MODELING AND PREDICTING SONG ADJACENCIES IN COMMERCIAL ALBUMS

Andy M. Sarroff
Department of Computer Science
Dartmouth College
sarroff@cs.dartmouth.edu

Michael Casey
Department of Music
Dartmouth College
michael.a.casey@dartmouth.edu

ABSTRACT

This paper examines whether latent structure may be discovered from commercially sold albums using features characterizing their song's adjacencies. We build a large-scale dataset from the first 5 songs of 8,505 commercial albums. The dataset spans multiple artists, genres, and decades. We generate a training set ($Train$) consisting of 11,340 True song adjacencies and use it to train a mixture of multivariate gaussians. We also generate two disjoint test sets ($Test_1$ and $Test_2$), each having 11,340 True song adjacencies and 45,360 Artificial song adjacencies. We perform feature subset selection and evaluate on $Test_1$. We test our model on $Test_2$ in a standard retrieval setting. The model achieves a precision of 22.58%, above baseline precision of 20%. We compare this performance against a model trained and tested on a smaller dataset and against a model that uses full-song features. In the former case, precision is better than the large scale experiment (24.80%). In the latter case, the model achieves precision no better than baseline (20.13%). Noting the difficulty of the retrieval task, we speculate that using features which characterize song adjacency may improve Automatic Playlist Generation (APG) systems.

1. INTRODUCTION

Music is often presented in a playlist, which is defined as "a set of songs meant to be listened to as a group, usually with an explicit order" [1]. The song sequence of an album has usually been determined by the artist, engineers, and record label. Similarly, the sequence of songs on a radio show is usually determined by the radio DJ and other human agents of the radio station. Playlists are an important component of music distribution services, and there is an implicit revenue model underlying the song sequence: Content providers maximize profit by retaining listeners' attention; thoughtful playlist design optimizes such retention.

Automatic Playlist Generation (APG) is an algorithmic means for ordering songs. APG systems are an integral part of automatic music recommendation systems; they generate sequences of songs and aim to maximize the listening time of their subscribers. A natural problem for APG systems is selecting an optimal song adjacency from a corpus of songs. Choosing optimal adjacent songs is hard; e.g. when adjacent songs sound too similar, listeners may lose interest quickly. It is important that adjacent songs preserve controlled novelty. There are various methods for addressing the issue—such as thresholding song similarity scores, or providing means for users to train an APG system through user interface feedback. Such methods of song adjacency optimization may be cumbersome, may assume that the user has high level knowledge of his or her preferences, or may overemphasize the importance of music (dis)similarity.

Music albums are meticulously fashioned by their producers to maintain the attention of listeners. An album sequence, which is often defined during the mastering stage of music production, is engineered so that it promotes momentum, continuity, and enjoyment. Albums share a similar revenue model as other forms of music distribution; if an album cannot retain listeners' attention, it is unlikely to be profitable, especially if there exists an option to purchase individual songs. We suggest that the knowledge encoded by transitions of adjacent songs is hidden from ordinary music listeners. However by cleverly modeling song adjacencies, we may learn the structure of this knowledge.

This paper examines whether latent structure can be discovered from commercially sold recorded albums using features characterizing the transitions of adjacent songs. We hand pick a set of features, some of which summarize the adjacencies of song endings and beginnings, rather than whole songs. We fit a mixture of multivariate gaussians to a large dataset and use the model to choose a subset of the hand-picked features in two rounds of feature selection. We test the model on an independent holdout test set in a standard retrieval setting. In a separate evaluation, we test precision on a smaller subset of the dataset. Finally, we evaluate a model that characterizes whole-song adjacencies, rather than adjacent song endings and beginnings. Our results suggest that there is latent structure of song adjacencies in commercial albums.

In the next section (2), we review the background literature on APG systems. Section 3 presents the motivation for the model. Our implementation is described in Section 4 and experiments are presented in Section 5. A discussion follows in Section 6. Concluding remarks are in Section 7.

2. BACKGROUND

The majority of APG systems reported in the literature optimize a cost function based on song similarity. These don't explicitly considering dual-song characteristics. Similarity metrics fall into three categories: metadata, audio content, and collaborative filtering. Modern APG systems incorporate hybrid versions of these categories. Earlier systems did not embrace the content-based retrieval methods that are popular today, but were more likely to consider the importance of a novelty factor. As content-based similarity approaches have taken hold in recent years, there has been decreasing constraint on novelty. Rather, models of individual user preference are often investigated. The following paragraphs briefly summarize some of the work on APG systems in the past decade and a half, but cannot be considered complete. For deeper discussion of the background of APG systems, we refer the interested reader to [2].

