Visualization of Spotify Song Attributes

Algot Sandahl

Abstract—This report covers the implementation of a web application for visualizing song attributes of top songs scraped from Spotify's API. The visualization consists of three main parts: a line chart of average song attributes, a parallel coordinates plot where interesting patterns can be discovered, and lastly a radar chart showing one song at a time. The lines in the parallel coordinate system are colored according to clusters and noise generated by the data mining algorithm DBSCAN. A small user evaluation was conducted to evaluate the application.

Index Terms—Multivariate data, Parallel coordinates, Radar plot.

1 INTRODUCTION

Different kinds of music is a big part of everyday life for many, be it when walking to work, when partying, or as background music when studying. As a way to classify songs, Spotify assigns a set of attributes to all songs on the platform, these attributes can for example describe how acoustic a song is. This report aims to describe an application that visualizes these attributes of popular songs to reveal interesting patterns. The goal is to design it in a way that is accessible to users with no background in information visualization. The application should allow the user to answer questions such as:

- To what extent have acoustic songs been represented in the top-performing songs over time?
- Do popular songs have any common traits?
- What songs stand out each year?

The presented approach relies heavily on parallel coordinates as well as interactivity through brushing and reordering of axes and additional information available on demand. It does also use a line graph showing averages over time, and a radar chart showing one song at a time.

2 BACKGROUND AND RELATED WORK

Peer-reviewed publications on the topic of visualizing song attributes in a way that is similar to the presented one is missing. There are nonetheless peer-reviewed papers on the topic of visualization of multivariate data in a more general sense. An example is [1], which compiles the current state of the art in terms of parallel coordinates. This report takes many of the concepts discussed in [1] and applies them to the context of song attributes. Applications that are more similar to the one presented exist on the internet, one example being [2], which visualize the same kind of data as presented in this paper, although partly based on the author's personal Spotify catalog instead of the most popular songs. What makes the presented application different is the use of parallel coordinates—which enables the visualization of many relationships at a time—and the separate treatment of the years.

3 DATA

There are many interesting aspects of songs that could be analyzed. This application used song attributes found in Spotify's API. The dataset used consists of top-performing songs during the years 2010–2019 and contained 603 items. Each item had the following parameters: title, artist, year, tempo, energy, danceability, liveness, valence, duration, acousticness, speechiness, and a few more parameters that were not used.

• *Algot Sandahl is a student at Linköping University, Sweden, e-mail: algsa119@student.liu.se.*

4 METHOD

To be able to provide insight to the questions presented in the introduction, the application was designed with three panes: one with a line chart with the average values of the attributes as a function of time; one with the songs visualized in a parallel coordinate system, complemented with a list; and finally a pane showing details for one selected song. The flow of the application followed the same order, which was indicated by arrows between the panes. The different parts are described in the following sections.

4.1 Line Chart

The first pane was located in the top left and contained a line graph. For each song attribute, a line with the mean attribute of all songs for each year was drawn. Each song attribute was assigned a color, and the meanings of the colors were shown in a legend. The line chart provided the user with the possibility to see changes over time in a straight-forward way. In addition, it served as a way to filter the information in the parallel coordinate system; a range could be selected to show only songs from that time range in the parallel coordinates plot and its accompanying list.

4.2 Parallel Coordinates

The next pane showed a parallel coordinate system, with each song represented by a polyline. Besides the parallel coordinates, there was a list showing the songs drawn in the plot in text form, as that information is not conveyed in the plot. Which songs were shown could be adjusted by brushing the coordinate axes, which updated both the plot and the list in real-time.

Many of the parameters visualized, for example, acousticness, are inherently restricted to the range 0–100, and the axes reflected this by showing the full range as opposed to the extent of the data. The other axes showed the extent of the data; they did not, however, change when filtering the data, as this could hinder the comparison between different time periods. To highlight a song in the plot, users could hover over it, either directly on the line or on the corresponding entry in the list. To reduce the clutter and make areas with high or medium density distinguishable, the lines were rendered semi-transparent.

