Using Song Social Tags and Topic Models to Describe and Compare Playlists

Ben Fields, Christophe Rhodes and Mark d'Inverno
Department of Computing
Goldsmiths University of London
New Cross
London, SE14 6NW
United Kingdom
[b.fields | c.rhodes | dinverno]@gold.ac.uk

ABSTRACT

Playlists are a natural delivery method for music recommendation and discovery systems. Recommender systems offering playlists must strive to make them relevant and enjoyable. In this paper we survey many current means of generating and evaluating playlists. We present a means of comparing playlists in a reduced dimensional space through the use of aggregated tag clouds and topic models. To evaluate the fitness of this measure, we perform prototypical retrieval tasks on playlists taken from radio station logs gathered from Radio Paradise and Yes.com, using tags from Last.fm with the result showing better than random performance when using the query playlist's station as ground truth, while failing to do so when using time of day as ground truth. We then discuss possible applications for this measurement technique as well as ways it might be improved.

Categories and Subject Descriptors

H.5.5 [**Sound and Music Computing**]: Signal analysis, synthesis, and processing; H.5.1 [**Multimedia Information Systems**]: Evaluation/methodology

Keywords

LDA, Topic Models, playlists, music, similarity, information retrieval, metric space, social tags

1. INTRODUCTION

Inherent to the design of any recommender or retrieval system is a means of display or delivery of selected content. For a system that recommends music this means playback of an audio file. Listening to or playing a piece of music take the length time of that piece of music. Given this link between music and time, when considering what information is relevant for a recommendation it is vital to consider the context of time; that is, what music has been played before or will be played after the current recommended song. Yet little is understood about how playback order affects the success or failure of a recommendation of a piece of music. Whether a system makes user-based, object-based or hybrid recommendations, a better awareness and use of playback order will yield an improved music recommender system.

In order to take advantage of the effect of playback order, it is necessary to have some means of comparing playlists with one another. While ratings-based generic recommender strategies could be employed, such techniques could only be used in systems which allow for the rating of playlists directly (as opposed to the much more common rating of member songs).Alternatively, n distance measure between playlists can be used to facilitate the prediction and generation of well-ordered lists of song sequences for recommendation. This has the advantage being applicable to the vast majority of existing playlist generation systems, many of which do not to collect playlist level ratings from their users. Further, a measure of playlist distance has a number of other applications in music recommender and discovery systems including label propagation, predictive personalization and context tuning to name a few.

In this paper we propose an objective distance measure between playlists. To better understand why such a measure is needed, Section 2 provides background information in existing playlist generation and evaluation techniques. While any sufficiently expressive and low-dimensional feature is compatible with our playlist measure, we use a novel social tag-based feature in this paper. This song-level feature is detailed in Section 3. This is followed by an explanation of our distance measurement itself in Section 4. Putting this into practice, we detail some proof of concept evaluation in Section 5. We discuss the results of this evaluation and possible extensions in Section 6.

2. PLAYLIST AS DELIVERY MECHANISM

In this section we survey the use of playlists in the delivery of content in existing recommendation and retrieval systems. This is followed by a review of current evaluation methods for generated playlists. These two survey points will show both the widespread use of playlist generation in music recommendation and discovery systems and the need for more quality evaluation of these systems.

While this brief survey is focused on automatic playlist generation, there is a wealth of both academic and lay work discussing various aspects manual human-driven playlist con-

WOMRAD 2010 Workshop on Music Recommendation and Discovery, colocated with ACM RecSys 2010 (Barcelona, SPAIN)

struction that may be of interest to the reader. Work in this area tends to deal with radio (e.g. [1]) or club and dance disc jockeys (e.g. [13]), being the two principal areas where the explicit construction of ordered lists of songs are tied to the field. It is with these areas of manual playlist construction in mind that we will examine past efforts in both automatic playlist construction and evaluation techniques.

2.1 Usage in the Wild

There have been many music recommendation and retrieval systems that employ some kind of automatic playlist construction within their system. Frequently this is done as a means of content delivery or, less often, as a way of facilitating human evaluation of an underlying process such as content-based music similarity or recommendation. What follows is a brief survey of existing methods of playlist generation both with and without human intervention.

A web based system for personalized radio is detailed in [20]. In this early system users create and publish playlists facilitated through a process analogous to collaborative filtering. This results in quasi-automatic playlist creation, with any sequence ordering depending entirely on the user. Another variation of the social interaction intermediary is shown in [27], which presents the *Jukola* system. This system creates playlists via democratic vote on every song using mobile devices of listeners in the same physical space. Furthering the ideas of collaborative human generation, [25] shows a system called *Social Playlist*. This system is based on the idea of social interaction through playlist sharing, integrating mobile devices and communal playback.