Perhaps the first researchers to assert the importance of song sequence for music recommendation systems were Pachet et al. in [3, 4]. Their papers present a system, RecitalComposer, that builds song sequences under a variety of constraints, including song variety. The annotations used to derive song similarity are hand-built, yet the authors suggest that automated methods for annotation may be implemented. Later, Aucouturier and Pachet address issues of scalability [5].

The Personalized Automatic Track Selection (PATS) [6] system generates metadata-based attribute weightings that are used in a clustering strategy. As the system learns user preference, it adapts its attribute weightings to provide more appropriate playlists in the future. Kernel Meta-Training is proposed in [7], which learns a kernel on the metadata for a large set of albums. Gaussian Process Regression is then used to adapt the kernel to user preference. While this method attempts to capture expert knowledge encoded by albums, it is based completely on metadata.

Playlist generation was investigated using graph theory in [8], which discusses a metadata-based use of the model as a possible application. The graph is presented as a network flow problem, and the resulting solution is NP-hard. Graph theory was further investigated by Beth Logan in [9, 10], who uses a simple (MFCC) content-based approach. Logan continues to explore content-based methods in [11]. Pohle et al. further utilize content-based similarity metrics, and test methods for circumventing the NP-hardness of a large graph by investigating several "traveling salesman" algorithms [12]. The same team expands their feature set and introduces new methods of learning users' preference by modeling their skipping behavior [13], a strategy also addressed in [14].

In recent years, authors have increasingly emphasized adaptation to user preference. For instance, [15] examines the listening habits of the user, and [16, 17] assert that users require different playlists at different times of the day. The "derived constraint" strategy described in [18] uses observations of user behavior and forms mutations based on genetic algorithms. Physiological sensor data has also been harnessed in several works [19, 20]. Whereas most systems use a single seed song as a starting point, [21] finds a smooth transition from a start point to an end point. This work is notable because it acknowledges that there may be an "ending" point to a playlist that is dissimilar to the starting point.

Expert knowledge was explicitly considered in [22]. The authors define an Expertly Authored Stream (EAS) as a sequence of songs selected by a professional DJ. They build a Markov random field from the sequential playlists reported by radio stations. The resulting model allows for quick generation of new playlists. However, it excludes any content-based analysis or an in-depth use of song metadata. A similar, case-based approach is presented in [23], which also uses existing playlists as input to the model. Steerable playlists are presented in [24], which uses radio playlists to generate a similarity space for songs. Several binary classification models are trained and tested on binary and ternary song sequences.

3. MOTIVATION

We consider possible oversights in existing APG systems. First, there tends to be an emphasis on the importance of the entire song, rather than the transitions between adjacent songs. As noted by mastering engineer Bob Katz in reference to album sequencing, "the listener actually reacts more to the musical transition than to the entire feel of the previous song," [25]. That is, an optimal sequence for any two songs may be determined by the transition between the first song's ending and the second's beginning, rather than characteristics of the whole songs. Yet many APG systems calculate similarity on a per-song basis by using song metadata or summarizing the song's frame-decomposed features. They usually do not consider explicitly the transitions between adjoining songs.

Radio playlists may capture important expert knowledge about sequence, as well as mix tapes. However, we believe that the produced album might offer an alternative paradigm for modeling song adjacency. With the exception of [7], albums have been largely ignored by APG systems. We suggest that the optimal playlist cannot be determined simply by the similarity of single songs or by user preference; rather, a deeper representation of knowledge should be considered when possible. Finally, no work has proven the scalability of playlist generation to real-world databases of scale.

This work proposes that latent structure may be appropriately modeled by learning from the "expert knowledge" encoded by the song adjacencies of commercially sold albums. In this paper, a mixture of multivariate gaussians is fitted to a large corpus of song transitions. We use the model to determine the posterior probabilities of instances of unseen song adjacencies. We test the model's efficacy in a retrieval task that uses the log likelihood of unseen transition observations. Section 4 presents our dataset and model implementation.

4. IMPLEMENTATION

Our model was trained and evaluated on a subset of the Million Song Dataset. The following sections describe the design of our dataset, its features, and the model.

4.1 Dataset

Columbia University's LabRosa provides the Million Song Dataset (MSD) [26]. The MSD is a comprehensive corpus of songs that were chosen based on criteria such as familiarity, tag prominence, cross-availability with other datasets, and extreme values. The song metadata and analysis features in the dataset are generated by the Echo Nest [1] . There are not many datasets available to the Music Informatics research community that include full or partial albums across many artists, genres, and decades; the MSD thus provides a unique opportunity to model album-based information.

Track numbers and total track counts are not included as metadata in the dataset. However, each song in the corpus includes 7digital [2] IDs for artists, albums, and songs, if they exist. 7digital provided us with the metadata of their American and British catalogues. The combined 7digital catalogues incorporate 316,702 distinct artists; 884,291 distinct albums; and 970,7168 distinct songs. Out of these, we found 341,544 songs that were also in the MSD. From this subset, we searched for all albums with the first 5 contiguous tracks existing in the MSD. This query left us with 44,445 songs across 8,889 albums and 5,076 unique artists.