To allow for the discovery of relations between all song attributes, the axes could also be rearranged by dragging. An alternative way to provide this functionality would be to provide the user with a few predetermined views, calculated in such a way that all correlations can be found. The dragging approach was chosen over the predetermined views because conveying the meaning of the predetermined views to users in a quick way was difficult.

4.3 Radar Chart

The last visualization presented information about one song at a time as a radar chart for all attributes given between 0 and 100. To the left of the radar plot, information about the duration, tempo, and year of a song was shown in text form. The aim of this visualization was to provide an uncluttered view.

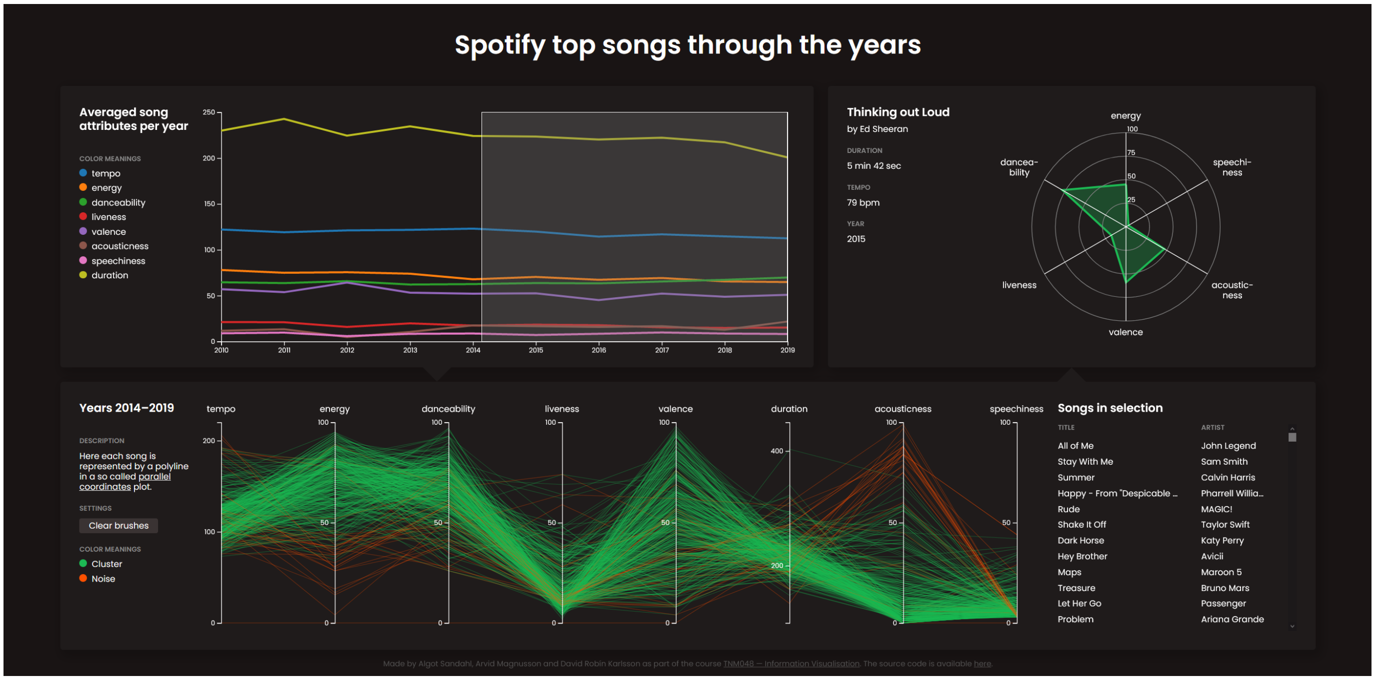

Fig. 1. The full application.

4.4 DBSCAN

To make it easier to identify outliers in the data, the clustering algorithm DBSCAN—density-based spatial clustering of applications with noise, was used to classify the data points as clusters or noise, which was used to color code the polylines in the parallel coordinate system.

