Evolution of Shape in Lady Gaga's Discography

Selen Berkman, Joey Li, Nathan Luksik

November 21, 2020

1 Problem Description

Lady Gaga is one of the world's best-selling music artists, and is particularly known for her consistent image reinventions [1]. Her 2009 debut album, *The Fame Monster*, yielded two chart-topping singles and established her prominence in the music world. Since then, she has released five other albums, with album number six, Chromatica, released in March of this year. Given her storied career as an artist, one might wonder how her discography has evolved over time, and in particular, whether there are noticeable differences in the "shape" of her music across her different albums.

To explore this question, we purchased three albums – The Fame Monster, Artpop, and Chromatica – spanning Lady Gaga's career. The Fame Monster, as mentioned above, was her first album, and falls into the dance-pop genre of music. ArtPop, her third album, represents a slight divergence from this pattern, being more EDM-influenced. Chromatica, her most recent album, is again of the dance-pop genre. In between these albums, she released Born This Way, which had slightly more of an electronic rock feel, Cheek to Cheek, which went into the jazz genre, and Joanne, which is more readily classified into country pop. Due to monetary constraints on this project, we chose to focus on only three albums, and we chose her first, third, and sixth albums because of their relative similarity in genre as compared to the other three. Additionally, chronologically, they fully span her career's work. Thus, though all three albums are of roughly the same genre, one might wonder whether her excursions into other genres in her fourth and fifth albums ended up influencing the shape of her music in Chromatica, which, ostensibly, is a return to the dance-pop style she started her career in with The Fame Monster.

TDA methods are especially well-suited for discovering shape features in audio time-series data, as has been demonstrated in a large body of previous research, some of which is detailed in 1.1. The principle is that periodicity, which is clearly present in most music, can be detected as a topological feature upon embedding the time series into a higher dimensional space. Finding this periodicity, whether in percussive sounds, pitch-class information, or timbre, can provide a useful way to analyze the shape of a song. For our project, we ask whether these shape features can be used to distinguish songs from different albums by examining different periodic patterns in the audio. In particular, we would like to be able to use shape features to classify songs as belonging to different albums, and furthermore, to interpret this classification as corresponding to differences in certain topological features across albums.

1.1 Related Work

In [2], the authors describe at a high level how to convert musical audio data into a point cloud and then run analyses on it. They consider a window of fixed size which slides over the waveform data and compute features within this window, creating a point in some high-dimensional Euclidean space. Then a tree structure is created on top of the data which summarizes clusters and local PCA is run on identified clusters to estimate the local dimension. This allows for automatic grouping of musical segments such as chorus and verses into strata. In our work, we do not use the full machinery of this cover tree structure, but we do adapt their process of extracting summary features from raw audio data to use in our context.

In [3], the authors prove a result on how to choose optimal values for dimension and τ to detect periodicity in signals from a sliding window embedding. This is of general interest for time-series data, and especially for audio data which inherently ought to have some level of periodicity when considering features such as percussive beats. This work informed our choices when computing our sliding window embeddings.

Our work benefited significantly from the demos of Chris Tralie in [4], which give some starter code for computing sliding window embeddings and audio novelty functions from a raw waveform. Additionally, we made use of the Librosa Python library and its built-in functions to compute Chroma features from raw audio.

2 Methods

At a high level, our pipeline proceeds as follows: we begin by converting sound clips in the form of raw waveforms into audio novelty functions (ANF), and then compute a sliding window embedding on each ANF to obtain a persistence diagram for the 1-dimensional persistent homology of each novelty function corresponding to a sound clip. Following, we give two approaches toward classifying sound clips as belonging to particular albums. In the first approach, we compute pairwise bottleneck distances between corresponding parts (verse, prechorus) of different songs and then compute and display a two-dimensional multidimensional scaling (MDS) embedding of this distance matrix. On top of this, we run a k-means clustering algorithm with $k = 3$ to try to recover the album divisions from this MDS embedding. In our second approach, we convert the persistence diagrams into persistence images, and following, vectorize these persistence images and run some standard classification techniques (logistic regression, support vector machines) on these vectorized images, where the labels correspond to the albums.

