MusicalNetwork: A Novel Visualization for Song Attributes

Adam Kutz, Anastasiia Budko, Fanrui Yan, James Kolonko, Ryan Gust, Yoon Ana Kim

April 2021

1 Introduction and the Problem

A growing complaint today is that the music recommended by popular streaming providers are stale, repetitive, or boring. We attempt to address this by developing a novel interface for users to discover new music from the attributes of the songs themselves, such as tempo and energy. We will allow users to easily visualize these attributes and combine them to discover new music they would not otherwise consider. More formally, we will employ radar graphs as a visualization technique for displaying multidimensional song attributes in a condensed space, allow users to interact with this visualization to create a desirable song profile, and receive recommendations using one of the techniques explored in our survey.

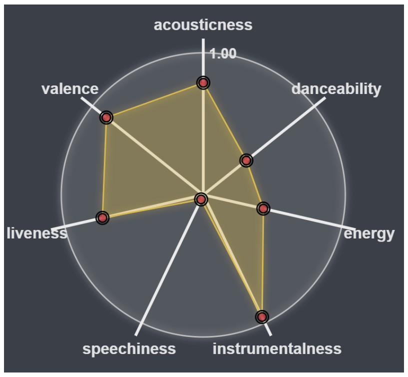


Figure 1: Redar Chart

2 Survey

Most major digital music providers use ”Collaborative Filtering (CF)” to power their music recommendation algorithms, which attempts to group similar users and provide recommendations based on the behavior of that group. Pichl, Zangerle, and Specht [13] delve into a practical implementation of a CF genetic algorithm which uses a ”Jaccard Coefficient” to determine similarity between individuals posting about their music preferences on Twitter. Uitdenbogerd and Schyndel [12] conducted a study of the factors behind the success of music recommenders and found ”...aspects of personality are correlated with music preference.” They tout the benefits of CF based on these findings, and recommend its combination with audio feature analysis to improve performance. An analysis by Datta, Knox, and Bronnenberg [15] found that for users who are averse to spend time searching for music, the CF approach employed by Spotify increased their variety in music taste upon adoption of the service. But they also found that for users who do invest more time into their music discovery process, the benefits were not as clear. Kowald et al [14], tried to address the difficulty of producing quality recommendations for non-mainstream music listeners by attempting to distinguish them from mainstream listeners using KNN and NMF techniques. They use the Spotify Audio Features API, to cluster music and produce high-quality visualizations to present their findings. Chen et al [17] explore the Cold-start problem in classic CF methods (CF needs data to compute similarity, which a new product may not have yet). To address this, the authors train 2 sets of embeddings for Users and Audio features using a modified YoutubeDNN and Siamese

Network architecture, respectively. Snickars et al [3] created Spotify bots to examine the effects of like, dislike, and skip actions, finding regardless of the action, the recommended playlist was essentially the same, repeating the same songs and artists multiple times within 50 songs.

Ethical concerns are raised by the likes of Karakayali [4] who from an anthropological perspective explores the extent to which music taste is driven by the recommendations provided by music services. If this is in fact the case, then alarming concerns are raised with the conjunction of the work of Mulder [2], who found correlation between music taste and certain antisocial behavior. We are not collecting user data, and so we will not consider these factors.