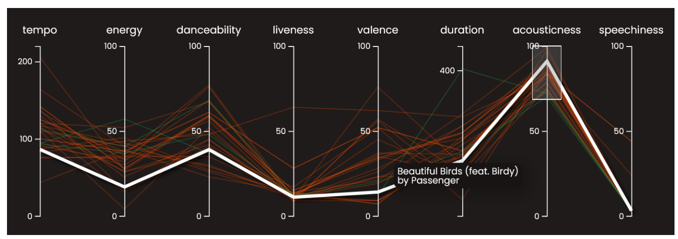

Fig. 2. The parallel coordinate system brushed to only show songs with high acousticness. The image also show a hovered line with accompanying tooltip.

5 Implementation

The implementation was done as a web app to allow it to be run without any installation. It was written in JavaScript using the data visualization library D3.js.

The application targeted modern laptops and desktops; no mobile support was added. The real-time updating of lines while dragging axes or brushing might not run perfectly smooth on low-spec devices.

6 Results

The full application is shown in Fig. 1. The line chart pane is at the top left, the parallel coordinates pane at the bottom, and the radar chart at the top right. Fig. 2 shows the brushing functionality of the parallel coordinates plot, as well as the tooltip shown on hover.

The line graph showing the average values provided a somewhat limited view of changes over time; the main advantage is that it was straightforward to read, as opposed to more information-dense representations.

The use of parallel coordinates enabled good discoverability of patterns in the data. It did not, however, provide an easy way of finding changes in the patterns with respect to time at a first glance, although the time filtering did provide this functionality but in a way that was less immediate and demands more of the user.

The final visualization, the radar plot, provided the user with an uncluttered representation of the song attributes for one song at a time. However, the highlighting of lines in the parallel coordinate system, as shown in Fig. 2, did render this representation somewhat redundant. The textual information shown does however provide more detailed information about the selected song.

The clustering algorithm always identified only one cluster in the data, in spite of the many parameters tried. Although visual inspection also primarily reveals one cluster, many of the acoustic songs could be thought of as a small cluster.

7 Evaluation

To evaluate the application, a small user test with three participants was conducted. The users were presented with four tasks to complete in order. Participants were not allowed to try the application before the tasks. Each task was timed and any unexpected usage was noted. The tasks were:

1. Find the years with the lowest and highest valence.
2. Describe the pattern to which most top songs adhere.
3. Identify songs that stick out, and filter away everything else.
4. Find the song with the most energy during the year 2013, and find out the duration of that song.

The time taken to solve the tasks are shown in Fig. 4.

After completing the tasks, the participants got to rate different aspects of the application on a scale ranging from one to ten, where one means difficult or bad, and ten means easy or good. The answers to these questions are shown in Fig. 3. The missing answers are because the user had not used the radar chart, and thus had no opinion.

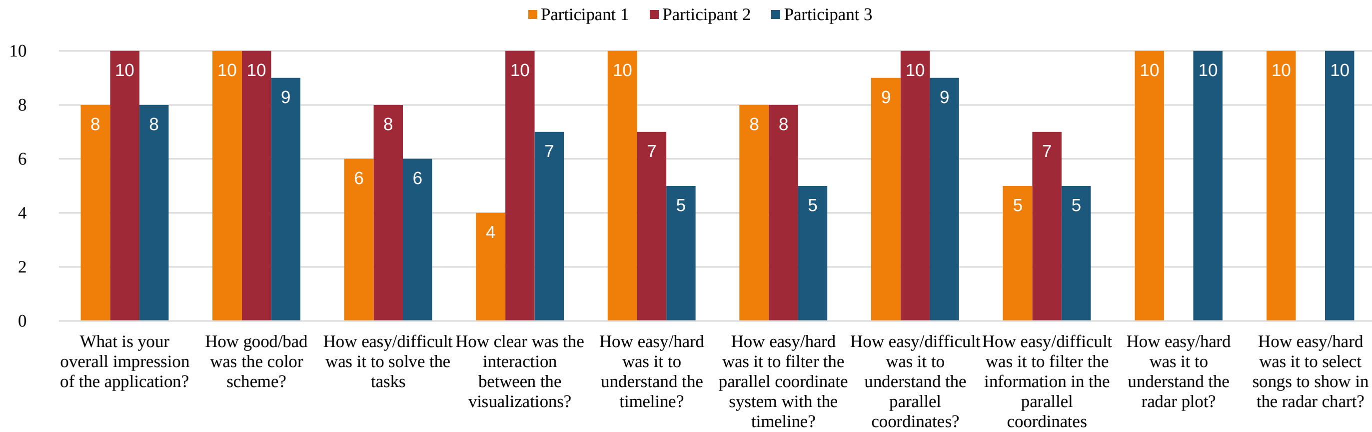

Fig. 3. Evaluation results for questions on a scale between 1, meaning bad/difficult, and 10, meaning good/easy.

