CS 6965 - Project 1

Maks Cegielski-Johnson

January 30, 2018

Q1

What is the effect of increasing interval overlap parameter on the final graph in the visualization?

Figure 1: 20% Overlap

Figure 2: 80% Overlap

Firstly, as we can see from the two figures, there are fewer clusters when the overlap is set to 80% than there are with 20% overlap. Secondly, upon inspecting the clusters, it becomes apparent that the clusters in Figure 3 have larger membership than those in Figure 1. This is not surprising due to the pigeonhole principle. Having a larger overlap means more points are members of multiple clusters.

Q2

What is the effect of increasing number of interval parameter on the final graph in the visualization?

Figure 3: 80% Overlap and 15 Intervals

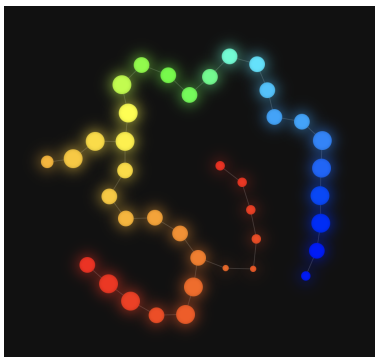


Figure 4: 80% Overlap and 30 Intervals

As seen in the figures, increasing the number of intervals from 15 to 30 increases the number of clusters from 18 to 39 clusters, as well as changing the color scale from $[lightgreen, red]$ to $[blue, red]$, indicating the addition of more clusters. There is also a larger amount of *small* clusters, with membership less than 100. This is visually indicated my the larger amount of circles with small radius.

Q3

Why does the resulting graph contain two connected component?

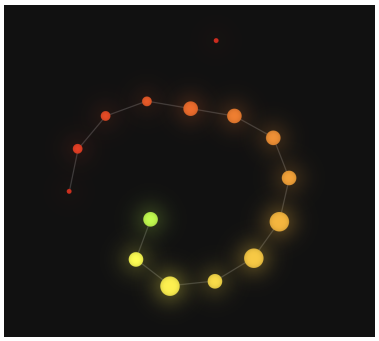


Figure 5: Mapper visualization of bunny

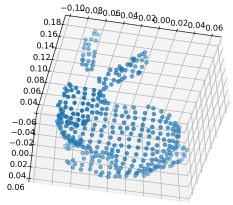


Figure 6: Original bunny point cloud

If we look at the original bunny point cloud Figure 9, then we see that one of the ears is relatively far from the rest of the bunny. Since these points are far *enough* away from the rest of the points, they end up in a disconnected cluster.

Q4

What does this modification do to my data? (Hint, what lens am I using for my data after the modification?)

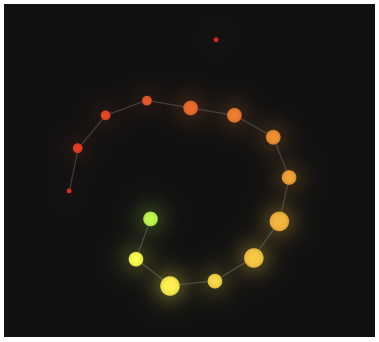


Figure 7: Mapper visualization of bunny

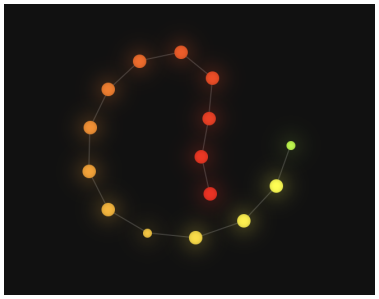


Figure 8: Mapper visualization with lens

Changing the projection to [1] projects our point cloud data onto an axis-aligned line for the second dimension (dimension with index 1), in this case mapping the data onto the y-axis. In other words, takes only the data in the y-dimension of our original point cloud (the height function in y dimension)

Q5

Does this modification help better capture the shape of the data? If yes, how?