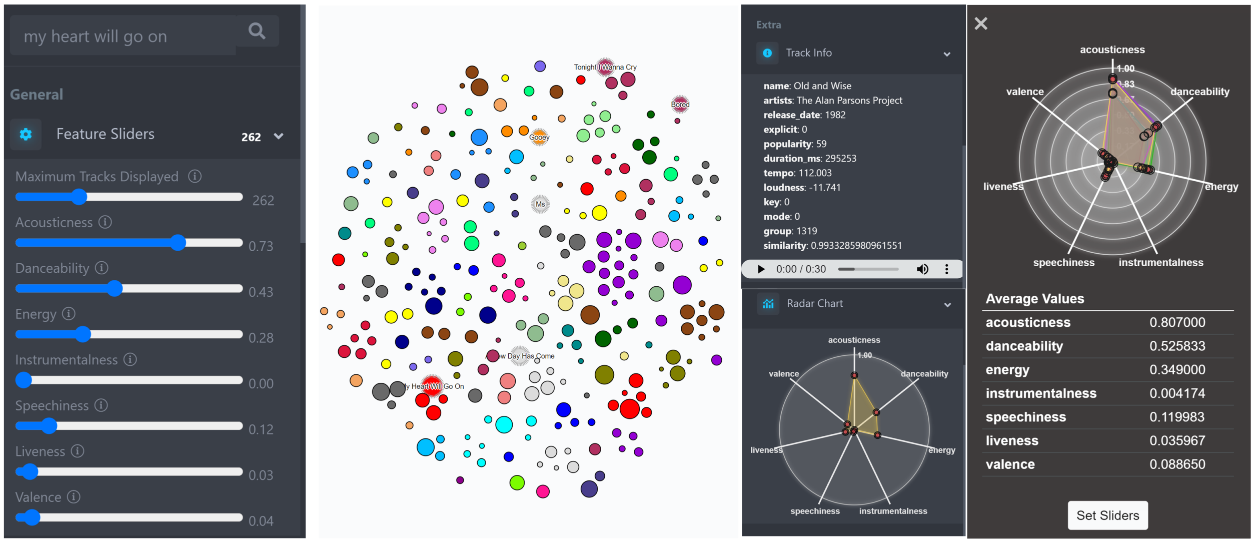
A challenge of this project is in the computation of similarity scores between tracks. Since we are not using user information, we have to rely on similarity between songs. Tsunoo et. al. [6] describes the analysis of percussive patterns within audio tracks to extract certain audio features, using a k-means approach to classify tracks with similar patterns. But, basic k-means may produce clusters with too few points in some datasets, reducing the usefulness of the results. Gangananth et. al. [8] use a modified k-means approach wherein they strategically select the starting centroids of the algorithm, and set a constraint on the minimum size of the cluster to achieve better results. Krey and his team [7] actually employ this form of modified k-means to audio features. Cohn [16] elaborates this even further with the concept of "semi-supervised" learning, where the algorithm will ask the human to label certain points to improve the overall clustering. Yet k-means is not the only available technique, in fact West and Lamere [9] criticize its usage in the music similarity space, opting instead to recommend Linear Discriminant Analysis and Regression Trees to model the genre similarity space. The work of Com and Siminoff [11] and their study of imputing missing data by the help of regression trees can be useful for us on the data preparation step. Jin et al [1], discusses in depth the importance of effective visualizations in a recommending system, highlighting the necessity of targeting an individual's "visual working memory" to achieve "Recommendation Acceptance". Radar Charts as described by Seide et. al. [18], while presented for use in a multi-dimensional network analysis, will be re-purposed as a visual feature in our project to help link music attributes to visual memory. We also found additional practical examples of using these graphs in the works of Porter and Niksiar [5] who walk through several examples with user data from the field of biology. Finally, Tominski et. al. [10] discussed approaches to manage performance for interactive graphs regarding information density.