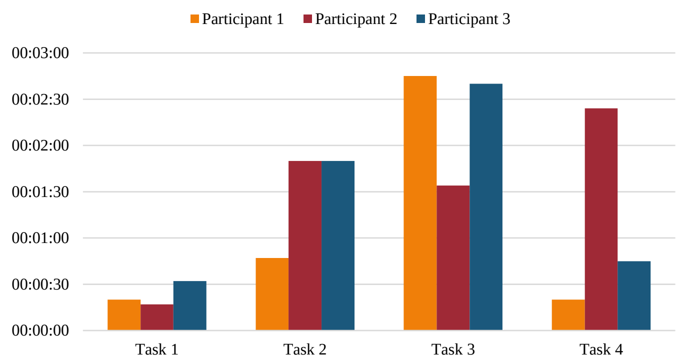

Fig. 4. The time it took each user to solve the tasks.

Finally the participants answered three open questions:

- Was there any interaction that was unclear or hard to understand?
- What is your understanding of the meaning of the red lines, i.e. the noise, in the parallel coordinate system?
- Do you have any other feedback?

One big takeaway from the user evaluation was that the brushing functionality needs to be shown to the user in some way, as it is otherwise hard to discover. This was seen both in the quantitative data as well as in the answers to the open questions.

A majority of the users were confused by the time brushing functionality; they tended to select two years instead of one. The application uses a typical time scale, where each axis tick marks the beginning of a new year. The data points were put at the beginning of each year. The problem was that when selecting a range, the filter selected all songs with a year that was in part covered by the brush—thus selecting songs whose year average was not actually inside the brush. This confused the users and was an oversight in the implementation.

As seen in Fig. 3, all participants understood the parallel coordinates plot well, despite it not being very well known to the public.

Two of the three participants correctly explained that the "noise" in the parallel coordinates plot was songs that did not follow the same pattern as the other songs.

Some suggestions from the evaluation participants were: adding the ability to filter the data based on the cluster/noise data, changing the color of the noise as it could be interpreted as those songs being worse in some way, and adding a help function.

8 Conclusions and Future Work

Based on the data from the user evaluation the interaction with the brushes should be improved. One possible way to make the brushing functionality more apparent would be to always show the brushes—initially ranging the full extent of the axes—with arrow indicators at the brush edges to indicate their interactiveness.

Another important change would be to make sure the time filtering works as expected, i.e. only data points actually inside the brush should be selected.

To allow the user to more easily see if patterns change with time would be to allow for color coding of the lines according to the year of the song. This would likely present challenges with respect to color blending as well as render ordering.

An interesting possibility would be to use a bigger data set, perhaps all songs on Spotify, and find interesting correlations between different parameters, not only for popular songs. Because of the extensive amount of data points, this would be very demanding in terms of computer resources. It would also mean that visual clutter would make the parallel coordinate system presented in this paper unfeasible. Data mining techniques could be used to find clusters, and the centroid or medoid of each cluster could be drawn.

A way to make the radar chart more useful would be to also draw the numerical values for the attributes—this way it would add information not easily found in the parallel coordinates plot.

In conclusion, the presented application can be used to answer questions of the type presented in the introduction. Some aspects of the application have usability issues, but the overall impression of the app remains good.

References

[1] J. Heinrich and D. Weiskopf. State of the Art of Parallel Coordinates. In M. Sbert and L. Szirmay-Kalos, editors, *Eurographics 2013 - State of the Art Reports*. The Eurographics Association, 2013.

[2] D. Spathis. Visualizing hundreds of my favorite songs on spotify. https://medium.com/cuepoint/visualizing-hundreds-of-my-favorite-songs-on-spotify-fe50c94b8af3. Accessed: 2021-03-14.