A fully automatic rule-based system is described in [2]. This system uses existing metadata such as artist name, song title, duration and beats per minute. The system is designed from the ground up to be scalable and is shown to work given a database of 200000 tracks. An approach that is derived from recommender systems is seen in [4]. Here the authors use the ratings and personalization information to derive radio for a group. An attempt to optimize a playlist based on known user preference as encoded in song selection patterns is shown in [30]. This effort uses Gaussian process regression on user preference to infer playlists. The system uses existing a priori metadata as the features for selection. A means of using webmining derived artist similarity with content-based song similarity is used to automatically generate playlists in [22]. This system combined these two spaces in such a way as to minimize the use of signal analysis. A byproduct of this optimization is improved playlist generation as is shown in a small evaluation with human listeners.

The *Poolcasting* system is detailed in [5, 6]. Poolcasting uses dynamic weighting of user preferences within a group of users who are all listening to a common stream with the goal of minimizing displeasure across the entire group. This results in a system that is very similar to popular commercial radio in terms of its output. A method for created playlists using an artist social graph, weighted with acoustic similarity is shown in [17]. This method takes a start and end song and constructs a playlist using maximum flow analysis on the weighted graph. Another technique for playlist construction based on the selection of paths between the start and end songs is shown in [18]. In this system content-based similarity is used to project a set of songs onto a 2-D map, then a path is found from the start song to the end song with the goal of minimizing the step size between each member song. A recent approach uses co-occurrence in n-grams extracted from the internet radio station Radio Paradise[1] to deform a content-based similarity space [26]. This deformed space is then used in a manner that is similar to [18] to generate paths from one song to another, minimizing step distance throughout the path.

Also of note is [31], which in contrast to most of the previous systems, uses nearest neighbor co-occurrence in radio playlist logs to determine song similarity. While the evaluation was preliminary this method shows promise.

2.2 Evaluation Methods

The most prevalent method of evaluation used in playlist generation systems is direct human evaluation by listening. The system detailed in [29], a rule-based automatic playlist generator that uses features derived from metadata, is similar to [2, 30]. Of note in [29] is the thorough human listener testing which shows the automatic playlist generator performing considerably better than songs ordered randomly. This evaluation, though better than most, still fails to compare the automatic playlists against human expert playlists. Additionally, to reduce test time, the evaluation uses arbitrary one minute clips from the songs rather than the entirety of the song or an intentionally chosen segment. A content-based similarity playlist generator with a novel evaluation is seen in [28]. Here the authors track the number times the user presses the *skip* button to move on from the currently playing song. All songs that are skipped are considered *false positives* and those that are completely played are treated as *true positives*. From this many standard information retrieval techniques can be used in the evaluation, resulting in a rich understanding of the results. Ultimately, it is still human user listening evaluation though and its biggest drawback is playback time. Assuming an average song length of five minutes it would take an an hour and 40 minutes (per listener) to listen to 20 songs with no time for the skipped songs. This skip-based evaluation framework is further used in [12] where existing last.fm user logs (which include skip behavior) are analyzed using fuzzy set theory to determine playlist generation heuristics in the system. Additionally, many systems of playlist generation lack formal evaluation all together.

2.3 Summary

While a number of techniques have been employed to create playlists for a variety of functions, there exist limited techniques in the evaluation of generated playlists. These evaluation techniques rely heavily on time consuming human evaluation. Beyond that, there is no studied means to objectively compare one playlist with another. In Section 4 we will propose just such a means. First we will describe a novel song level feature based on tags. A tag-based feature will encode socio-cultural data that is missing from analogous content-based features, though social tags bring about some other problems.

3. TOPIC-MODELED TAG-CLOUDS

In order to encode playlists in a low dimensional representation we must first represent their member songs in as a low dimensional vector. Here we use a Topic-Modeled Tag Cloud (TMTC) as a pseudo-content-based feature, in a way

[1] `http://radioparadise.com`

Figure 1: The tag cloud for Bohemian Crapsody by Sickboy, from Last.fm.

that is functionally analogous to various pure content-based methods. Using tags and topic models in this way is novel and what follows is an explanation of the process of building this feature.

3.1 Tags as Representation

A *tag* is a word or phrase used to describe a document of some kind, typically on the Web. Various kinds of documents are described using tags on the Web including photos[2], videos[3] and music[4]. An aggregated collection of tags, weighted by the number of users who ascribe it to a given object, is commonly referred to as a *tag cloud*.