3 Proposed Method

Current state-of-the-art music exploration/recommendation methods abstract away the low-level audio characteristics from the end-user for ease of use. Our approach aimed not to discard this data, but rather present it in an intuitive way so as to provide users with a more productive search experience. We used two datasets containing 1.2M [19] and 174k [20] songs respectively, sourced from Kaggle and built from the Spotify API with the following music features: acousticness, danceability, energy, duration ms, instrumentalness, valence, popularity, tempo, liveness, loudness, and speechiness. For our submission, we focused our efforts on the smaller of the datasets, which was both cleaner and more information rich. Our initial findings indicated that the noise introduced by larger dataset led to less consistent results and, further, made our means of visualization impractically slow.. Exploratory analysis revealed the impact of missing data was negligible, but we had to drop some duplicate data. We opted to standardize, rather than normalize the audio feature data as this resulted in better performance. We used a python script to compute the 5-nearest neighbors of each track prior to loading the data into the visualization. Then, we applied a label propagation algorithm to partition the dataset into groups which are presented as different colored groups in the visualization. Label Propagation is a semi-supervised learning algorithm that iteratively assigns labels to communities of nodes based on an initial set of labels assigned to a subset of nodes in a graph. Iscan and his team [21] used Label Propagation to create a nearest neighbor graph based on the embedding of that network, Label Propagation has helped them improve performance dramatically. We didn't initially plan to use this technique in our approach until we run into performance issues with performing clustering of large datasets in real-time from a web browser, which is a major challenge for our team.

3.1 Visualization

The visualization is built in D3, and consists of the following elements: A "bubble chart" comprises most of the screen area; here songs are visualized as clusters of circles in a D3 force simulation and are color coded based on the group assigned via the KNN and label propagation process described above. Additionally, the circles are scaled based on the cosine similarity to a set of audio attribute values chosen by the user in the Control Panel's sliders. The Control Panel occupies the left margin of the visualization, and contains sliders which the user can dynamically adjust to rescale the tracks shown on the bubble chart in real time. The Control Panel also displays information about a particular track when the user hovers over it on the Bubble Chart, allows the preview of a 30 second sample of the song (if available), and displays our novel radar chart visualization of the song's audio attributes. The user can select tracks on the bubble chart that are of interest to them. Once a number of songs are selected, the user can click the "merge" button, which will combine the attributes of selected songs. After merging the songs, users can select the "Set Sliders" button, which will set the slider values on the Control Panel to the combined average attribute values of the selected songs. This will update the Bubble Chart to show songs/circles close to the attribute values of the songs selected by the user in the previous step. This process in effect generates song recommendations derived from the tracks users have explored in our chart. This unique approach is what sets our tool apart from most state of the art music recommendations in use today. The efficacy of this approach will be discussed in section 5.



Maximum Tracks Displayed	262
Acousticness	0.73
Danceability	0.43
Energy	0.28
Instrumentalness	0.00
Speechiness	0.12
Liveness	0.03
Valence	0.04

name: Old and Wise
artists: The Alan Parsons Project
release_date: 1982
explicit: 0
popularity: 59
duration_ms: 295253
tempo: 112.003
loudness: -11.741
key: 0
mode: 0
group: 1319
similarity: 0.9933285980961551
0:00 / 0:30

Average Values	
acousticness	0.807000
danceability	0.525833
energy	0.349000
instrumentalness	0.004174
speechiness	0.119983
liveness	0.035967
valence	0.088650

Figure 2: A Novel Visualization for Song Attributes

3.2 Key Innovations

Our innovative ideas for this project are: 1) Slider values dynamically drive the generation circles in real time, allowing the user to easily see which features contribute most to a given song. 2) The D3 radar chart conveys multi-dimensional audio features in a compact and interesting way. 3) 30 second song preview to guide the user's search. 4) Instant large pool of recommended songs by resetting slider values.

4 Experiments

A key technical element of this project was the computation of similarity scores, as our tool aims to identify and visualize songs similar to user specified values. We explored the following concepts:

- K-means: Hard to select a reasonable value for k without prior assumptions about cluster size, values found by silhouette or explained variance offered little intuition and would be difficult to explain/rationalize in visualizations
- HDBSCAN* produced nearly 500 separate groupings which represented only 8.8% of the data, with the remaining 91.2% being classified as outliers. Although these figures could be improved with hyperparameter tuning, it is computationally prohibitive to do so at length
- PCA: uses a simple linear technique to reduce dimension but it fails to capture the data structure. Another drawback is that distances were not guaranteed to be representative of the data
- TSNE/UMAP: computationally expensive, difficult to tune hyperparameters at length. Both are non-linear dimensionality reduction techniques, still not providing any guarantees that distance relationships are preserved through transformation
- Manual Feature Averaging: produces decent results when tuned in isolation, but depends entirely on the choice bounds which vary wildly between samples. Additionally, much of the nuance in the data is lost with this simple aggregate statistic
- KNN clustering: performed the best, since it is the most computationally efficient, gives fairly small error and the most sensible results. The songs sound somewhat similar based on our perception most of the time, but a challenge is that it tends to "recommend" songs that are completely dissimilar in some cases. We attempted the following distance metrics: euclidean, minkowski, cityblock, standardized euclidean, squared euclidean, cosine, correlation, chebyshev, canberra, braycurtis, mahalanobis.

When we attempted to use the concatenation of both datasets to run the program, our web-based application (especially D3's force simulation) struggled to maintain an acceptable level of performance, rendering the application more or less unusable. To address this issue in a more mature application, we may need to explore the use of cloud databases, parallel processing, other optimization techniques, or to work on a standalone application to improve the speed. However, for the purposes of this course, we were able to achieve an acceptable level of usability through the implementation of the label propagation approach described earlier. This offloaded the work of clustering songs to the preprocessing script, resulting in noticeable performance improvements and a better user experience. We also cut down on the maximum number of nodes that can be simultaneously displayed to ease the workload placed on the D3 force simulation. In the future, the tool might benefit from migrating away from a force simulation entirely, though with time constraints imposed by the schedule, exploring reasonable alternatives added too much scope to the project.

5 Evaluation of Results

We divide our attention between three aspects of evaluating the finished product: 1) Technical Performance, in which we discuss the accomplishments and drawbacks of our implementation; 2) User Experience, in which we explore the utility of the tool from a user perspective; 3) Comparison to Existing Platforms, in which we compare the recommended tracks by Spotify to our tool.

5.1 Technical Performance

It proved to be straightforward enough to launch our tool in a web server and examine its performance. While the final product was somewhat of a departure from our original proposal, the implementation in many respects was a success. The slider values approach produces sensible recommendations for most cases; for example, setting high values for Danceability and Energy produces recommendations that seem to make use of synthesizer and heavy bass lines, which is in line with common interpretations of dance music. However, in other circumstances we found that attempting to process a song as a combination of continuous variables can yield odd results; namely in situations where the slider values are relatively far from most songs. In such circumstances, the visualization displays the best matches it can, which often includes a variety of esoteric songs that sound fairly dissimilar to one another. This issue may be assuaged by the inclusion of a larger dataset which, after extensive preprocessing, could populate the sparser regions of the song attribute space. Scalability was a major challenge in this project; as we moved beyond the realm of a toy dataset, we encountered massive performance issues with D3 attempting to search and render tracks without degradation of performance. Given the apparent audience of our tool discussed in the User Experience section, perhaps the tool would better serve such individuals with a standalone application, as opposed to relying on a web browser. Another proposed alternative is to further expand on some of the preprocessing we performed with our use of label propagation, perhaps some form of indexing would yield better search performance.

5.2 User Experience

To evaluate the user experience of our tool, we had a number of friends, family, and coworkers try the tool, and answer a brief survey. The survey aimed to determine A) Whether an individual is a high-volume music consumer/power user. B) Whether they thought our tool was usable, and useful.
The survey respondents were generally supportive and eager to try out our tool, the feedback we received was somewhat mixed. Of the users classed as ordinary music consumers, many felt our tool was non-intuitive and difficult to use, which *was* expected, as our tool does require more active involvement to use when compared to streamlined, popular music services such as Spotify. We may have also achieved better results in this category by providing some form of 'instructions' landing page, or a help menu to better explain to users how to actually *use* the tool. While a smaller number of survey respondents were classed as power users, we did receive significantly more positive feedback from this category of individuals. Such users stated that they thought the tool was usable and had a clean design, with many reporting that they did find new songs they

liked using the tool. From this, we conclude that the tool in its present form would primarily serve a niche category among a subset of music enthusiasts.

