Unraveling Cain's Jawbone: Narrative Detection using Graph Clustering

MAXIMILIAN STOLLMAYER
ABDELBAST NASSIRI

July 2023

1 Introduction

Cain's Jawbone, a literary puzzle book published in 1934 under the author's pseudonym "Torquemada", presents an intriguing murder mystery that has captivated readers for decades. The challenge lies in rearranging the book's randomized 100 pages to unveil the correct sequence of events and solve the crime. This intricate puzzle has been successfully cracked by only three individuals to date, highlighting its complexity and allure.

Our project sets out to apply graph algorithms to Cain's Jawbone in an attempt to shed more light on the puzzle or potentially solve it. One of the main challenges lies in finding the correct reading sequence of the 100 pages. Initially we considered approaching this problem by finding a sequence of the 100 pages that maximizes similarity between them. However, with approximately $100! \approx 10^{158}$ possible permutations, this approach becomes computationally infeasable. Moreover, focusing solely on similarity between pages may not guarantee the correct sequence, as multiple narratives might follow one another without necessarily being similar.

One avenue we explored to remedy this was the application of a graph neural network (GNN) trained on actual books to retrieve a reading sequence for Cain's Jawbone. Although inspired by existing works [4] and [5], which address the traveling salesperson problem using GNNs and reinforcement learning, this approach proved quite complex and too ambitious for the scope of this project. Our experimental efforts can be found within our notebook `GNN.ipynb` on our GitHub [8].

Given the complexity of finding a reading sequence, we shifted our focus to a more feasible task: identifying the six murder victims and their six murderers. The puzzle-solving community theorizes that six distinct narratives surround each murder within the book. To accomplish this, we employed spectral clustering and Louvain community detection algorithms to identify groups of pages corresponding to these narratives.

By employing these graph algorithms, we hope to gain deeper insights into Cain's Jawbone and contribute to the ongoing fascination surrounding this timeless literary puzzle.

2 Methodology

In order to use these graph algorithms on our book we first need to convert it into a graph. Since we do not know the actual ordering of the pages we can consider a fully connected graph, where each node represents a page, with the edges weighted by how similar two pages are.

2.1 Data Preparation

Before constructing the graph, we perform essential and standard routines in natural language processing to clean and standardize the text. The preprocessing steps include:

- Tokenization of the text into individual words.
- Removing special characters, punctuation, and any non-alphanumeric symbols.
- Lemmatization of the words so that they are in their standard form.
- Removing stop words, i.e. common words such as "the", "is" and "and", that are frequently used but add little contextual value.

2.2 Graph Construction

The book can be represented as a fully connected graph, with each page as a node and the edges weighted by how similar the two connected pages are.

To quantify this similarity, we vectorize the pages, which involves calculating the word frequencies on each page and representing them as vectors. By doing so, we create a numerical representation of the book's content.

Now we can use the cosine similarity metric, denoted as $S_C(x, y) = \frac{\langle x,y \rangle}{\|x\|\|y\|}$, where x and y are the vectorized representations of the respective pages. Cosine similarity measures the cosine of the angle between two vectors, which ranges from -1 to 1. However, since our vectors consist of positive frequencies, the cosine similarity will lie within the interval [0, 1]. This similarity score provides insight into how closely related two pages are in terms of their word usage.

2.3 Spectral Clustering Algorithm

Spectral clustering is a powerful technique for partitioning data points based on the Laplacian matrix. It arises as a relaxation to the graph cut problem, where one tries to minimize the number of cuts, or rather the summed value of the cuts, needed to separate the graph into the desired number of partitions.

We implemented spectral clustering using the normalized Laplacian approach described in [1]. The algorithm takes a graph $G = (V, E)$ and the number of clusters k as input and consists of the following steps:

1. Compute the normalized graph Laplacian $L = I - D^{-1/2}WD^{-1/2}$, where I is the identiy, D the diagonal degree and W the weight matrix of the graph.

2. Compute the eigenvectors $u_1, \ldots, u_k$ of L corresponding to the k smallest eigenvalues.

3. Construct the matrix $U \in \mathbb{R}^{|V| \times k}$ containing these eigenvectors as columns.

4. Normalize the rows of U to norm 1.

5. Cluster the the rows of U using k-means clustering from scikit learn [6] into clusters $C_1, \ldots, C_k$. Alternatively one could also try to minimize $x^\top Lx$ with $\|x\| = 1$ using for example Reileigh quotients.