Tag clouds get their name from the most common visualization method used with them, where each tag is displayed with the font size in proportion to the weight, arranged in a way that resembles a cloud. An example of a tag cloud[5] can be seen in Figure 1 As can be seen in this example, tag clouds provide a rich description of the music it describes. Tags and collections of tags in various forms provide the basis for many techniques within music informatics including recommendation, retrieval and discovery applications [3,23].

In addition to human generated tags being used, there is some research directed toward the automatic application of tags and inference of associated weights on unlabeled pieces of music [7,9,16,21].

3.2 Reducing the Dimensionality

There exist some techniques (such as [8]) to determine semantic clustering within a tag cloud; however, these systems are built to facilitate browsing and do not create a sufficiently reduced dimensional representation. The previous work of [24] comes the closest to the needed dimensional reduction, also dealing with social tags for music. This work, through the use of aspect models and latent semantic analysis, brings the dimensionality down into the hundreds, while preserving meaning. This order of dimensions is still too high to compute meaningful distance across multi-song playlists. A feature with dimensionally of the order 10^2 would suffer from the curse of dimensionality [33]: because of its high dimensionality, any attempt to measure distance becomes dominated by noise. However, a technique developed for improved modelling in text information retrieval, *topic models* provide the reduced dimensional representation we require. Topic models are described in [10] as "probabilistic models for uncovering the underlying semantic structure of [a] document collection based on a hierarchical Bayesian analysis of the original text." In topic modeling, a *document* is transformed into a *bag of words*, in which all of the words of a document are collected and the frequency of the occurrence in recorded. We can use the weighted collection of tags in a tag cloud as this bag of words, with tags serving as tokenized words.

[2]e.g. `http://flickr.com`

[3]e.g. `http://youtube.com`

[4]e.g. `http://last.fm` or `http://musicbrainz.org`

[5]This tag cloud is for the track Bohemian Crapsody by the artist Sickboy. The tags and the rendering both come from last.fm, available at `http://www.last.fm/music/Sickboy/_/Bohemian+Crapsody/+tags`

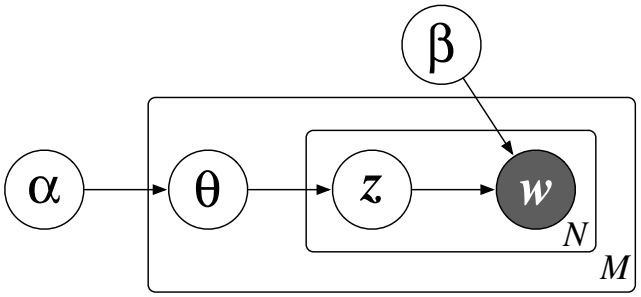

Figure 2: The graphic model of LDA [11]. The replicates are represented as the two boxes. The outer box M represents the corpus of documents, while the inner box N represents the repeating choice of topics and words which make up each document.

There are a few different ways of generating topic models; for our feature generation we will be using latent Dirichlet allocation [11], treating each tag cloud as a bag-of-words. In LDA, documents (in our case tags clouds of songs) are represented as a mixture of implied (or *latent*) topics, where each topic can be described as a distribution of words (or here, tags).More formally give the hyper-parameter α, and the conditional multinomial parameter β, Equation 3.2 gives the joint topic distribution θ, a set of N topics $\mathbf{z}$ and a set of N tags $\mathbf{w}$.

$$p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta)p(w_n|z_n, \beta) \qquad (1)$$

In Figure 2 LDA is shown as a probabilistic graphical model. In order to create topic models using LDA, we need to specify $p(\theta|\alpha)$ and $p(z_n|\theta)$. We estimate our parameters empirically from a given corpus of tag clouds. This estimation is done using *variational EM* as described in [11].This allows topic distributions to be generated in an unsupervised fashion, though the number of topics in a corpus must be specified a priori.