5.3 Comparison with Existing Platforms

To attempt to evaluate whether our tool provided suitably distinct recommendations from major music service providers, we compared the top 5 recommendations from Spotify for each song in our dataset to the 5 nearest neighbor recommendations made by our algorithm, which had some surprising results. Roughly 99.45% of tracks recommended by our tool are distinct from Spotify recommendations. When expanding this analysis to 10 recommendations, this number dropped slightly to 98.14%; the overlap of Spotify recommendations and our MusicalNetwork tool is less than 1.86%. Using Youtube as a secondary source, we noticed a 99.97% difference in suggested songs. On one hand, this would suggest our approach is extremely far off base. But this conclusion seems incongruous with our own use of the tool, as well as that of the survey respondents, which suggested that in at least some circumstances our algorithm's results were sensible. One possibility for this discrepancy might be that Spotify is able to reliably leverage a substantially larger dataset than we were able to achieve.

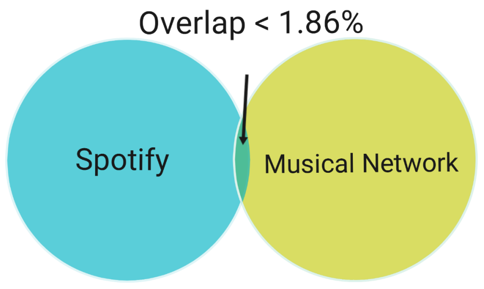


Figure 3: Overlap with Spotify is less than 1.86%

6 Division of Labor

The work is currently broken down as follows: Adam Kutz - Design / Author / Editor, Anastasiia Budko - Author / Editor, Fanrui Yan - Coordinator / Author / Editor, James Kolonko - UI Developer / Video, Ryan Gust - Principal Developer, Yoon Ana Kim - Developer / Editor / Poster Design.

7 Conclusion

All things considered, our efforts have produced a musical exploration tool of nominal interest to music enthusiasts, but of negligible utility to the average music listener. While our tool is unlikely to supplant mainstream music services such as Spotify or Pandora, the exercise of producing this tool helped us develop knowledge that may support future endeavors.

References

[1] Y. Jin, N. Tintarev, N. N. Htun, and K. Verbert, “Effects of personal characteristics in control-oriented user interfaces for music recommender systems,” *User Model. User-adapt. Interact.*, **30**(2), 2020 pp. 199–249, doi: 10.1007/s11257-019-09247-2.

[2] J. Mulder, T. Ter Bogt, Q. Raaijmakers, and W. Vollebergh, “Music taste groups and problem behavior,” *J. Youth Adolesc.*, **36**(3), 2007 pp. 313–324, doi: 10.1007/s10964-006-9090-1.

[3] P. Snickars, “More of the Same – On Spotify Radio,” *Cult. Unbound. J. Curr. Cult. Res.*, **9**(2), 2017 pp. 184–211, doi: 10.3384/cu.2000.1525.1792.

[4] N. Karakayali, B. Kostem, and I. Galip, “Recommendation Systems as Technologies of the Self: Algorithmic Control and the Formation of Music Taste,” *Theory, Cult. Soc.*, 35(2), 2018 pp. 3–24, doi: 10.1177/0263276417722391.

[5] M. M. Porter and P. Niksiar, “Multidimensional mechanics: Performance mapping of natural biological systems using permutated radar charts,” *PLoS One*, **13**(9), 2018 p. e0204309, doi: 10.1371/journal.pone.0204309

[6] E. Tsunoo, G. Tzanetakis, N. Ono, and S. Sagayama, “Audio genre classification using percussive pattern clustering combined with timbral features,” in *Proceedings - 2009 IEEE International Conference on Multimedia and Expo, ICME 2009*, 2009 pp. 382–385, doi: 10.1109/ICME.2009.5202514.

