto standard for web-based music artist promotion. While the sample driving the prototypical deployment comes from Myspace, the techniques described in this paper can be generalized to any social network system involving music.

It is important to note we are only concerned with a subset of the Myspace social network – the Myspace *artist* network. Myspace artist pages are different from standard Myspace pages in that they include a distinct audio player application containing material uploaded by that user. We therefore use the presence or absence of this player to determine whether or not a given page is an artist page where *artist page* is used to refer to the collection of social links and audio material assumed to be generated by the same person or group of people. For our sampling we only encode and follow the social links of 'top friends', those connections that

	n	m	$\langle k \rangle$	l	d_{max}
undirected	15478	91326	11.801	4.479	9
directed	15478	120487	15.569	6.426	16

Table 1: The network statistics for the Myspace artist network sample where n is the number of nodes, m is the number of edges, $\langle k \rangle$ is the average degree, l is the mean geodesic distance, and d_{max} is the diameter.

are unilaterally promoted by a user, implying greater significance and less social noise.

In addition to these social connections, we also gather metadata about each artist including: the name of the artist, the number of page views, and genre labels associated with the artists. The audio files associated with each artist page in the sampled network are also collected for feature extraction. Note that genre tags collected are at the level of artists, rather than audio files; therefore all audio files associated with that artist will have the same genre labels applied.

There are several network sampling methods; however, for networks like the Myspace artist network, snowball sampling is the most appropriate. The sample begins with a seed node (artist page), then the seed node's neighbours (top friends), then the neighbours' neighbours, are added to the sample. Here, we randomly select a seed artist[1] and collect all artist nodes within 6 edges to collect 15,478 nodes. If the size of the Myspace artist network is around 7 million, then this is close to the 0.25% sampling ratio suggested for accurate degree distribution estimation in sampled networks. Note that the sampling ratio is not sufficient for estimating other topological metrics such as the clustering coefficient and assortativity; but such global measures are not required for this work.

The Myspace artist network sample exhibits many of the network characteristics common to social networks and other real-world networks and our sample is similar in composition to other samples taken. Basic statistics of the sample network are summarized in Table 1.

Unlike in other work based on Myspace samples, we set the UGC (in our case songs) as the nodes in the graph with the top friend connections between the creators of works used to define edges. These edges are then weighted based on a song-to-song timbre similarity measure using Mel-Frequency Cepstral Coeffiecients (MFCC). More detail of this process and a complete analysis of this dataset is presented in other work (Fields et al. 2011; Jacobson and Sandler 2008).

Before discussing the particulars of our system, we will first highlight approaches to generate playlists which better meet the needs and requirements of the listener. We consider: input given to a playlist generator is a *query*; similarly, generated playlists are *retrieved results* so then their *relevance* to the query is measurable.

Playlists are vastly improved by the additional information provided by including a second song in a query system. Although this is not the only method of further query specification, the critical point is that any viable novel system must have a means for the user to specify not only a starting position but the initial trajectory as well. Our particular model, as described in the next section, does this through an awareness of the current song playing and asking users for songs to play in the future. In this way we give our automatic system a sense of *direction*.

3 Our Interactivity Model

The inspiration for the interactive model we employ in SoSoRadio comes from terrestrial radio music programs and specifically *periodic request* shows, that mix songs requested by the audience with the presenter's own selections. In these shows, listeners would contact the show while on air with a song they wanted to be included in the upcoming programming (commonly by voice call, historically by handwritten letter, however more recently the contact methods have broadened to include email and text message). While some of these shows would have the entire make-up of a program dictated by listeners, the most common format entails a mix of requested songs from listeners coupled with presenter selections. Ideally, this non-requested material is selected and ordered to construct coherent transitions from one user request to another. Thus the song selection of the DJ (or music director or whomever is ultimately programming the radio show) is being steered by her audience's requests while at the same time acting as a curator of those requests through ordering and selection of other material. (Fields 2011, ch.4) expands our interactive model, based on these request radio shows.

4 The System

SoSoRadio implements the ideas in this paper in a concrete system. This system combines the hybrid similarity space described in (Fields et al. 2011) with the interactivity model detailed in the previous section, creating a complete automatic music-delivery system that responds to users while curating underlying musical data so that users find it compelling. A summary of this system follows while (Fields 2011, ch. 4) presents a complete description.

The front end of our system is designed to reference familiar means of interacting with music collections and social networks, loosely merging a local music library playlist with a social network's 'friend list'. As the content is dynamically generated based on the currently playing song and nominees the particular images will vary, but the layout remains the same.

In this initial landing page the user is presented with three distinct actionable items; the user can: open the stream and listen to the audio content; view the current playlist, see the current track and rate it; and examine the nominees for the current request cycle, including external links for further information, and vote for a nominee.