Once the LDA model is generated, it is used to infer the

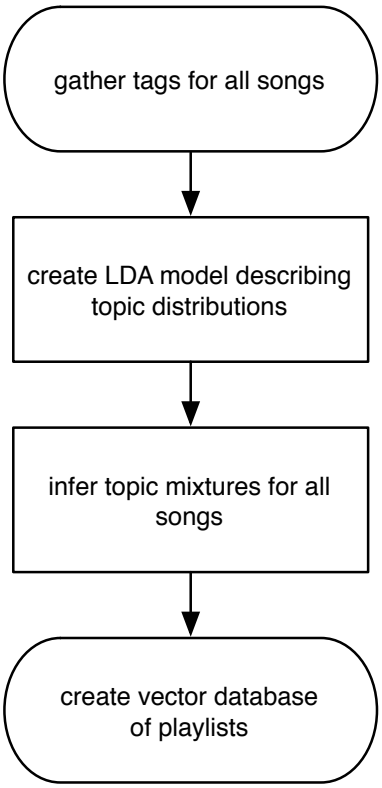

Figure 3: The complete process for construction of a TCTM feature set.

mixture of topics present in the tag cloud for a given song. This is done via *variational inference* which is shown in [11] to estimate the topic mixture of a document by iteratively minimizing the KL divergence from variational distribution of the latent variables and the true posterior $p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta)$.

This process in it's entirety is shown as a block diagram in Figure 3. Once this process is completed for every song in our dataset, we will have a single vector with a dimensionality equal to the number of topics in our LDA whose entries indicate topic occupancy for that song.

4. PLAYLISTS AS A SEQUENCE OF TOPIC WEIGHTS

Given the single vector per song reduction, we represent the playlists these song are in as ordered sequences of these vectors. Thus each playlist is represented as a $l \times d$-dimensional vector, where l is the number of songs in a given playlist and d is the number of topics in our LDA model.

4.1 Measuring Distance

To both manage and measure the distance between these $l_i \times d$ dimensional vectors we use audioDB[6]. The use of audioDB to match vectors of this type is detailed in [32]. Briefly, distance is calculated by means of a multidimensional Euclidian measure. Here l_i is an arbitrary length subsequence of i vectors. In practice, i is Casey:2008selected to be less than or equal to the smallest sequence length for a complete playlist in a dataset. The distance between two playlists is then the minimum distance between any two length i sub-vectors drawn from each playlist. One effect of this technique is easy handling of playlists of unequal length.

This type of distance measurement has been used with success on sequences of audio frames [14, 15]. The distance measure in use between vectors can also be changed. In particular there has been work showing that statistical features (such as topic models) may benefit from the use of Manhattan distance [19], however for our prototypical evaluation we have used simple Euclidean distance as seen in equation **??** above.

5. EVALUATION

The goal of our evaluation is to show the fitness of our distance measurement through preliminary retrieval tests: searching for playlists that start at the same time of day as our query playlist and searching for the playlists from the same station from a database of stations of the same genre. We examine the logs of a large collection of radio stations, exhaustively searching example sets. Through precision and recall we see that our measure organizes playlists in a predictable and expected way.

5.1 Dataset

In order to test these proposed techniques a collection of radio station logs were gathered. These logs come from a collection of broadcast and online stations gathered via Yes.com[7]. The logs cover the songs played by all indexed stations between 19-26 March 2010. For our evaluation task using this data source we looked at subsets of this complete capture, based on genre labels applied to these stations. Specifically we examine stations of the genres *rock* and *jazz*. The complete Yes.com dataset also includes stations in the following genre categories: *Christian*, *Country*, *Electronica*, *Hip-Hop*, *Latin*, *Metal*, *Pop*, *Punk*, *R&B/Soul*, *Smooth Jazz* and *World*. These labels are applied by the stations themselves and the categories are curated by Yes.com. Additionally, the play logs from Radio Paradise[8] from 1 January 2007 to 28 August 2008 form a second set. We then attempted to retrieve tag clouds from Last.fm[9] for all songs in these logs. When tags were not found the song and its associated playlist were removed from our dataset

These logs are then parsed into playlists. For the radio logs retrieved via the Yes api, the top of every hour was used as a segmentation point as a facsimile for the boundary between distinct programs. This is done under the assumption that program are more likely than not to start and finish on the hour in US commercial broadcast. Note that this method of boundary placement will almost certainly oversegment radio programs as many radio programs are longer than one hour. However, given that our distance measure compares fixed length song sequences across playlists, this over-segmentation should produce only minimal distortion in our results. The Radio Paradise logs include all the *links* or breaks between songs where the presenter speaks briefly. For experiments using the Radio Paradise logs these links are used as playlist boundaries. This leads to a slight difference in the type of playlist used from Radio Paradise versus Yes.