We further filtered out any artists that had the Musicbrainz [3] tag "classical" and any song sequences having a song with a duration less than or equal to 40 seconds. Song sequences in classical music albums are often determined by different factors (such as broad compositional structure) than popular music genres. We removed albums with short songs because the feature set is meant to characterize song transitions, rather than full songs. As explained below, we arbitrarily assigned a duration of 40 seconds as the length of time for adjacent song transition.

After filtering, the dataset included 8,505 albums of 5 songs each. We note that it is likely that instances of classical or non-musical albums remain in our dataset. We did not perform any finer screening for such occurrences. Henceforth, when we refer to an album we refer to a sequence of the first five songs from an album in our filtered dataset.

The dataset was randomly and equally partitioned into a training set ($Train$) and two test sets ($Test_1$ and $Test_2$), each consisting of 2,835 disjoint albums. Each set was transformed into a new set characterizing song adjacencies. Song adjacencies were labeled "True" or "Artificial". True song adjacencies are those that actually occurred in an album, i.e.

$$True = \{(a,b)|a,b \in \{1,2,3,4,5\} \wedge b = a+1\},$$

where a and b are song indexes to an album. There are therefore 4 True song adjacencies for every 5-song album sequence.

Artificial song adjacencies were generated from permutations of within-album non-sequential songs. Self-song adjacencies were not permitted:

$$Artificial = \{(a,b)|a,b \in \{1,2,3,4,5\} \wedge a \neq b \wedge (a,b) \notin True\}.$$

There were therefore 16 Artificial song adjacencies for every 5-song album sequence. Artificial song adjacencies were not generated across differing albums. Artificial song adjacencies were generated for the two test sets, but not for the training set.

Each set had 11,340 True adjacencies; the test sets each had 45,360 Artificial adjacencies. $Test_1$ and $Test_2$ were further randomly partitioned into N folds each. All experiments except the fourth had $N = 1000$ folds; each fold of each test set of these experiments had approximately 57 song adjacencies. The underlying distribution of True Adjacencies in the test sets was 20%. Song adjacencies were characterized by feature vectors incorporating dual song adjacencies. Features are described in the next subsection.

Feat. Set (f_i)	Feature	# Dim.
1	Fade Dur End/Start	2
2	Timbre Mean End/Start	24
3	Key and Mode	8
4	Timbre Var End/Start	24
5	Loudness Var End/Start	2
6	Loudness Mean End/Start	2
7	Duration	2
8	Tempo	2

Table 1. Features subsets selected from the Million Song Dataset. Feature subsets are ordered in decreasing average precision @ recall=0.01, as shown in Figure 1. The shaded rows indicate feature subsets chosen for final model evaluation.

4.2 Features

We hand-selected 8 feature subsets and performed model-based feature selection on the subsets. Feature selection is described in Section 5. Table 1 shows the feature subsets. Each feature subset includes features from two adjoining songs, a and b. Note that features such as Tempo, Duration, and Key and Mode, are full-song descriptors. Summary statistics for "End" and "Start" features were calculated over a duration of the final 20 seconds of song a and the initial 20 seconds of song b. This duration was selected arbitrarily.

All features were retrieved from The Million Song Dataset and had been calculated by The Echo Nest. The Echo Nest does not provide technical documentation of their feature generation algorithms. Timbre is a 12-dimensional vector described in the Million Song Dataset documentation as "MFCC-like". Tempo and fade in/out durations are estimated by the Echo Nest. Loudness is

[1] http://the.echonest.com/company/
[2] http://about.7digital.net/
[3] http://musicbrainz.org/

expressed in decibels. Timbre and Loudness are each given in the Million Song Dataset at the "segment" level. According to documentation, a segment is a musical event or onset; the algorithm for segment partitioning is not provided by the Echo Nest. We summarized Timbre and Loudness across segments using their mean and variance. Key and Mode, independent features estimated by the Echo Nest, are quantitatively represented using Krumhansl's multidimensional scaling of key and mode [27].

The feature subsets were hand-picked from those available in the Million Song Dataset. We chose to ignore artist-level features such as "familiarity". We also ignored more opaque features such as "danceability" and "hotttnesss"; the research community would have difficulty accurately replicating such features.

We note that there may be better features for modeling song adjacency structure. However, there does not exist a publicly available dataset of commercial music at the scale of the Million Song Dataset. We also acknowledge that we cannot directly assess the integrity of Echo Nest features. We decided to use a large public dataset at the expense of finer control over features.

