The Six degrees of ‘The Six Degrees by Kevin Bacon’ by Sohaib and Dylan

An Inspection of YouTube’s suggestion algorithm.

# Quick Recap

This project was inspired by the idea of six degrees of separation. The initial idea was to determine if you could get from one video to the Ted Talk, “The six degrees | Kevin Bacon”, on YouTube within six hops, and how YouTubes suggestion algorithm influences this. While a fun idea, it was essentially meaningless. This led us to sit back and think of how YouTubes suggestion algorithm impacts its users in a meaningful way. The question then arose; Does YouTubes suggestion algorithm have a bias towards certain genres?

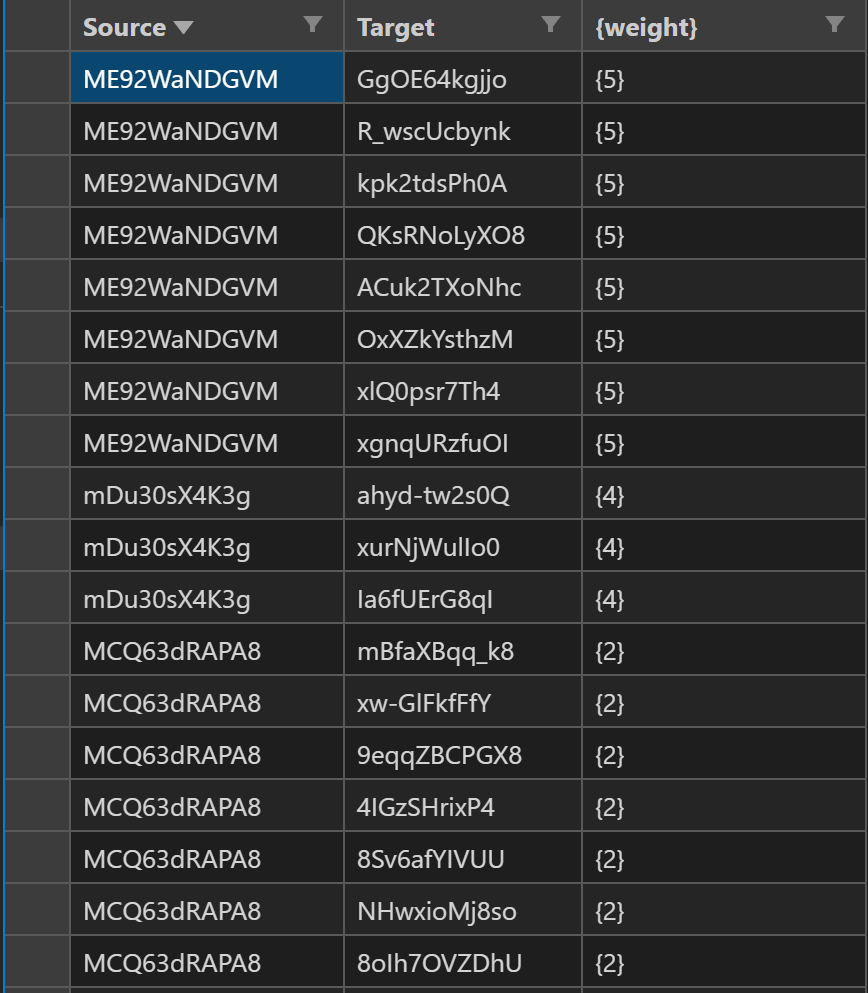
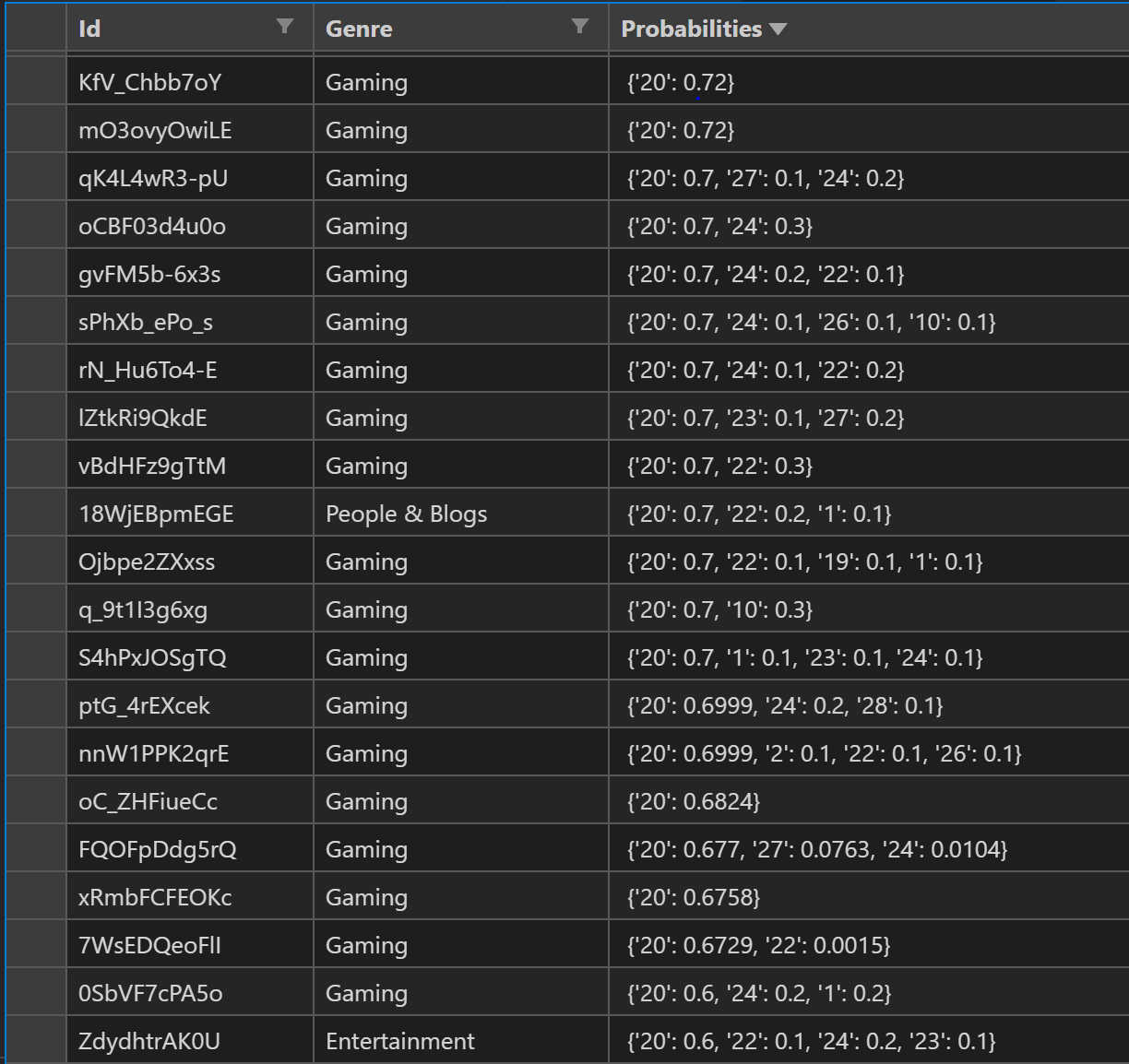
Most social media companies utilize data they have on individuals to provide them with content or advertisements, but the users of interest to us are people that social media, specifically YouTube in our case, has no data on. We are trying to find how the YouTube suggestion algorithm will handle such a user. The videos on YouTube will be our nodes and two videos will be linked if one appears as a suggested video on the page of another. These links will be weighted and directed.

In the midst of its approach, we will record the genres of the suggestions for each video visited, and then move on to the next video at random. We want to determine which, if any, of the genres defined by YouTube tend to be more suggested, or in another sense are easier to get into, which are less suggested, or rather harder to get out of, and which specific videos pop up more as a suggestion (I.E. which are the hub nodes of our network) and if the genres influence which videos those are.

# Screenshot of Our Data

Following is a subset of one of the crawler’s data we will obtain from the videos and then this data will be filtered and weighted into a final compiled graph, which will have the same form.