[6]source and binary available at `http://omras2.doc.gold.ac.uk/software/audiodb/`

[7]`http://api.yes.com`

[8]`http://www.radioparadise.com/`

[9]`http://last.fm`

source	S_t	S_{mt}	P_t	$P_{avg(time)}$	$P_{avg(songs)}$
whole set	885810	2543	70190	55min	12.62
"Rock" stations	105952	865	9414	53min	11.25
"Jazz" stations	36593	1092	3787	55min	9.66
"Radio Paradise"	195691	2246	45284	16min	4.32

Table 1: Basic statistics for both the radio log datasets. Symbols are as follows: S_t is the total number of song entries found in the dataset; S_{mt} is the total number of songs in S_t where tags could not be found; P_t is total number of playlists; $P_{avg(time)}$ is the average runtime of these playlists and $P_{avg(songs)}$ is the mean number of songs per playlist.

The playlists coming from Radio Paradise represent strings of continuously played songs, with no breaks between the songs in the playlists. The playlists from Yes are approximations of a complete radio program and can therefore contain some material inserted between songs (e.g. presenter link, commercials).

Statistics for our dataset can be see in Table 1 we then use the tags clouds for these songs to estimate LDA topic models as described in Section 3[10]. For all our experiments we specify 10 topic models a priori. The five most relevant tags in each of the topics in models trained on both the rock and jazz stations can be seen Table 2.

5.2 Daily Patterns

Our first evaluation looks at the difference between the time of day a given query playlist starts and the start time for the closest n playlists by our measure. For this evaluation we looked at the 18 month log from Radio Paradise as well as the "Rock" and "jazz" labelled stations from Yes.com, each in turn. Further we used a twelve hour clock to account for The basis for this test relies on the hypothesis that for much commercial radio content in the United States, branding of programs is based on daily repeatable of tone and content for a given time of day. It should therefore be expected that playlists with similar contours would occur at similar times of day across stations competing for similar markets of listeners.

Figure 4 shows the mean across all query playlists of the time difference for each result position for the closest n results, where n is 200 for the Radio Paradise set and 100 for the Yes.com set. The mean time difference across all three sets is basically flat, with an average time difference of just below 11000 or about three hours. Given the maximum difference of 12 hours, this result is entirely the opposite of compelling, with the retrieved results showing no corespondance to time of day. Further investigation is required to determine whether this is a failure of the distance metric or simply an accurate portrail of the radio stations logs. A deeper examination of some of the Yes.com data shows some evidence of the latter case. Many of the playlist queries exactly match (distance of 0) with the entirity of the 200 returned results. Further these exact match playlists are repeated evenly throughout the day. One of these queries is shown in Figure 5. The existance of these repeating playlists throughout the day, ensures this task will not confirm our hypothesis, perhaps due to progaming with no reliance on time of day, at least in the case of Radio Paradise.

[10]Our topic models are created using the open source implementation of LDA found in the gensim python package available at `http://nlp.fi.muni.cz/projekty/gensim/` which in turn is based on Blei's C implementation available at `http://www.cs.princeton.edu/~blei/lda-c/`

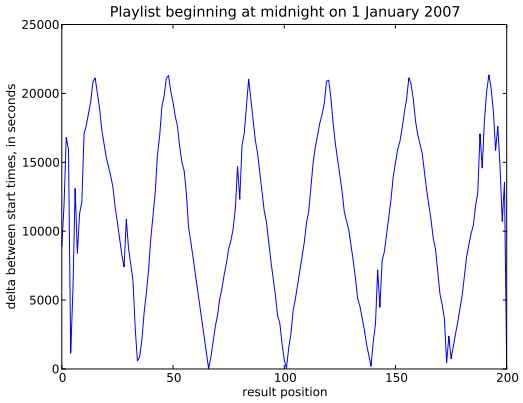

Figure 5: The time of day difference from the query playlist for 200 returned results, showing even time of day spread. Note that all the results show here have a distance of 0 from the query.

5.3 Inter-station vs. Intra-station

In this evaluation we examined the precision and recall of retrieving playlists from the same station as the query playlist. Here we looked at the "Rock" and "Jazz" labelled stations retrieved via the Yes API, each in turn. Similar to the first task, it is expected that a given station will have its own *tone* or particular *feel* that should lead to playlists from that station being more apt to match playlist from their generating station then with other stations from the same genre. More formally, for each query we treat returned playlists as relevant, true positives when they come from the same station as the query playlist and false positives otherwise. Based on this relevance assumption, precision and recall can be calculated using the following standard equations.

$$P = \frac{|\{relevantplaylists\} \bigcap \{retrievedplaylists\}|}{|\{retrievedplaylists\}|} \quad (2)$$

$$R = \frac{|\{relevantplaylists\} \bigcap \{retrievedplaylists\}|}{|\{relevantplaylists\}|} \quad (3)$$