4.3 Gaussian Mixture Model

For every experiment, a gaussian mixture model was trained on 11,340 True adjacencies using the $Train$ set. We first computed $k = 5$ cluster components over $Train$ using k-means with random sampling and a Euclidean distance metric. For each model, the k-means algorithm was trained 5 times and the index components with the lowest summed within-centroid distances were chosen. The assigned cluster indexes were used as the initial component assignments for a Gaussian Mixture Model with k components and full covariance. The number of components was arbitrarily chosen based on an informal grid search for maximum log likelihood; the search was not exhaustive and was performed only on the $Train$ set.

5. EXPERIMENTS

We performed five experiments. The first two experiments are used for selecting an optimal union of feature subsets. The third experiment uses the optimal feature set and evaluates the model on holdout test data, $Test_2$. In the fourth experiment, we evaluate a model that is trained and tested on a smaller partition of the data. In the last experiment, we train a new model employing only whole-song features.

Let $F = \{f_1, f_2, \ldots, f_8\}$ be the set of feature subsets reported in Table 1. Let S_8 be the set of all permutations of F. In the first four experiments, a model is trained and tested with data having features that are a subset of S_8.

All experiments evaluate, in a retrieval setting, how well the test data fits to a trained model. Given a model, we find the log likelihood (i.e. the probability of the data given the model) of all song adjacencies in each fold of a test set. For each fold, we rank the song adjacencies by their decreasing log likelihood. We then test the precision for each fold at equally spaced recalls ranging from 0.1 to 1. More precisely, let P^s be the output of a model trained and tested on data having features $s \in S_8$. We define $p^s_{i,j}$ to be the precision of the i-th fold at recall level j, where $1 \leq i \leq N$, $0.1 \leq j \leq 1$, and N is the number of folds.

In document query settings, precision is defined as the fraction of retrieved documents that are relevant. Recall is defined as the fraction of relevant documents that are retrieved. In our case, a document is a song adjacency; a relevant document is a True song adjacency. A retrieval query is performed across a fold of a test set. The underlying distribution of True adjacencies is fixed at 20%. However, the actual distribution of True adjacencies varies for each fold. Therefore the expected values of precision and recall when randomly retrieving unranked documents is 0.2.

In order to determine how suitable a feature subset is for our retrieval task, our measure of goodness for the model evaluated on P^s is:

$$\text{score}(P^s) = \frac{1}{N} \sum_{i=1}^{N} p^s_{i,0.1}.$$

In other words, we determine the goodness of a model trained using a feature set s by evaluating its average precision on the test data at a recall level of 0.1.

We note that there is no standard means by which to evaluate the quality of a song adjacency or playlist. Importantly, we have no way of knowing whether an Artificial song adjacency is or is not a valid song adjacency. McFee and Lanckriet have suggested against using an information retrieval approach for evaluating APG systems [28]. Given that we compare model performance across datasets with varying feature dimensionality, the approach that McFee and Lanckriet suggest is more difficult to interpret than ours. Yet there are similarities between our evaluation method and theirs; the evaluation criteria for our models are implicitly based upon the data likelihood given the models.

5.1 Feature Subsets

In the first experiment, we trained and tested models using each of the feature subsets listed in Table 1. The number of folds N was set to 1000. More formally, we compute $Q^1 = \{\text{score}(P^f) \mid f \in F\}$. We report $\langle q_1, q_2, \ldots, q_8 \rangle$, where $q_i \geq q_{i+1}$ and $q_i \in Q^1$. Figure 1 shows the sorted average precision at recall=0.1 for each element in F. The reader may refer to Table 1, which lists the feature subset indexes in the same sorted order as Figure 1. We also report the confidence intervals (at 95% confidence) for Q^1 and the baseline distribution of True adjacencies.

We observe that the best performing feature subset, and the only feature subset giving statistically significant performance above baseline, is 2-dimensional. The fade in and out times achieve average precision of 0.2137. The worst performing feature subset is the two-dimensional feature vector describing the tempos of the first and second song, at 0.1983. We cannot infer that any single feature subset performs better than any other feature subset.

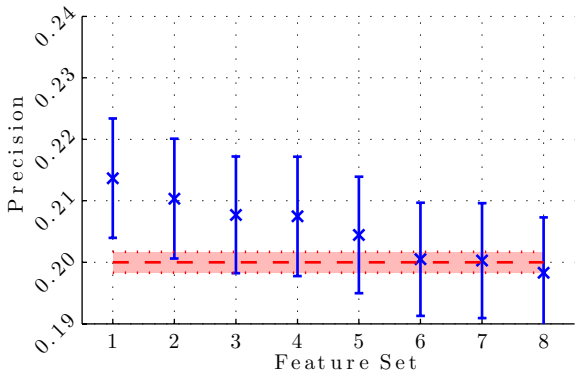

Figure 1. Solid blue: average precision at recall=0.1 for each feature subset shown in Table 1. The models were trained on the $Train$ set and tested on the 1,000 folds of $Test_1$. The error bars show confidence intervals at the 95% confidence level. The red dashed line shows the underlying distribution and confidence interval for True adjacencies.

