Modeling Movie Character Networks with Random Graphs

Andy Chen

December 2017

Introduction

For a novel or film, a character network is a graph whose nodes represent individual characters and whose edges represent interactions between those characters. Depending on our topic of interest, we can choose what relationships these network edges symbolize: conversations, co-occurrence in scenes, etc. Such networks can illuminate social relationships at a macroscopic level, giving us insight into the overall structures. For instance, we might wish to know which characters reinforce stereotypes in movie dialogue or what proportion of the characters participate heavily in conversation.

In the past years, written and filmed media has become more readily available in digital form. These electronic formats allow us to easily gather tabulated information about these works at a large scale. This means we can not only analyze the structure of one novel or film at a time, but also to easily compare character interactions between hundreds of different works. Graph theoretic techniques can quantitatively express aspects of character networks and thus permit data analysis on different written works.

One major theme of social network research involves determining how the networks in question were generated. In other words, we often wish to know what underlying process generated the observable character network. Knowing this information could help address questions such as:

- How closely do writer-created social interactions resemble natural social networks?
- How might authors initially conceive of characters in their plot?

In class, we discussed random undirected graph models such as the Erdos-Renyi and Watts-Strogatz small-world models, which serve as simplified versions of social phenomena. However, much of existing work focuses on undirected simple networks, which inherent lack potentially vital information about the character interaction.

In the following work, we first construct character networks from raw movie dialogues. Then, we choose several generative stochastic graph models that could potentially model the character networks. To determine which ones most accurately represent those character networks, we design a classifier that predicts which generative model the character networks belongs to.

Literature Review

Much of the previous work on characterizing character networks explores undirected graphs. In "Mining and Modeling Character Networks", Bonato et al. attempt to characterizing undirected character networks; they use co-occurrence to match undirected edges between characters, causing the resulting networks to be dense in edges.[4] They then build upon previous literature on picking features from undirected graphs. In particular, they examine the topologies of "graphlets", or subgraphs of k nodes. In addition, since the Laplacian matrices of these undirected networks are symmetric and nonnegative, their eigenvalues are all real and reveal some attributes about the network (see Spectral Graph Theory).[12] These features allow Bonato et al. to distinguish between these different undirected graph random models.[4]

There appears to be less focus on finding features for directed graphs. However, scholars have explored the directed graph analogs of the features mentioned above. For instance, Bauer defines a general Laplacian operator for directed (and possibly weighted) graphs.[3] In addition, Aparicio et al. analyze the graphlets of various sizes k (subgraphs constructed by taking k nodes from the original graph).[2]

As an analog to the undirected Chung-Lu random graph model, Durak et al. propose the Fast Reciprocal Directed (FRD) graph generator. The generative process is somewhat similar to that of the directed Chung-Lu model.[8] However, it also incorporates information about *reciprocated edges*, which they argue are often missing from graphs generated by the directed Chung-Lu model.

There are also multi-edged graph analogs of undirected simple graph models such as Preferential Attachment and Erdos-Renyi. Shafie, for instance, discusses a class of multigraphs generated by Independent Edge Assignment (IEA).[11] Rath and Szakacs also discuss the multigraph variants of the configuration model and the preferential attachment model.[10]

Methods

To find the random graph model(s) that best describe character networks based on conversation, we follow three main steps:

1. Extracting the character social networks from the raw dataset
2. Selecting the random graph model candidates
3. Generating random graphs from a character network and training a classifier to distinguish between those random graphs (Figure 4 yields a visual representation of this last step)

Extraction of Social Networks

Dataset

As raw data for constructing the character networks, we use the Cornell Movie Dialogs Corpus, which contains a total of 220,000 conversational exchanges between about 9000 total characters across 617 movies.[7] Note

Figure 1: Example of Movie Conversation

PATRICK	A soft side? Who knew?
KAT	Yeah well don't let it get out
PATRICK	So what's your excuse?
KAT	Acting the way we do.
PATRICK	Yes
KAT	I don't like to do what people expect.
.	Then they expect it all the time and they get disappointed when you change.
PATRICK	So if you disappoint them from the start you're covered?
KAT	Something like that
PATRICK	Then you screwed up
KAT	How?
PATRICK	You never disappointed me.

that each conversation recorded takes place between exactly two characters, who alternate lines. In addition to the raw text of the conversations, we also receive metadata for the movies themselves, containing genre information.