The precision versus recall for a selection of stations' playlists from both the "Rock" and "Jazz" stations are show in Figure 6. When considering the precision and recall performance it

station label	t_1	t_2	t_3	t_4	t_5
"Rock"	Snow Patrol	Bob Marley	female vocalists	aupa Pete	80s
	rumba	Feist	Anna Nalick	whistling	new wave
	90s	john mayer	Chicas	Triple J Hottest 100	david bowie
	green day	drunk love	playlist 2009	review	neuentd
	Dynamit	feist backing vocals	Sarah McLachlan	fun as fuck	synth pop
"Jazz"	motown	john mayer	60s	Sade	Flamenco
	soul	acoustic	jazz - sax	deserves another listen	tactile smooth jazz
	70s	corinne bailey rae	acid jazz	till you come to me	guitar ponder
	funk	bonnie raitt	reggae	piano	cafe mocha
	Disco	David Pack 2	cool jazz	2010	wine

station label	t_6	t_7	t_8	t_9	t_{10}
"Rock"	classic rock	TRB	reminds me of winter	Needtobreathe	Krista Brickbauer
	60s	ElectronicaDance	kings of leon	plvaronaswow2009	day end
	70s	mysterious	songs that save my life	The Script	i bought a toothbrush
	The Beatles	best songs of 2009	songs to travel	brilliant music	bluegrass
	the rolling stones	tribute to george	Muse	van morrison	omg
"Jazz"	follow-up	rnb	female vocalists	classic rock	Smooth Jazz
	jazz	soul	norah jones	80s	saxophone
	instrumental	female vocalists	dido	rock	smooth jazz sax
	guitar	Neo-Soul	jazz	70s	contemporary jazz
	latin jazz	Robin Thicke	vocal jazz	yacht rock	instrumental

Table 2: The five most relevant tags in each topic. Upper model is all the Yes.com Rock stations, lower model is all Yes.com Jazz stations.

is useful to compare against random chance retrieval. There are 100 stations labeled "Rock" and 48 labeled "Jazz". Under chance retrieval a precision of 0.01 would be seen for "Rock" and 0.0208 for "Jazz".

5.4 Summary

Two different evaluation tasks have been run using real world radio log data to examine the usefulness of our playlist match technique. The first of these, an examination the time difference was flat across result length variance. While this implies lack of discrimination into daily patterns, it is not possible to determine from the available data whether this is an accurate reflection of the progamming within the dataset or distance measure not being sufficient for the task. The second task shows the performance of retrieving hourly playlists from a selection of stations using playlists from that station as a query. Here we see a great deal of promise, especially when comparing the query results against random chance, which it outperforms considerably.

6. CONCLUSIONS

Having reviewed recent work in various methods of playlist generation and evaluation in Section 2, it is apparent that there is a need for better ways to objectively compare playlists to one another. We detailed a method of doing so in Section 4, though first, to better filter content-based data through listeners' experience we presented a novel tag-based feature, TMTC, using tags summarized using LDA topic models in Section 3. This was follow by two task evaluations to examine out playlist matching technique and song feature on real world playlist data from radio logs in Section 5.

While our evaluation shows the promise of this technique on sampled data, there is much room for improvement. Principal among these is the exploration of non-Euclidean distance measures. Manhattan distance (or L_1) seems to have the most direct applicability and its use could prove to be quite beneficial. Another area for future work is in the use of the measure on further data and datasets. One of the best ways to improve here would be in the use of datasets with a more exact known ground truth, in order to best apply known recommender and retrieval evaluation methods to them.

This leads to a further avenue of future work, testing the measure against direct human evaluation. While our matching technique has many uses with recommendation and discovery, if it proved to align with human evaluation it would be considerably more useful.

7. ACKNOWLEDGMENTS

This work is supported in part by the Engineering and Physical Sciences Research Council via the Online Music Recognition And Searching II (OMRAS2) project, reference number EP/E02274X/1. Additional support provided as part of the Networked Environments for Music Analysis (NEMA) project, funded by The Andrew W. Mellon Foundation. Thanks also to Paul Lamere for some dataset acquisition assistance.

8. REFERENCES

[1] J. A. Ahlkvist and R. Faulkner. 'Will This Record Work for Us?': Managing Music Formats in Commercial Radio. *Qualitative Sociology*, 25(2):189–215, June 2002.