The core system itself can be considered as comprising three parts: the streaming service itself[2], the playlist gen-

[1]The artist is *Karna Zoo*, Myspace url: `http://www.myspace.com/index.cfm?fuseaction=user.viewProfile&friendID=134901208`

[2]The streaming service is built using the icecast library lib-

erator and the nomination generator. Each of these systems depends on a weighted-directed-graph representation of the songs available to the system. The prototype live instance of the system uses the songs-as-nodes graph described in Section 2, with the weights set as the song-to-song similarity found using Marsyas[4].

The playlist generator finds the shortest path through nodes that have not been visited in the past *session*[5] between the end of the current playlist and the next request track. The playlist generator also performs the vote aggregation to determine the request song.

Finally, the nomination generator determines the songs which will be made available to listeners to request for the upcoming request period. The nomination process first breaks the graph of songs into communities by way of the Fast-Greedy modularization, using the audio weights applied to edges. For each community, excluding the community containing the last song of the current playlist, a representative song is found by the following method.

1. The duration of the potential playlist formed between the last song of the current playlist and a potential nominee must fall within the period range, set a priori[6].
2. The pagerank of each song in the community is then calculated taking the community as a discrete subgraph for the purpose of the pagerank calculation. In this way pagerank serves to describe the inter-community relevance of each track.
3. A song from the 85^{th} to 95^{th} percentile is selected as the representative nominee track for the community. This range is selected to bias nominations toward artists of medium popularity, since recommendations have a tendency to favor the head of the popularity distribution.

At the end of this process each community will have selected a single representative track, with the exception of communities where no tracks can be found that meet the first requirement. From this pool a random subset of 9 is selected and recorded into the database for the interface to access[7].

5 Playlist Analysis and Evaluation

When evaluating playlists it is important to consider what evaluation is for. In the SoSoRadio system, we seek playlists that balance between popular and unknown material, as a stand in for novelty-curve analysis, and playlists should be enjoyable, by way of generating positive ratings. The SoSoRadio system has been running since May 4^{th} 2010 and the following analysis is based on a portion of the playlists that have been created from then until November 2010. Some basic statistics for the playlists being used for evaluation can be seen in Table 2. Myspace artist pages are self-labelled

number of playlists	857
avg. number of songs	6.18
most prevalent genre label	'hip-hop'

Table 2: Basic statistics for the evaluation playlists from SoSoRadio.

genres	≤ 3	$4-6$	$7-9$	≥ 10
playlists	184	326	153	77

Table 3: Number of playlists using a given number of genre labels.

with between zero and three genre labels from a list of options provided by Myspace. These genre labels are used to assess the variance in style seen in the playlists produced by SoSoRadio. Table 3 shows the distribution of the number of genre labels used to describe all artists in a playlist. While the most common number of labels used to describe all the artists in a playlist is three (the maximum number of labels used to describe a single artist), the majority of playlists use at most six genre labels showing a genre coherence.

In addition to considering the coherence of the entire playlist, it is also important to consider the similarity of neighbouring songs in a playlist. While it may be desirable to have some variety in a playlist, this must balance with small changes between neighbouring songs in the majority of cases. This can be measured by finding the change in genre labels from one song's artist to another that neighbours it in playback order, extending the measure used in (Knees et al. 2006) to consider multiple genre labels per song. We call this measure *smoothness*. Given two songs i and j, with genre label sets I and J, the smoothness between them, S_{ij}, is defined as

$$S_{ij} = \frac{|I \cap J|}{\max(|I|, |J|)} \quad (1)$$

$|I|$ and $|J|$ are the number of genre labels used to describe songs i and j respectively. For the Myspace artist pages used by the prototypical instance of SoSoRadio, $|I \cap J|$, $|I|$, and $|J|$ are all between zero and three inclusive.

For example, given song i, associated with an artist labelled with a two genre set ('hip-hop', 'rap'). Each of these genre labels can be considered as 0.5 of the full genre description of the artist. Similarly, song j is associated with an artist labeled with three genre set ('blues', 'funk', 'rap'). Each of these genre labels can be considered a 0.33 of the full genre description of artist. Therefore, by Equation 1, $S_{ij} = 0.33$. Note that this is based on variable genre labels that all carry equal weight, as is the case in Myspace. There are weighted textual descriptions of songs which could be used in a similar fashion (*e.g.* social tags from Last.fm) though richer label sets bring different problems (Fields, Rhodes, and d'Inverno 2010).

Taking the simple mean of the smoothness of all transitions in a playlist provides a measure of how much stylistic change occurs between songs across a playlist. This average across the test set of playlists is seen in Figure 1.

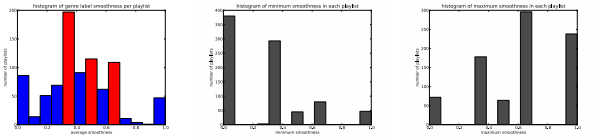
shout[3] via the Python wrapper shout-python.

[4] `http://marsyas.info`

[5] A session is a look back window containing everything that has been played in the last n length of time. In the live prototype, n is 3 hours.

[6] In the live prototype this is 30 ± 5 minutes.