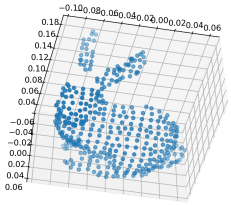


Figure 9: Original point cloud of bunny

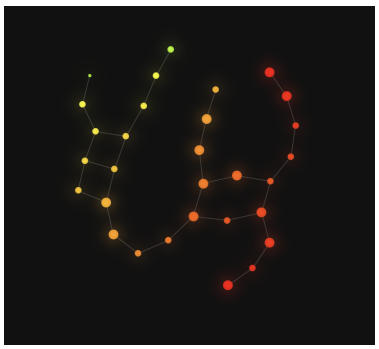


Figure 10: Mapper visualization with K-Means

Yes it does. There appears to be a clear section for the head, ears, body, tails, and legs.

Q6

Why are the results not necessarily identical?

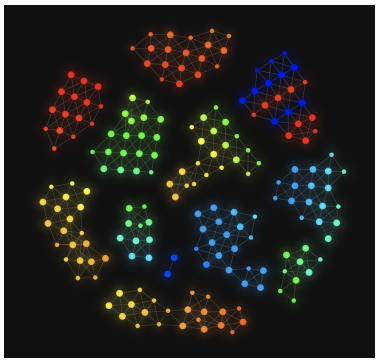


Figure 11: First Run

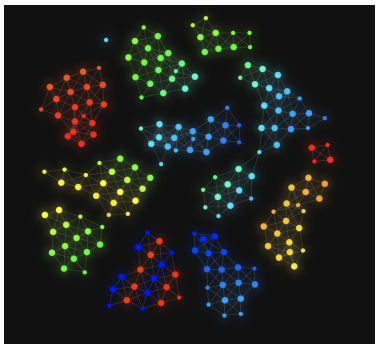


Figure 12: Second Run

The results are not identical because there is randomization during the initialization of the gradient descent.

begin
  compute pairwise affinities $p_{j|i}$ with perplexity *Perp* (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  for t=1 **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

Figure 13: t-SNE Pseudocode

This can be seen in Figure 13 on the line which randomly samples $\mathcal{Y}^{(0)}$ from $\mathcal{N}(0, 10^{-4}I)$.

Q7

What is the difference between the resulting using Spectral Embedding in comparison to the results using t-SNE?

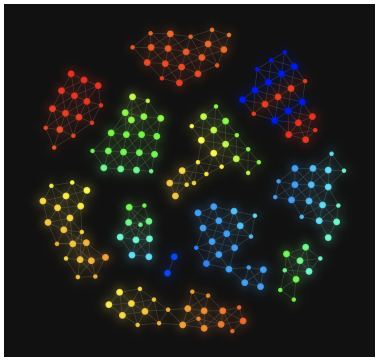


Figure 14: Mapping with t-SNE

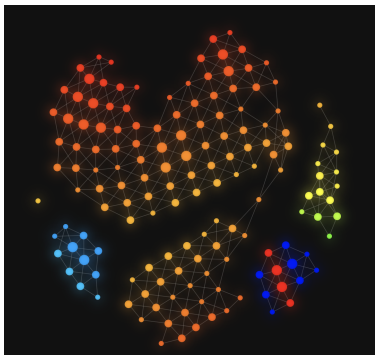


Figure 15: Mapping with Spectral Embedding

There is a smaller number of connected components in the Spectral Embedding graph than in the t-SNE graph, with most of the connected components being the same size as in t-SNE except for one which is significantly larger.

Q8

What is your modification and its effect on the data?

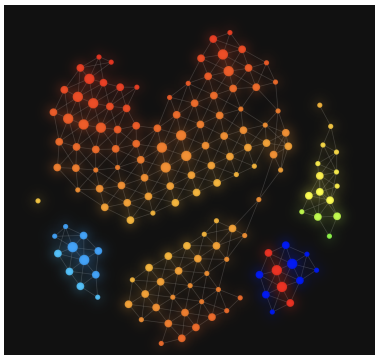


Figure 16: Spectral Embedding, dimension 2

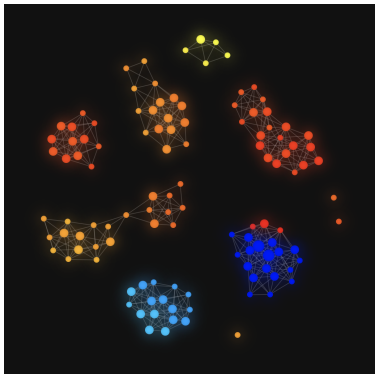


Figure 17: Spectral Embedding, dimension 3

Since we were asked the only change the parameters of the spectral embedding, I left the mapper parameters the same. I increased the dimension of the projected subspace from 2 to 3, with the resulting graph in Figure 17. The result was that the large connected component from the original spectral embedding with dimension 2 being split up to a set of few clusters, splitting up the numbers much better. There was still a little bit of noise in some of the groups, and two groups of different digits were connected by one noisy cluster (the yellow/orange group in the bottom left, above the cyan group in Figure 17).