Nodal data: Edges data:



# Construction of Our Data

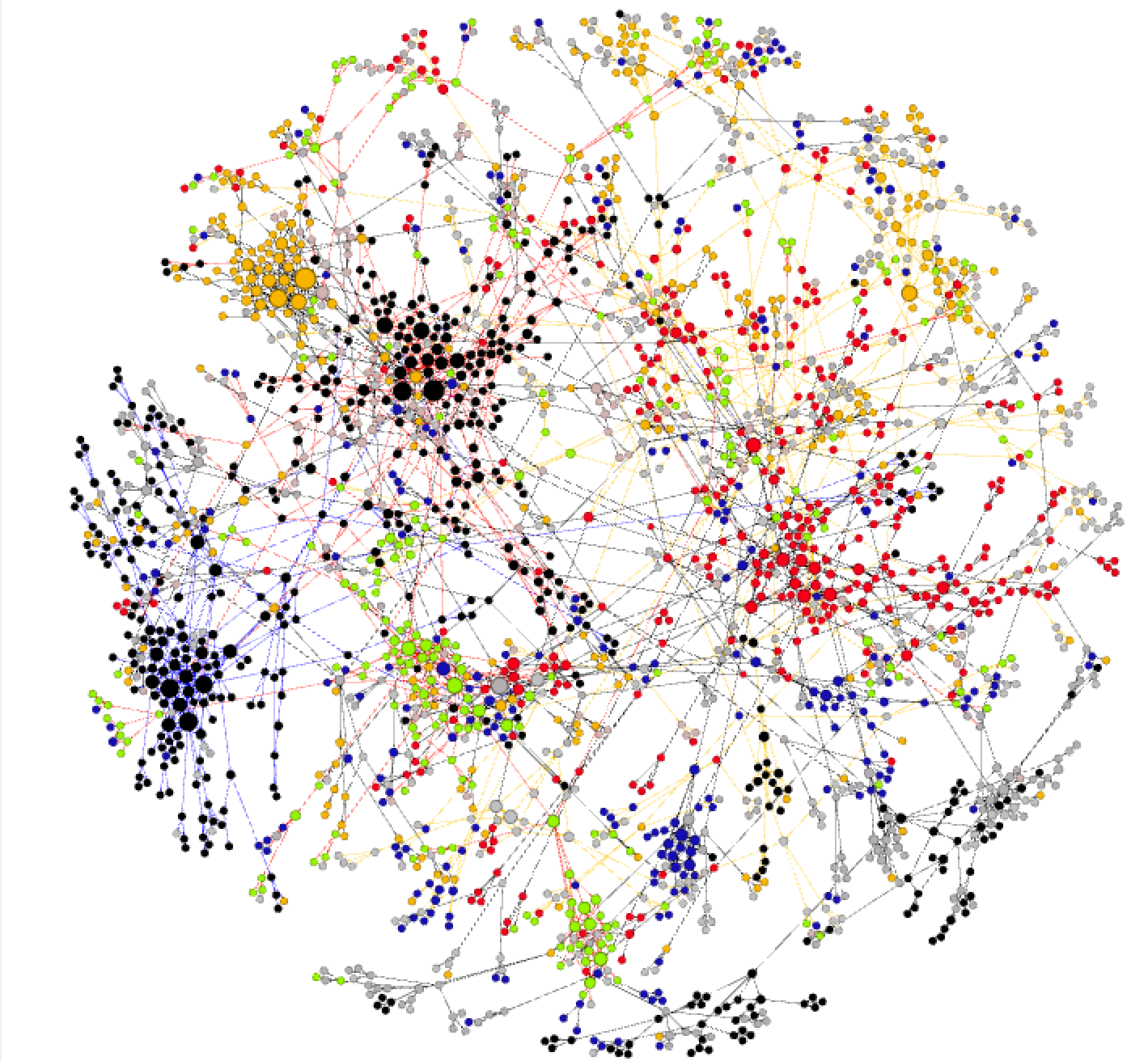
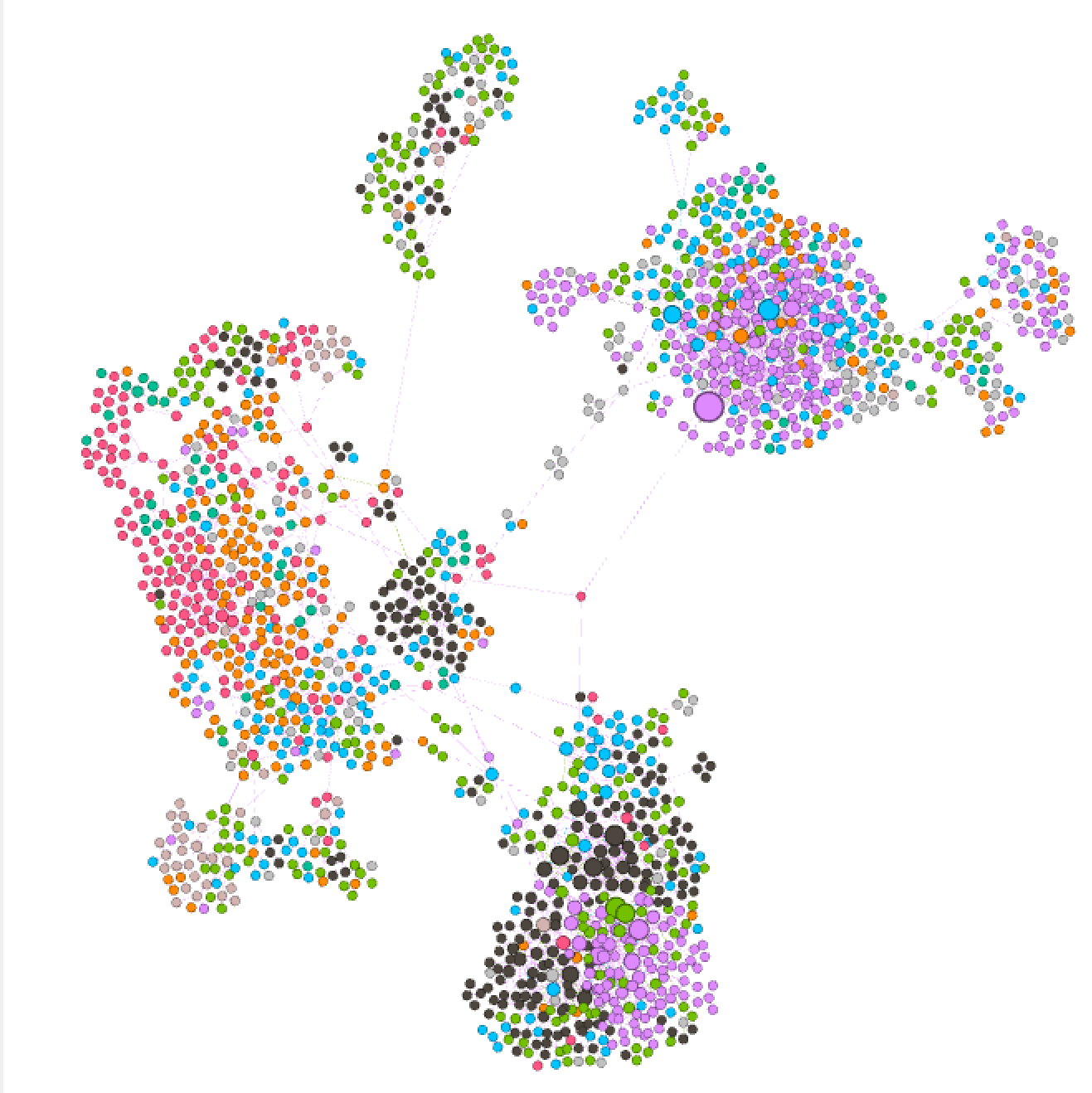
In order to collect a subset of the vast amount of videos that YouTube has to offer, we have created a network crawler in python. This crawler is spun up with a video on which to start, a Time To Live, and a Branch argument. When a crawler is “spawned”, it gets the HTML file provided by YouTube when you visit that video on YouTube.com. It extracts a JSON object stored in that file, which contains the IDs of the suggested videos along the right hand side of the generated page. These are the IDs provided by the suggestion algorithm in question, and form the node IDs in our networks. Also, we extract the genre of this video from the same JSON object for later analysis. After this data is recorded, we use the Data API V3 for YouTube provided by the Google Cloud Platform to get the genres of all suggested videos for this video and record that data as well. We do not save the genres for each suggestion, but rather calculate the percentage of genre that the suggested videos fall under, and add this distribution to the nodal data for the current video.