The specific details of our procedure are as follows. First, we converted the song files to waveforms so that we could analyze them as time series, and performed a standard sliding window embedding on these raw waveforms. To select our window and step size for any given song, we first used the song's beats per minute (bpm) count to establish a period and then set our window size based on this period. Finally, we computed the 1-dimensional persistent homology on these embeddings to get persistence diagrams for our analysis.

Unfortunately, computing persistence diagrams from the raw audio is unsuccessful, since the sample rate is too high and the signal is messy. Moreover, from a computational stand-

point, the matrices involved in such a calculation are unworkably large. Finally, working over the entire song is not desirable from a theoretical standpoint – sliding window embeddings seek to detect periodic behavior in a signal, and it is much more likely to find such behavior in a smaller segment of a song, where the periodicity is more pronounced.

To clean up the signal from the raw waveform, we turned to Audio Novelty Functions (ANF). An ANF is a version of the original audio signal that correlates with certain percussive events. They detect sudden changes in audio signal by analyzing self-similarity at each instant as well as cross-similarity between the signal before and after each instant. An especially novel point will have a high degree of self-similarity and a low degree of cross-similarity. An ANF can be derived from a spectrogram of a song and picks up on major percussive events in the song, which on the spectrogram are represented by visible vertical lines.

We computed the ANF of each song and performed the sliding window embedding on that signal instead of the raw waveform. The parameters for the sliding window embedding of any given song were chosen to be a constant dimension of 20, with step sizes chosen in accordance with the tempo of the song such that the window size covers exactly two beats. There was no particular reason to expect this to be the optimal set of parameters for the sliding window embedding, but it turned out to yield workable diagrams, which were in particular, much cleaner than the original persistence diagrams computed.

To deal with the multiple issues discussed above associated with computing embeddings over full songs, we manually segmented the songs, marking timestamps where there seemed, heuristically, to be a shift in the music. This could correspond to a change in the musical tone of the piece, or to shifts between lyrical portions, such as from verse to chorus to bridge. Then the embeddings of these audio novelty functions were computed on these segments of songs rather than full songs. This change also greatly improved the quality of our persistence diagrams, as shown in Section 3.

One final step to improve the quality of our persistence diagrams was to reduce noise by performing a density filtration on our point cloud, computing the density around each point in our point cloud and eliminating points in less-dense areas. In our implementation, we planned to do this by taking a ball around a point and growing it until that ball contains a particular amount of other points within it. We would then let the reciprocal of this number represent the density of that point, and then decide upon a threshold and for density, removing points which did not meet the minimum density standard. In the end, filtering by density was not necessary for our persistence diagrams since they already yielded workable features, but we prepared some of the code to do such a filtration, and could employ this technique to further clean up our persistence diagrams in future explorations.

After generating persistence diagrams for each part of each song, we ran two analyses on our data.

First, we categorized our persistence diagrams by the part of the song they corresponded to, such as 'intro,' 'verse,' 'prechorus,' and so on. Then for each type of segment, we considered the space of all diagrams corresponding to this sort of segment in a song and computed pairwise bottleneck distances between these persistence diagrams, creating distance matrices corresponding to each type of segment in a song. This allowed us to compare the persistence diagrams to each other to see which intros or prechoruses were "closest." To visualize this relationship, we applied multidimensional scaling to embed these distances into a plane and

see the relative distances between segments from different songs. To achieve the classification task, we ran a k-means clustering algorithm on this plane embedding to try to recover clusters corresponding to the three albums.

In our second analysis, we generated 20-by-20 persistence images from the already-computed persistence diagrams, and used these images as inputs toward other machine learning methods for multiclass classification with three classes corresponding to the three albums. Qualitatively, looking at the persistence images, they seemed to be similar across albums. We trained both logistic regression and SVM models using a one-versus-rest classification method and tuned hyperparameters using 5-fold cross validation to find an optimal model.