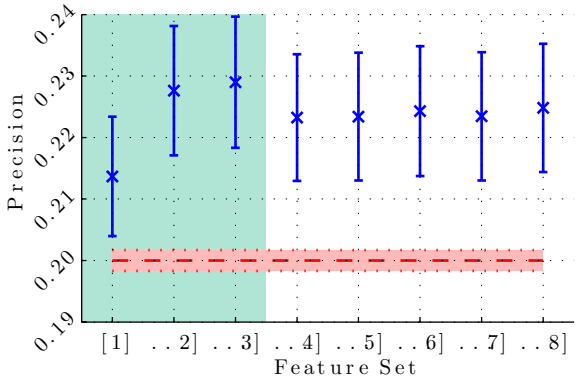

Figure 2. Solid blue: average precision at recall=0.1 for each aggregated feature subset from $[f_1]$ to $[f_1, \ldots, f_8]$ (feature subsets shown in Table 1). The models were trained on the $Train$ set and tested on the 1,000 folds of $Test_1$. The error bars show confidence intervals at the 95% confidence level. The red dashed line shows the underlying distribution and confidence interval for True adjacencies. The green shaded background shows the maximum performing aggregated feature subset (also shaded in Table 1).

5.2 Aggregated Feature Subsets

The second experiment trained and tested the model at aggregated sets of the feature subsets sorted by elements in Q^1. That is, we first tested the best performing feature subset; we added the next best performing feature subset and retrained and retested; we repeated until we tested the model on an aggregated superset of all the feature subsets. More formally, let

$$G = \left\{ \left\{ \bigcup_{i=1}^{j} f_i \right\} \middle| \; 1 \leq j \leq 8, q_j \geq q_{j+1}, q_j \in Q^1, f_i \in F \right\}.$$

We find $Q^2 = \{\text{score}(P^g) \mid g \in G\}$ and report this in Figure 2, which additionally shows confidence intervals and baseline performance. The number of folds was $N = 1000$.

The best aggregated feature subsets are the three subsets highlighted in Table 1: fade durations, averaged timbre over the last 20 seconds of song a and the first 20 seconds of song b; and the key and mode of the two songs represented in 4 dimensions of Multidimensional Scaling. However we cannot infer that any one aggregated feature subset is better than another. The average precision at recall=0.1 of the three subsets was 0.2290. Aggregating additional feature subsets caused a decrease in the mean performance of the model. We infer with 95% confidence that all aggregated feature subsets perform better than baseline.

5.3 Model Performance on Holdout Test Set

We evaluated the best performing set of features from Experiment 2, $f_1 \cup f_2 \cup f_3$, on the holdout test set, $Test_2$. Figure 3 shows that the model's maximum precision is 0.2258, above a baseline of 0.2. The shaded region shows the statistical significance interval (at 95% confidence) for each measured recall. We infer that precision was statistically significant from the baseline at recalls up to and including 70%. The highest precision of this model is less than that found in Experiment 2. This is not surprising, as the experiments were evaluated using disjoint test sets.

5.4 Model Performance on Data Partition

The dataset is quite large and incorporates thousands of genres across many decades of music. For instance, there are 3750 unique Echo Nest "terms" (tags) associated with our dataset. To test the effect of data diversity, we imposed a filter on $Train$ and $Test_2$: We selected all albums whose artists had the Echo Nest tag "Pop" and whose songs had been marked by Echo Nest with a production year in the range $[1990, 2000)$. There were 722 such albums. We trained a model using the same feature set, $f_1 \cup f_2 \cup f_3$, as in the third experiment. To accommodate for the smaller data size, we used $N = 100$ folds in the evaluation. Figure 4 shows average precision. It can be seen that performance improves with a less diverse dataset.

5.5 Model Performance Using Whole Song Features

Finally, we tested the performance of a similar model using only features that describe whole songs, rather than song endings and beginnings. We created a new feature set comprised of Key and Mode (row 2 of Table 1) and Timbre Mean-summarized across whole songs, yielding a 32-dimensional feature vector per transition. The feature vectors were built as described above, such that each feature vector characterized two songs. The performance of this model was 20.13% and had no statistically significant difference from the baseline at any recall level.