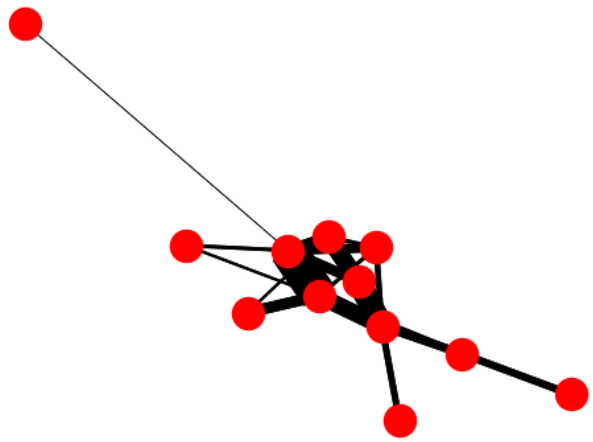
Multi-Edge Networks

Given the corpus of movie dialogs, we may design a character network for each movie as follows:

- Represent each character A in a movie by a node labeled A
- For any two characters A and B in the same movie, draw an undirected edge AB for every conversation they have. Note that the number of edges between is often zero.

We can view the resulting graph as either a multi-graph (a graph that can have multiple edges between nodes, but no self-loops) or a weighted graph with integer weights on the edges. Such a graph may look similar to Figure 2. Note that there is one connected component, as every character present in the corpus

Figure 2: Example of a Multigraph Character Network



participates in at least one conversation. In many of the character networks, there also appears to be several nodes forming a cluster with high edge density i.e. many edges between a few vertices.

Directed Networks

We construct one example of a directed character network as follows:

- Represent each character A in a movie by a node labeled A
- For any two characters A and B in the same movie, we draw an edge from character A to character B if there exists a conversation for which A utters more words than B. (If we were constructing weighted networks, then the edges could represent a more quantitative measure.)

Each movie network contains up to 44 nodes and up to 77 edges. Figure 3 displays an example of such a character network. Note that many, but no all of the edges, are reciprocated i.e. an edge from character A to character B also has an edge from character B to character A. Note also that this method of constructing networks only produces unweighted graphs.

Selecting Random Graph Models

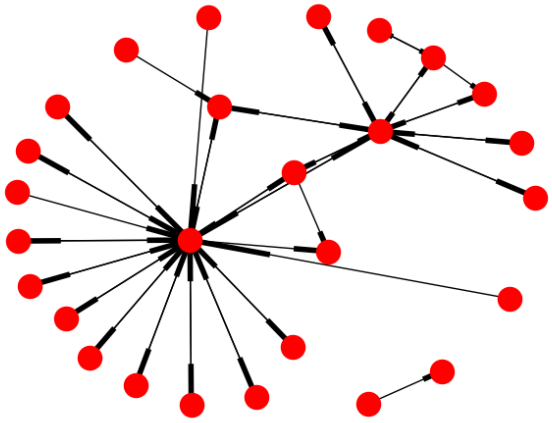
Next, we must choose candidate random graph models; in the next classification step, we determine which of these candidates is most likely to generate a graph resembling the original character network. We describe how to generate a sample of each random model given the original character network.

Multi-Edge Networks

We see that empirically, the multi-edged character networks have many of their edges centralized. In other words, a significant portion of the edges only connect a few nodes in total. Therefore, we choose random graph models that either 1) reflect the high degrees of a few nodes (likely representing main characters) or 2) accurately reflect the degree distributions of the original character networks.