[2] J.-J. Aucouturier and F. Pachet. Scaling up playlist generation. In *Proc. IEEE International Conference*

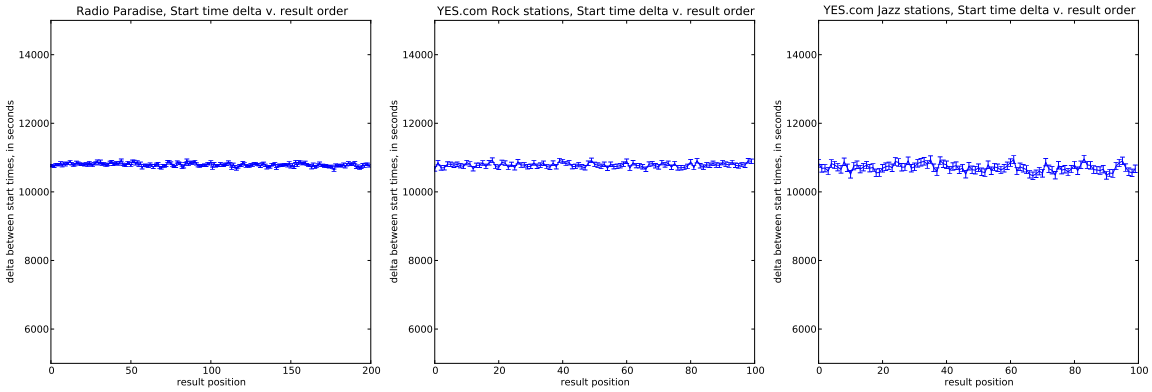

Figure 4: The mean start time difference, with squared error of the mean.

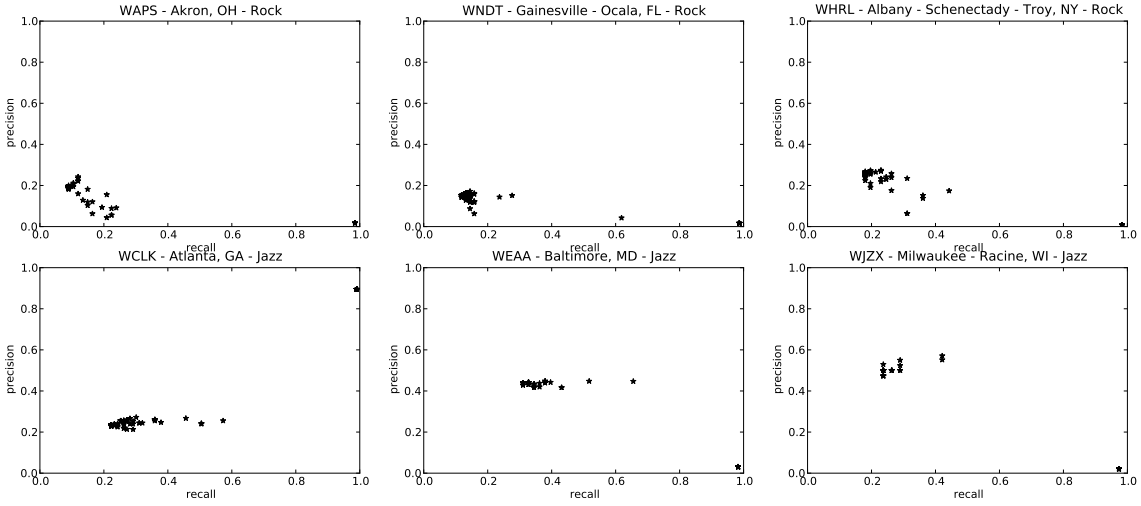

Figure 6: Precision versus Recall for six stations when using their hourly playlists to query for other playlists from the same station. In each query the number of results retrieved is selected to maximize the F1 score.

on Multimedia and Expo, 2002.

[3] J.-J. Aucouturier and E. Pampalk. Introduction-from genres to tags: A little epistemology of music information retrieval research. *Journal of New Music Research*, 37(2):87–92, 2008.

[4] P. Avesani, P. Massa, M. Nori, and A. Susi. Collaborative radio community. In *AH '02: Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 462–465. Springer Berlin / Heideberg, January 2002.

[5] C. Baccigalupo. *Poolcasting: an intelligent technique to customise music programmes for their audience.* PhD thesis, Institut d'Investigació en Intelligència Artificial, 2009.

[6] C. Baccigalupo and E. Plaza. Sharing and combining listening experience: A social approach to web radio. In *Proc. of the International Computer Music Confernce*, 2007.

[7] L. Barrington, D. Turnbull, and G. Lanckriet. Auto-tagging music content with semantic multinomials. In *Proc. of Int. Conference on Music Information Retrieval*, 2008.

[8] G. Begelman, P. Keller, and F. Smadja. Automated Tag Clustering: Improving search and exploration in the tag space. In *In Proc. of the Collaborative Web Tagging Workshop at WWW'06*, 2006.