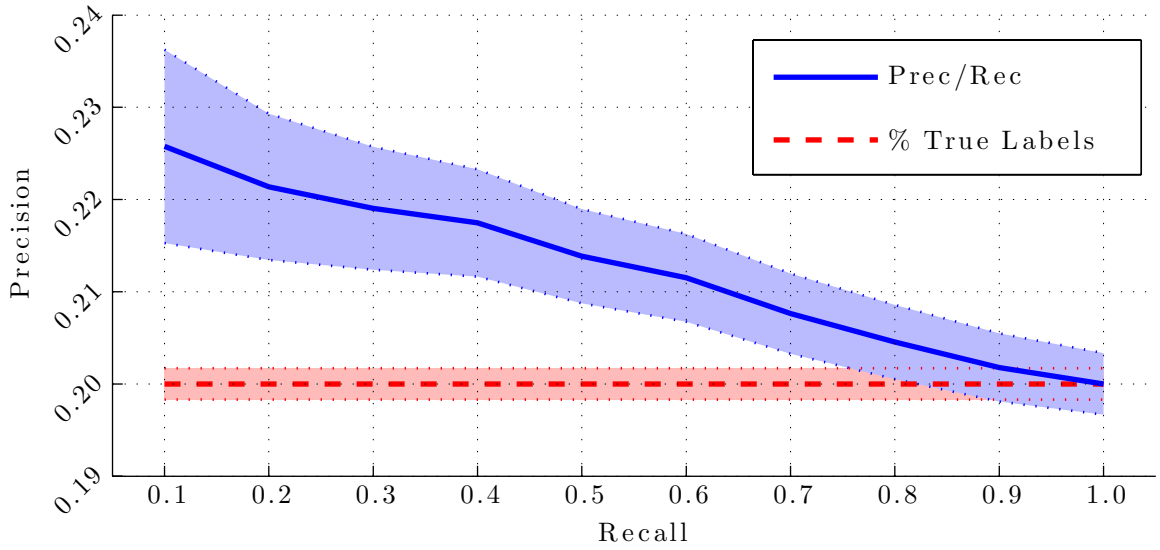

Figure 3. Solid blue: average precision at measured recall levels using the feature subset $f_1 \cup f_2 \cup f_3$. The model was trained on the $Train$ set and tested on the 1,000 folds of $Test_2$. The shaded blue region show confidence intervals at the 95% confidence level. The red dashed line shows the underlying distribution and confidence interval for True adjacencies.

6. DISCUSSION

The reader will observe that the model's final average precision in the third experiment is 2.58% above the baseline. We note that despite the small increase over baseline, the results are statistically significant with 95% confidence. Secondly, we make some observations about the test sets and the difficulty of the retrieval task.

The test sets are comprised of 80% Artificial song adjacencies. Each song adjacency is generated by a song order permutation from songs within our dataset of contiguous albums. It is likely that a high percentage of the Artificial song adjacencies are correct adjacencies. We define a correct song adjacency as one that equally might occur in a commercial album as a True adjacency. We lack means to asses the correctness or incorrectness of Artificial adjacencies. We are therefore unable to identify an upper bound for achievable precision. We can only observe whether the model retrieves True adjacencies with significant precision. We further note that Artificial adjacencies are generated from within albums. Intuitively, this increases the likelihood that a large number of Artificial adjacencies are correct.

We further note that statistically significant precision is higher than baseline up to and including 70% recall. With a simplistic model and a low-dimensional (34 dimensions) feature set we may expect to retrieve up to 70% of True adjacencies at higher chance than random retrieval. We find these results encouraging, given the hardness of the retrieval task.

We attempted to perform an "easier" experiment by training and testing a model using only rock pop songs from the 90s. While the model had higher precision than it did on the larger dataset, we cannot infer with 95% confidence that the performance was any different. This experiment remains quite difficult, considering that Artificial adjacencies are generated from within the same albums.

The successfulness of these models would be easier to interpret if we had a ground truth for invalid song adjacencies. It is difficult to build such a ground truth without extensive subjective testing. Unfortunately such testing was beyond the scope of the work.

We had expected loudness and tempo to play larger roles in retrieving True adjacencies. The loudness of recordings has increased dramatically in the last 20 years. We hypothesize that loudness is less meaningful as a feature to albums produced recently. Perhaps a comparative study on decade-specific subsets of the data would yield different results. It is well-known that it is difficult to estimate tempo correctly. In particular, divide-by-two and multiply-by-two errors are common. We have no means to directly evaluate the accuracy of the Echo Nest's tempo features; it is possible that the dataset has noise.

We find it notable that the best performing feature subset is the fade out duration of song a and the fade in duration of song b. Fade duration is the only isolated feature subset from Table 1 that we can infer statistical significance for discriminative capability. These results suggest that fade times may be relevant to song adjacency structure and hence should be investigated further. The low dimensionality of the feature is further cause for investigation. We also find it notable that a multidimensional scaling of key and mode may have contributed to the success of the model. To our knowledge, song fades, key, and mode have not been investigated in the literature for APG systems. There may be more transition-encoding features that should be explored by the research community.