Outside of features derived from the audio novelty function, which correspond to percussive events, we also considered using other summary features from the waveform, such as Chroma, to try to detect pitch-level periodicity in the music. Some preliminary analysis was conducted on sliding window embeddings of Chroma features, but the initial persistence diagrams were not very promising, even when computed on our segmented song clips. Intuitively, this was perhaps not surprising, since it might be difficult for periodicity to arise in the pitch class data of, say, a single verse, and thus the analysis on Chroma features was postponed for a later date.

3 Results

We began by computing persistence diagrams corresponding to audio novelty functions on segments of songs, with sliding window parameters as specified in Section 2. The results are shown in Figure 1 below, which displays persistence diagrams corresponding to segments of the single song *Applause* from the album *Artpop*. Note that in most of these diagrams, there is a prominent nontrivial 1-dimensional homology class.

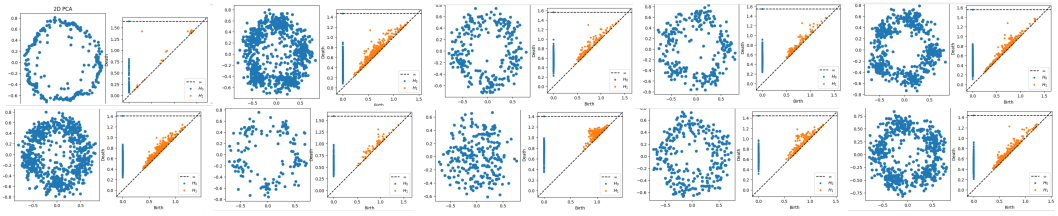

Figure 1: Persistence Diagrams Computed On Segments of *Applause* from *Artpop*

Next, as discussed in Section 2, we categorized the segments of our songs into intro, prechorus, verse, and so on, and computed MDS embeddings from the pairwise bottleneck distances between, say, different intros. Additionally, we computed k-means clustering for $k = 3$ clusters on these MDS embeddings. The results are shown below in Figure 2.

In the plots shown, the colors correspond to the clusters discovered by k-means, while the type of marker represents the actual album which the segment came from. The color of the marker displayed in the legend is not important – it simply arises from the way we plotted the data using matplotlib. As we can see, the clustering on the intros failed because there was one outlier from the Artpop album which was distant from every other song and thus

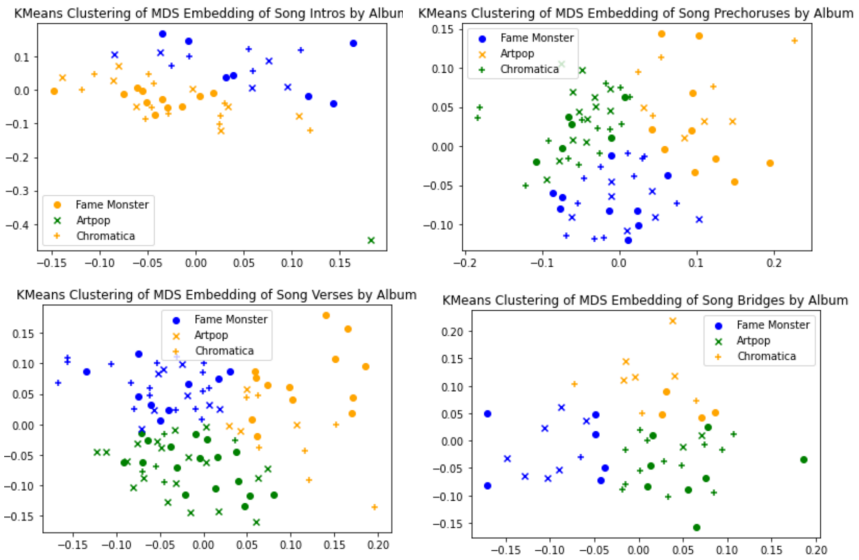

Figure 2: k-means Clusterings on Segments of Songs

consistently was given its own cluster. Similarly, there are no apparent clusters corresponding to albums in the MDS embeddings on prechoruses and verses. In the diagram for bridges, there does seems to be some relative closeness between the points from Chromatica, but the clustering is still relatively ineffective.