[7] The size of the subset is arbitrary and can be tuned to optimize the interactive model.



(a) Average per playlist smoothness. (b) Minimum per playlist smoothness. (c) Maximum per playlist smoothness.

Figure 1: The histogram of average, minimum, and maximum smoothness for SoSoRadio playlists. The histogram bins containing 0.33, 0.5 and 0.66 in Figure 1a are lightly shaded (red in colour).

The minimum and maximum smoothness for each playlist is also shown in histograms (Figure 1b and Figure 1c). These histograms give the distribution of the least and most abrupt stylistic change in each of session. Taken together, the total genre label counts and mean, minimum, and maximum smoothness show SoSoRadio's output to be stylistically heterogeneous while generally keeping neighbouring songs similar, in so far as the genre labels provide an adequate approximation for stylistic description.

The playlists produced by SoSoRadio are also examined on the basis of artist familiarity. *Pageviews*, or the number of times an artist's profile page has been accessed, are used as a measurement of the familiarity as it is reasonable to assume that an artist whose page has been accessed frequently is more well known than an artist whose page has a low pageview count. The examination of pageviews will follow the course of the previous analysis of genre labels, first looking at the set of songs in each playlist, then examining the transitions.

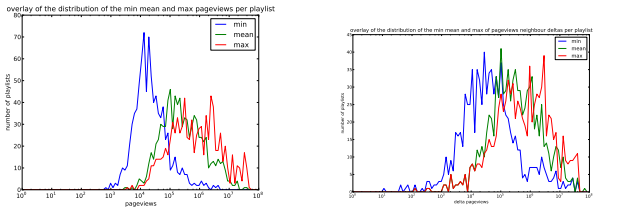
Figure 2a show the distribution of the mean, minimum and maximum artist pageviews, for each playlist in the test set.

Again, an analysis is run on the neighbours, now concerning familiarity. This is investigated by taking the absolute value of the difference between pageviews for each neighbouring song's artist's in a playlist. The mean can then be taken for these values across the entire list. Figure 2b shows the contour of the histogram of this mean, overlaid with similar histograms of the minimum and maximum delta pageviews per playlist.

These analyses show a familiarity that varies in a way that mirrors our underlying dataset's degree distribution (Section 2). That is, playlists are composed mostly of songs from artists with moderate page views (around 100,000) with occasional material from artists with considerably higher page views (two to three orders of magnitude).

6 Summary and Discussion

Given the comprehensive analysis performed on the set of playlists SoSoRadio has produced in prototypical deployment, what have we learned about its output? Our genre smoothness analysis shows the neighboring songs tend to be stylistically overlapping somewhat, with some amount of continuity as measured by the artists's genre labels. At the same time, the analysis of familiarity via artist page



(a) artist pageviews, by playlist (b) magnitude delta artist pageviews, by playlist

Figure 2: An overlay of histograms of the mean, minimum and maximum of each playlist's: artist pageviews, and magnitude delta artist pageviews.

views shows playlists containing artists of varying familiarity (individual artist pageviews vary by 10^5 views in an average playlist, with almost all the system's playlists varying in artist pagesviews by at least 10^3 views). Taken together, this tendency toward smooth genre labels of neighbors and varying familiarity gives the *controlled variance* we designed our system to generate. Given the system's competing constraints of content-based song timbre similarity and community-to-community start and end songs (forced though the nomination process) this is not unexpected.

One of the goals behind the SoSoRadio system is to provide a means for communities of listeners to form. While the prototype has only one station, a means of self-organisation is possible simply by having multiple instances running in parallel. In this case, when a user goes to the site, they would be presented with a selection of stations, seeing a visual representation of what is currently playing on each of these stations with a history. Through this initial process a user places themselves in a community of listeners. If the user's needs aren't being met – they are consistently being out-voted – they can change to a different station, and so to a different community.

References

Fields, B.; Jacobson, K.; Rhodes, C.; Sandler, M.; d'Inverno, M.; and Casey, M. 2011. Analysis and exploitation of musician social networks for recommendation and discovery. *IEEE Transactions on Multimedia* PP(99):1.

Fields, B.; Rhodes, C.; and d'Inverno, M. 2010. Using song social tags and topic models to describe and compare playlists. In *Workshop on Music Recommendation and Discovery, Co-located with ACM RecSys*. Barcelona: ACM.

Fields, B. 2011. *Contextualize Your Listening: The Playlist as Recommendation Engine*. Ph.D. Dissertation, Goldsmiths, University of London.

Jacobson, K., and Sandler, M. 2008. Musically meaningful or just noise, an analysis of on-line artist networks. In *International Symposium on Computer Music Modeling and Retrieval (CMMR)*, 306–314.

Knees, P.; Pohle, T.; Schedl, M.; and Widmer, G. 2006. Combining audio-based similarity with web-based data to accelerate automatic music playlist generation. In *ACM International Workshop on Multimedia Information Retrieval*, 147 – 154.