I also tried setting the subspace to 4 dimensions, but this didn't have any interesting results.

Q9

Custom dataset

For my dataset I reused my Spotify Chart dataset which I used for the CS 6630 (data visualization) final project[1] last semester. The data represents all of the songs that were in the Spotify Weekly Global Top 200 throughout 2017, and some relevant information about these songs. One specific set of data associated with each song is it's track features, which is a set of extracted values describing different aspects of the track, all computed and provided by Spotify.

- **Acousticness**

  A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.

- **Danceability**

  Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.

- **Energy**

  Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.

- **Instrumentalness**

  Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.

- **Liveness**

  Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.

- **Speechiness**

  Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.

- **Valence**

  A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

- **Loudness**

  Average Decibel measure of the song.

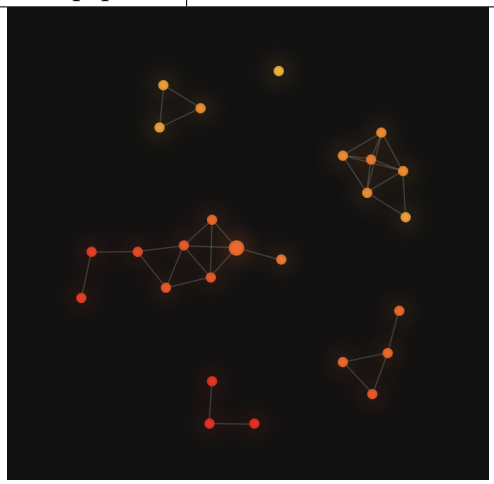
I then use these features and over 1000 songs in conjunction with t-SNE and the keppler mapper to visualize the data. Although not in the scope of this assignment, the resulting graph could be incorporated with my visualization from CS 6630 to allow for better user interaction to see if any patterns can be discovered.

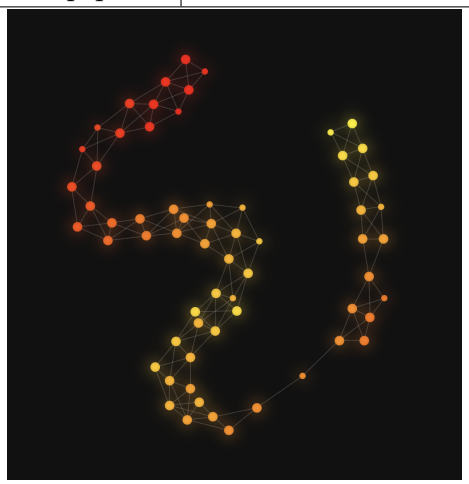
[1] `https://makscj.github.io/visualization-project/`

Since the point-cloud is more or less artificially induced through other machine learning tools, the expected outcome of the keppler-mapper visualization is unknown. Playing with the parameters results in different visualizations, all of which are interesting and seem to provide some insight to the data.

Below are several of the results displayed for different parameters, with a referenece to the particular HTML file which can be viewed in more detail.

DBSCAN		DBSCAN	
music_vis_1.html		*music_vis_2.html*	
n_components	3	n_components	2
perplexity	40	perplexity	30
eps	0.4	eps	0.6
min_samples	15	min_samples	15
nr_cubes	15	nr_cubes	15
overlap_perc	0.5	overlap_perc	0.5





DBSCAN		K-MEANS	
music_vis_3.html		*music_vis_4.html*	
n_components	2	n_components	2
perplexity	40	perplexity	30
eps	0.3	n_clusters	2
min_samples	10		
nr_cubes	20	nr_cubes	20
overlap_perc	0.2	overlap_perc	0.2