Our second procedure was to compute persistence images from the already-computed diagrams and use the vectorized images as input to a multiclass classification problem with three classes corresponding to the three albums. Qualitatively, we can see that the images were not especially distinguishable, as shown in Figure 3, where we display a persistence image computed from a segment of one song from each album.

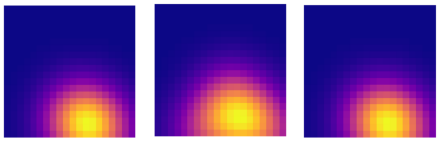

Figure 3: Persistence Images for *Dance in the Dark*, *Jewels N Drugs*, and *Free Woman*

We vectorized the images, split the dataset 60-40 for a train-test split, and used 5-fold cross validation to tune the regularization parameter for a logistic regression model and an SVM model. The logistic regression model consistently overfit, as indicated by extremely high training accuracies in Figure 4. The SVM model did much better, with 10 an optimal value for the regularization parameter C.

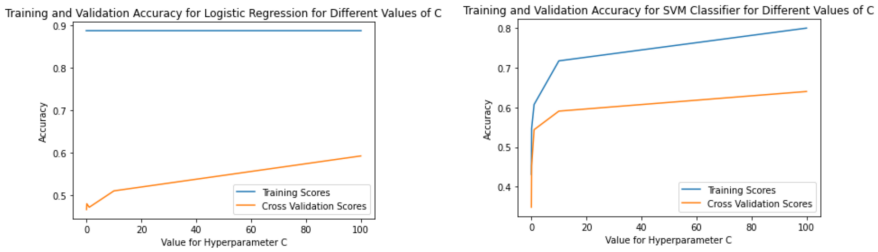

Figure 4: Training and Validation Accuracies for Logistic Regression and SVM

Finally, evaluating this best SVM model on the test set, we achieved an accuracy of 0.62, which is significantly better than simply guessing the label of the most numerous class, which would be 0.43 in this case. At the same time, however, we have a relatively small data set, and the accuracy is still not particularly high.

In conclusion, we have performed two analyses which had middling success in classifying songs into albums by their shape features computed from the audio novelty function. Given that the audio novelty function captures percussive events, this perhaps signifies that Lady Gaga's music has not made any large-scale shifts in its percussive instrumentation. Though there may certainly be songs that had particularly different shape, such as the one outlier among the intros as pictured in Figure 2, there may not have been wholesale shifts. In order to detect changes in shape, then, we may have to focus on other aspects of her music, such as pitch-class information or timbre.

The latest version of the code is maintained at https://github.com/Octophi/TDA-Project.

4 Future Directions

In a future iteration of this project, we would seek to explore the use of other features such as Chroma, MFCC, or timbre, to try to analyze other aspects of the evolution of Lady Gaga's music. As noted above, perhaps the larger shifts in Lady Gaga's music are not percussive in nature, but correspond to other changes.

Additionally, it would be helpful to analyze her other three albums, which are of different genres than the three we have worked with here. Presumably, the differences between these albums would be more prominent, and thus trying to carry out a similar classification task would be useful in checking our methodology.

Finally, we would consider computing persistence landscapes from our persistence diagrams so that we can compute means and variances in persistence landscapes across the albums. This would allow us to make other judgments on her music, such as determining which album is the most diverse and which album has songs most similar to each other.

References

[1] Wikipedia contributors. Lady gaga — Wikipedia, the free encyclopedia, 2020. [Online; accessed 15-October-2020]. 1

[2] Paul Bendich, Ellen Gasparovic, John Harer, and Christopher Tralie. Geometric models for musical audio data. In *32nd International Symposium on Computational Geometry (SoCG 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016. 2

[3] Jose A Perea and John Harer. Sliding windows and persistence: An application of topological methods to signal analysis. *Foundations of Computational Mathematics*, 15(3):799–838, 2015. 2

[4] Chris Tralie. Tdalabs. https://github.com/ctralie/TDALabs, 2020. 2