It is the “watch” argument that appears in each YouTube videos URI that distinguished the nodes (i.e. forms their IDs), the time to live that dictates how many levels of recursion will be executed, and the branch argument that determines how many of the suggest videos the crawler should “watch” next for each level of recursion. Upon each visit to a video, a node is either added or updated in the crawlers graph using NetworkX in python, and upon the return of the recursive call, an edge is either added or updated in the same graph. Once all recursive calls have been exhausted, the program then outputs the generated network as an edge and node list in a csv format. These files can be visualized to get a small picture idea of what is happening.

Once completed, these crawler networks are iterated over and compiled into a final graph. If a node has not appeared, it and its probabilities are added to the final graph. If a node has appeared, its genre distribution is added to the node existing in the final graph, and then normalized. If an edge has not appeared, it is added with a weight of one. If it has, the existing edge has its weight incremented.

# Visualisations

Crawler: Compilation:



# Degree distribution

Within a single Crawler network, the degrees of the average node are low, as there are only a few dozen, to a few hundred nodes depending on how the Crawler was created. The average size of these networks is varying as we determine an efficient and appropriate size for them, but they tend to be sparse. However we are seeing initial hub nodes growing and clustering of genres. The graph overall isn’t quite dense or sparse, however it is interesting to see how some genre clusters tend to be more sparse or dense compared to others.

The Compiled crawler networks show more of the bigger picture. Again we see clustering by genre, however with the broader view we can better see which genres contain hub nodes, which genres are more closely related, and which genres have edges with higher weights.

A general pattern in the data collection has been that the starting genre is suggested to the crawler with varying probability based on which genre it starts in. This leads the crawler to create a cluster of the genre it started in first, and then due to the probabilistic nature of how we chose the next video to watch, the suggestion algorithm leads the crawler to other genres. How quickly youtube shows these other genres determines how clustered nodes for each genre.

# Have your research questions changed?

Our research questions have more or less remained the same, but they have become more focused, and the way in which they are answered has shifted. We previously chose the next videos based on the videos tags and genre, however this was introducing bias and answered the questions in a manner of "How can we move from genre to genre". By choosing the next videos at random with equal probability the methodology behind answer our question shifts to "How does Youtube's algorithm lead us from genre to genre" Since we choose which video to watch next out of the suggested videos at random, the of each genre of each video we watch is determined by YouTubes suggestion algorithm.

We will have a special focus on genres like “News & Politics” and “Education” that could form “bubbles” or “echo chambers”, which tend to be breeding grounds for radicalization, anger, disdain, and tribalism. As for the remaining genres, we aim to see which are related to each other, and if the suggestion algorithm leads a user towards or away from certain genres, if it will bounce users between genres, and which genres tend to pop up more than others. We will look to see if users can become stuck in these topics, and based on our edge weights we can determine how easy or hard it would be to escape that specific genre as the weights are indicative of which videos are being suggested.

We originally asked whether or not some genres are easier to get into compared to other genres and upon collection of the data we now add the following, more focused, questions:

1. Are some genres suggested more by YouTube’s algorithm?
2. Do certain genres cluster more than others?
3. What are the hub nodes like for each genre?
4. Which genres relate to each other? I.E. which are suggested to each other?

# Conclusion

We will examine the paths that YouTube's suggestion algorithm leads users down by selecting videos at random, thus leaving the probabilities up to Youtube’s suggestion algorithm, and which genres a user would traverse into or out of, with a focus on the issue of political radicalization induced by "political bubbles" that can be found on YouTube.