[7] S. Krey, U. Ligges, and F. Leisch, “Music and timbre segmentation by recursive constrained K-means clustering,” *Comput. Stat.*, **29**(1–2), 2014 pp. 37–50, doi: 10.1007/s00180-012-0358-5.

[8] N. Ganganath, C. T. Cheng, and C. K. Tse, “Data clustering with cluster size constraints using a modified k-means algorithm,” in *Proceedings - 2014 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, CyberC 2014*, 2014, pp. 158–161, doi: 10.1109/CyberC.2014.36.

[9] K. West and P. Lamere, “A model-based approach to constructing music similarity functions,” *EURASIP Journal on Advances in Signal Processing*, **2007**(1), 2007 pp. 1–10, doi: 10.1155/2007/24602.

[10] C. Tominski, J. Abello, and H. Schumann, “CGV-An interactive graph visualization system,” *Comput. Graph.*, **33**(6), 2009 pp. 660–678, doi: 10.1016/j.cag.2009.06.002.

[11] Yufeng Ding; Simonoff, Jeffrey S., “An Investigation of Missing Data Methods for Classification Trees Applied to Binary Response Data Yufeng Ding,” *Journal of Machine Learning Research*, **11**(1), 2010 pp 131-170.

[12] R. V. S. Alexandra L. Uitdenbogerd, “A Review of Factors Affecting Music Recommender Success — Semantic Scholar,” *ISMIR 2002, 3rd International Conference on Music Information Retrieval.*

[13] M. Pichl, E. Zangerle, and G. Specht, “Now playing on Spotify: Leveraging spotify information on twitter for artist recommendations,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, **9396**, 2015 pp. 163–174, doi: 10.1007/978-3-319-24800-$4_1$4.

[14] D. Kowald, P. Muellner, E. Zangerle, C. Bauer, M. Schedl, and E. Lex, “Support the Underground: Characteristics of Beyond-Mainstream Music Listeners,” *EPJ Data Sci.*, Feb. 2021, Accessed: Mar. 03, 2021. [Online]. Available: http://arxiv.org/abs/2102.12188.

[15] H. Datta, G. Knox, and B. J. Bronnenberg, “Changing their tune: How consumers’ adoption of online streaming affects music consumption and discovery,” *Mark. Sci.*, **37**(1), 2018 pp. 5–21, doi: 10.1287/mksc.2017.1051.

[16] R. M. A. K. Cohn, David; Caruana, “Semi-Supervised Clustering with User Feedback,” in *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, K. Basu, Sugato; Davidson, Ian; Wagstaff, Ed. Chapman and Hall/CRC, 2008 pp. 17–31.

[17] Chen, Ke; Liang, Beici; Ma, Xiaoshuan; Gu, Minwei. ” Learning Audio Embeddings with User Listening Data for Content-based Music Recommendation,” *Computer Science - Sound; Electrical Engineering and Systems Science - Audio and Speech Processing*

[18] Seide, Svenja E.; Jensen, Katrin; Kieser, Meinhard. ”Utilizing radar graphs in the visualization of simulation and estimation results in network meta-analysis,” *Research synthesis methods.*, **12**(1), 2021 pp 96-105, Great Britain: John Wiley Sons Ltd, 2021.

[19] Figueroa, R. (2021, January). Spotify 1.2M+ Songs, Version 1. Retrieved January 24th, 2021 from https://www.kaggle.com/rodolfofigueroa/spotify-12m-songs

[20] Ay, Y. (2021, January). Spotify Dataset 1921-2020, 160k+ Tracks, Version 10. Retrieved January 24th, 2021 from https://www.kaggle.com/yamaerenay/spotify-dataset-19212020-160k-tracks?select=data$_b y_y ear.csv$

[21] Iscen, Ahmet.; Tolias, Giorgos; Avrithis, Yannis; Chum, Ondrej. “Label Propagation for Deep Semi-Supervised Learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019*, 2019 pp. 5070-5079.