Finally, we remark that a similar model using whole-song descriptors achieved performance no better than baseline (Experiment 5). This result suggests that endings and beginnings of songs may carry more weight than the middles of songs for album song adjacencies. This knowledge may be useful for building better automatic song sequencers.

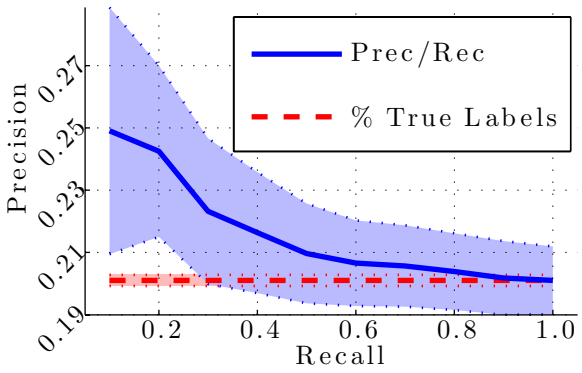

Figure 4. Using a smaller data partition. Solid blue: average precision at measured recall levels using the feature subset $f_1 \cup f_2 \cup f_3$. The shaded blue region show confidence intervals at the 95% confidence level. The red dashed line shows the underlying distribution and confidence interval for True adjacencies. The data partition consists only of artists having an Echo Nest tag of "Pop" and albums with a production date in the range $[1990, 2000)$. The model was trained on the $Train$ set and tested on 100 folds of $Test_2$.

7. CONCLUSION

Automatic Playlist Generation is integral to successful music distribution. While current systems are adaptive to human preference or have sophisticated content-based methods, they do not harness expert knowledge that may be implicitly encoded by the song transitions in produced music albums. This work examined whether latent structure can be discovered from commercially sold albums using features characterizing their songs adjacencies.

We performed two feature subset selection experiments using a gaussian mixture model and large-scale data culled from commercially sold recorded albums. The results of the first experiment show that, of the features being considered, fade durations are the most discriminative. The second experiment shows a potential boost in precision by also incorporating the mean timbre features of song endings and beginnings and the quantified key and mode of adjacent songs.

Our evaluation of the model on holdout test data shows that we can infer significantly higher retrieval precision than random by using these features. In the fourth experiment we find that the model performs slightly better on a smaller dataset of artists having the tag "Pop" and albums having been produced in the 90s. Finally, we find that a similar model that doesn't explicitly take into account song endings and beginnings performs no better than baseline.

In all experiments, the models were not highly sophisticated, the features were hand selected, and the problem space was difficult. We note that this paper does not consider song sequences of length greater than 2. There is perhaps important broader time-scale structure over albums that may be considered in future work. We also remark that we arbitrarily chose to generate the dataset using the first 5 sequential songs of commercial albums. We did not consider that the ordering of the first few songs may be motivated by marketing potential rather than content. It perhaps would have been wiser to choose sequences of songs from the middle of albums. We will explore this and easier datasets in future work.

Nevertheless, we infer statistical significance in the results. Given the hardness of the retrieval task—in particular that we have no way to estimate the percentage of incorrect *and* Artificial song adjacencies in the test set—we find these results to be of interest. We believe that this work encourages deeper investigation into features that characterize song adjacencies and transitions, as opposed to the full-song features commonly used today.

Acknowledgments

The authors wish to thank 7digital. Their data allowed us to assign track numbers for songs cross-indexed with the Million Song Dataset.

8. REFERENCES

[1] B. Fields and P. Lamere, "Finding A Path Through The Jukebox—The Playlist Tutorial," 2010, tutorial, 11th International Society for Music Information Retrieval Conference.

[2] B. Fields, "Contextualize your listening: The playlist as recommendation engine," Ph.D. dissertation, Department of Computing Goldsmiths, University of London, 2011.

[3] F. Pachet, P. Roy, and D. Cazaly, "A combinatorial approach to content-based music selection," in *Multimedia Computing and Systems, 1999. IEEE International Conference on*, vol. 1, Jul. 1999, pp. 457–462.

[4] ——, "A combinatorial approach to content-based music selection," *Multimedia, IEEE*, vol. 7, no. 1, pp. 44–51, 2000.

[5] J.-J. Aucouturier and F. Pachet, "Scaling up music playlist generation," in *Multimedia and Expo, 2002. ICME '02. Proceedings. 2002 IEEE International Conference on*, vol. 1, 2002, pp. 105–108.

[6] S. Pauws and B. Eggen, "PATS: Realization and user evaluation of an automatic playlist generator," in *Proceedings of the 3rd International Conference on Music Information Retrieval*, 2002, pp. 222–230.

[7] J. Platt, C. Burges, S. Swenson, C. Weare, and A. Zheng, "Learning a Gaussian process prior for automatically generating music playlists," *Advances in neural information processing systems*, vol. 2, pp. 1425–1432, 2002.