6. Output clusters $A_1, \ldots, A_k$ with with $A_i = \{j \mid y_j \in C_j\}$, where y_j is the j'th row vector of the matrix U.

The resulting partition obtained from the spectral clustering algorithm represents different clusters within the graph. We consider each cluster as a potential narrative within the book Cain's Jawbone.

2.4 Louvain Algorithm

In order to compare the results of the spectral clustering we will also use the Louvain algorithm. The Louvain algorithm is a community detection algorithm. It was proven to be effective in detecting cohesive clusters within complex networks. In our implementation, we used `louvain_communities` function provided by the NetworkX library in Python [7]. This function iteratively optimizes the graph partitioning by maximizing the modularity measure, which is a measure of the quality of the clustering. For more mathematical information see [2].

3 Comparison

To compare the effectiveness of the two clustering algorithms, we can utilize two distinct approaches: quantitative analysis and qualitative analysis. Given the complexity and difficulty of the book's content, conducting a full-fledged qualitative analysis using domain knowledge becomes impractical. Instead, we opted for a quick examination of the best clusters produced by the two algorithms to observe any discernible differences.

For a quantitative approach to gauge the "goodness" of the clusters generated by each algorithm, we need a suitable measure. One such measure is the concept of coherence. Coherence, in the context of topic modeling, refers to the thematic consistency and meaningfulness of the topics. Coherence is a measure that evaluates the relatedness and coherence of words within clusters. It helps determine if the clusters reflect cohesive narratives or themes present in the dataset. See [3] for more mathematical details and how to apply it.

We ran each algorithm 100 times to account for their inherent randomness of the clustering. The computed mean coherence scores for the spectral clustering and the Louvain algorithm were found to be 0.658 and 0.652 respectively. These scores indicate a slight advantage in favor of the spectral clustering algorithm, suggesting that its output clusters exhibit better thematic coherence and overall consistency on average compared to the alternative method. Since both score much higher than 0.5 we consider them to be very coherent. A simple visual

inspection of both of the best scoring clusters with a coherence of roughly 0.683 achieved by both algorithms shows that these clusters are very similar.

4 Results

The application of spectral clustering and the Louvain algorithm allowed us to identify cohesive clusters of the nodes within the constructed graph. These clusters (possibly) correspond to the distinct narrative threads in Cain's Jawbone and if we continue with trying to recover a reading sequence this helped us reduce the seach space of possible permutations from $100! \approx 10^{158}$ to the number of permutations within each cluster, roughly 10^{23}, which is an improvement of 135 orders of magnitude!

For the full analysis and plots of these clusters see our notebook `graph_analysis.ipynb` on our GitHub repository [8].

References

[1] Ulrike von Luxburg. "A tutorial on spectral clustering". In: (2007). URL: https://arxiv.org/abs/0711.0189.

[2] Renaud Lambiotte Vincent D. Blondel Jean-Loup Guillaume and Etienne Lefebvre. "Fast unfolding of communities in large networks". In: (2008). URL: https://arxiv.org/abs/0803.0476.

[3] Andreas Both Michael Röder and Alexander Hinneburg. "Exploring the Space of Topic Coherence Measures". In: (2015). URL: http://svn.aksw.org/papers/2015/WSDM_Topic_Evaluation/public.pdf.

[4] Herke van Hoof Wouter Kool and Max Welling. "Attention, Learn to Solve Routing Problems!" In: (2019). URL: https://arxiv.org/abs/1803.08475.

[5] Louis-Martin Rousseau Chaitanya K. Joshi Quentin Cappart and Thomas Laurent. "Learning the Travelling Salesperson Problem Requires Rethinking Generalization". In: (2022). URL: https://arxiv.org/abs/2006.07054.

[6] *K-means Clustering in SciKit Learn.* URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans.

[7] *Louvain Community Detection in NetworkX.* URL: https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.community.louvain.louvain_communities.html.

[8] Abdelbast Nassiri and Maximilian Stollmayer. *GitHub Repository: Graph Algorithms Project.* URL: https://github.com/maxstolly/Graph-Algorithms-Project/.