[9] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: a model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research*, 37(2):101–121, 2008.

[10] D. Blei and J. Lafferty. *Topic Models.* Text Mining: Theory and Applications. Taylor and Francis, 2009.

[11] D. M. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[12] K. Bosteels, E. Pampalk, and E. E. Kerre. Evaluating and analysing dynamic playlist generation heuristics

using radio logs and fuzzy set theory. In *Proc. of Int. Conference on Music Information Retrieval*, October 2009.

[13] B. Brewster and F. Broughton. *Last Night A DJ Saved My Life; The history of the disc jockey*. Headline Book Publishing, London, United Kingdom, 2nd edition, 2006.

[14] M. Casey, C. Rhodes, and M. Slaney. Analysis of minimum distances in high-dimensional musical spaces. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(5):1015 –1028, jul. 2008.

[15] M. Casey and M. Slaney. The importance of sequences in music similarity. In *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing*, Toulouse, France, 2006.

[16] D. Eck, P. Lamere, T. Bertin-Mahieux, and S. Green. Automatic generation of social tags for music recommendation. In *Neural Information Processing Systems Conference (NIPS) 20*, 2007.

[17] B. Fields, K. Jacobson, C. Rhodes, and M. Casey. Social playlists and bottleneck measurements : Exploiting musician social graphs using content-based dissimilarity and pairwise maximum flow values. In *Proc. of Int. Symposium on Music Information Retrieval*, September 2008.

[18] A. Flexer, D. Schnitzer, M. Gasser, and G. Widmer. Playlist generation using start and end songs. In *Proc. of Int. Symposium on Music Information Retrieval*, October 2008.

[19] K. Grauman and T. Darrell. Fast contour matching using approximate earth mover's distance. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

[20] C. Hayes and P. Cunningham. Smart radio: Building music radio on the fly. In *Expert Systems 2000*, pages 2–6. ACM Press, 2000.

[21] M. D. Hoffman, D. M. Blei, and P. R. Cook. Easy as cba: a simple probabilistic model for tagging music. In *Proc. of Int. Conference on Music Information Retrieval*, 2009.

[22] P. Knees, T. Pohle, M. Schedl, and G. Widmer. Combining audio-based similarity with web-based data to accelerate automatic music playlist generation. In *Proc. 8th ACM international workshop on Multimedia information retrieval*, pages 147 – 154, 2006.

[23] P. Lamere. Social tagging and music information retrieval. *Journal of New Music Research*, 37(2), June 2008.

[24] M. Levy and M. Sandler. Learning latent semantic models for music from social tags. *Journal of New Music Research*, 37(2):137 – 150, 2008.

[25] K. Liu and R. A. Reimer. Social playlist: enabling touch points and enriching ongoing relationships through collaborative mobile music listening. In *MobileHCI '08: Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, pages 403–406, New York, NY, USA, 2008. ACM.

[26] F. Maillet, D. Eck, G. Desjardins, and P. Lamere. Steerable playlist generation by learning song similarity from radio station playlists. In *Proc. of Int. Conference on Music Information Retrieval*, October 2009.

[27] K. O'Hara, M. Lipson, M. Jansen, A. Unger, H. Jeffries, and P. Macer. Jukola: democratic music choice in a public space. In *DIS '04: Proceedings of the 5th conference on Designing interactive systems*, pages 145–154, New York, NY, USA, 2004. ACM.

[28] E. Pampalk, T. Pohle, and G. Widmer. Dynamic playlist generation based on skipping behavior. In *Proc. of Int. Symposium on Music Information Retrieval*, 2005.

[29] S. Pauws and B. Eggen. Pats: Realization and user evaluation of an automatic playlist generator. In *Proc. of Int. Conference on Music Information Retrieval*, 2002.

[30] J. C. Platt, C. J. Burges, S. Swenson, C. Weare, and A. Zheng. Learning a gaussian process prior for automatically generating music playlists. In *Proc. Advances in Neural Information Processing Systems*, volume 14, pages 1425–1432, 2002.

[31] R. Ragno, C. Burges, and C. Herley. Inferring similarity between music objects with application to playlist generation. In *Proc. 7th ACM SIGMM international workshop on Multimedia information retrieval*, 2005.

[32] C. Rhodes, T. Crawford, M. Casey, and M. d'Inverno. Investigating music collections at different scales with audiodb. *Journal of New Music Research, to appear*, 2010.

[33] R. Weber, H. J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proc. of the Intl. Conf. on Very Large Databases*, 1998.