[8] M. Alghoniemy and A. Tewfik, "A network flow model for playlist generation," in *Multimedia and Expo, 2001. ICME 2001. IEEE International Conference on*, Aug. 2001, pp. 329–332.

[9] B. Logan and A. Salomon, "A content-based music similarity function," Cambridge Research Labs, Tech. Rep., 2001.

[10] B. Logan, "Content-based playlist generation: Exploratory experiments," in *Proceedings of the International Symposium on Music Information Retrieval*, 2002.

[11] ——, "Music recommendation from song sets," in *Proceedings of the 5th International Conference on Music Information Retrieval*, 2004, pp. 425–428.

[12] T. Pohle, E. Pampalk, and G. Widmer, "Generating similarity-based playlists using traveling salesman algorithms," in *Proceedings of the 8th International Conference on Digital Audio Effects (DAFx-05)*, 2005.

[13] E. Pampalk, T. Pohle, and G. Widmer, "Dynamic playlist generation based on skipping behavior," in *Proceedings of the 6th International Conference on Music Information Retrieval*, 2005, pp. 634–637.

[14] D. Gartner, F. Kraft, and T. Schaaf, "An Adaptive Distance Measure for Similarity Based Playlist Generation," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 1, Apr. 2007, pp. 229–232.

[15] A. Andric and G. Haus, "Automatic playlist generation based on tracking user's listening habits," *Multimedia Tools and Applications*, vol. 29, pp. 127–151, 2006, 10.1007/s11042-006-0003-9. [Online]. Available: http://dx.doi.org/10.1007/s11042-006-0003-9

[16] N.-H. Liu and S.-J. Hsieh, "Intelligent Music Playlist Recommendation Based on User Daily Behavior and Music Content," in *Advances in Multimedia Information Processing—PCM 2009*, ser. Lecture Notes in Computer Science, P. Muneesawang, F. Wu, I. Kumazawa, A. Roeksabutr, M. Liao, and X. Tang, Eds. Springer Berlin/Heidelberg, 2009, vol. 5879, pp. 671–683. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-10467-1_59

[17] N.-H. Liu, S.-J. Hsieh, and C.-F. Tsai, "An intelligent music playlist generator based on the time parameter with artificial neural networks," *Expert Systems with Applications*, vol. 37, no. 4, pp. 2815–2825, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/B6V03-4X6MSSX-5/2/85f0089a34613641c67250af2be3717d

[18] J.-L. Hsu and S.-C. Chung, "Constraint-based playlist generation by applying genetic algorithm," in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, oct. 2011, pp. 1417–1422.

[19] D. Cliff, "hpDJ: An Automated DJ with Floorshow Feedback," in *Consuming Music Together*, ser. Computer Supported Cooperative Work, K. O'Hara and B. Brown, Eds. Springer Netherlands, 2006, vol. 35, pp. 241–264. [Online]. Available: http://dx.doi.org/10.1007/1-4020-4097-0_12

[20] N. Oliver and L. Kreger-Stickles, "PAPA: Physiology and purpose-aware automatic playlist generation," in *Proc. 7th Int. Conf. Music Inf. Retrieval*, 2006, pp. 250–253.

[21] A. Flexer, D. Schnitzer, M. Gasser, and G. Widmer, "Playlist generation using start and end songs," in *Proceedings of the 9th International Conference of Music Information Retrieval (ISMIR 2008)*, 2008, pp. 173–178.

[22] R. Ragno, C. J. C. Burges, and C. Herley, "Inferring similarity between music objects with application to playlist generation," in *Proceedings of the 7th ACM SIGMM international workshop on Multimedia information retrieval*, ser. MIR '05. New York, NY, USA: ACM, 2005, pp. 73–80. [Online]. Available: http://doi.acm.org/10.1145/1101826.1101840

[23] C. Baccigalupo and E. Plaza, "Case-Based Sequential Ordering of Songs for Playlist Recommendation," in *Advances in Case-Based Reasoning*, ser. Lecture Notes in Computer Science, T. Roth-Berghofer, M. Göker, and H. Güvenir, Eds. Springer Berlin / Heidelberg, 2006, vol. 4106, pp. 286–300. [Online]. Available: http://dx.doi.org/10.1007/11805816_22

[24] F. Maillet, D. Eck, G. Desjardins, and P. Lamere, "Steerable playlist generation by learning song similarity from radio station playlists," in *Proceedings of the 10th International Conference on Music Information Retrieval*, 2009.

[25] B. Katz, *Mastering Audio: the art and the science*, 2nd ed. Focal Press, 2007.

[26] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[27] C. Krumhansl, *Cognitive foundations of musical pitch*. Oxford University Press, USA, 2001.

[28] B. McFee and G. Lanckriet, "The natural language of playlists," in *Proceedings of the 12th International Conference on Music Information Retrieval*, 2011.