The generative multi-graph models that we use as possible labels are:

Figure 3: Example of a Directed (Simple) Graph Character Network



- Erdos-Renyi: given the number of vertices V and edges E in the original graph, we construct the new graph as follows: for each of E added edges, choose two distinct nodes A and B uniformly at random and add an edge (A, B). This model is used as a null baseline; ideally, the original character networks should not resemble these random graphs.
- Configuration Model: the multigraph version of the configuration model is identical to the simple graph version, but we do not reject configurations that have multiple edges (though we do reject ones with self-loops). Given that the original graph has V nodes, with degrees d_i for $i = 1, \ldots, V$, we perform the following operations:
  - Start with a graph with V nodes and zero edges
  - Create an array A of length $2E$ such that the element i appears d_i times
  - Shuffle the array
  - For $j = 1, 2, \ldots, E$, add an edge $(A[2j-1], A[2j])$. If $A[2j-1] = A[2j]$, then we reshuffle the array and repeat.
- Preferential Attachment: this scheme is sometimes referred to as the "rich get richer" scheme because new edges are added to nodes that already have high degree. Specifically, each new edge follows a Chinese Restaurant Process distribution in selecting its endpoints. Formally, the process for creating a new random graph given the original character network with V nodes and E edges is:[10]
  - Start with an empty list L of nodes
  - For $i = 1, \ldots, 2E$, choose a node j with probability $\frac{d_j + p}{i - 1 + pV}$, where $p = 0.5$ is a custom hyperparameter and d_j represents the current number of times j appears in L
  - Create a new graph with V nodes, and add edges $(L[2i-1], L[2i])$ for $i = 1, \ldots, E$.
- Chung Lu: this algorithm uses the indegree and outdegree distributions of the original character network. Suppose in the character network, the indegree of node i is $d_i^{\Leftarrow}$ and the outdegree of node i is $d_i^{\Rightarrow}$. Given two nodes i and j, the probability of an edge between then existing is:

$$p_{ij} = \frac{d_i^{\Rightarrow} d_j^{\Leftarrow}}{\sum_{a,b \in V} d_a^{\Rightarrow} d_b^{\Leftarrow}}$$

  where we don't account for self-loops.

Directed Networks

The directed character networks tend to be sparse, with a large proportion of reciprocated edges. They also tend to have a few central nodes, which likely represent main protagonists or antagonists. This means that we wish to find generative graph models that generally produce networks with these properties.

- Erdos-Renyi: given the number of nodes V and the number of edges E in the original graph, we generate a random graph with V nodes. Each edge in the graph has probability p of existing, where we set:

$$p = \frac{E}{V(V-1)}$$

  (Note that since edges are directed, the total number of possible edges is $V(V-1)$.) This model acts as the null baseline for modeling the character network.

- Chung-Lu: this random model uses more information from the indegree and outdegree distributions over the nodes. In particular, for the original character network, suppose the indegree of node i is $d_i^{\Leftarrow}$ and the outdegree of node i is $d_i^{\Rightarrow}$. Given two nodes i and j, the probability of an edge between then existing is:[6]

$$p_{ij} = \frac{d_i^{\Rightarrow} d_j^{\Leftarrow}}{\sum_{a,b \in V} d_a^{\Rightarrow} d_b^{\Leftarrow}}$$

- Fast Reciprocal Directed Model (FRD): Durak et al. argue that the above Chung Lu model (or variants of it) produce graphs that rarely have reciprocated edges, which is often not true of natural character or social networks.[6]

  In this FRD algorithm, we first find the node distributions $\{d_i^{\Leftarrow}, d_i^{\Rightarrow}, d_i^{\Leftrightarrow}\}$ for indegree, outdegree, and reciprocated degree, which is the number of neighbor nodes connected with both directions of edges. If E is the total number of edges and $E_{\Leftrightarrow}$ is the number of (unordered) pairs of nodes with reciprocal edges, then to construct a random FRD graph G as follows:[8]

  - Choose two distinct nodes i, j according to the distribution $\{d_i^{\Leftrightarrow}\}$. Then, add edges (i, j) and (j, i) to G. Repeat the above process for a total of $E_{\Leftrightarrow}$ reciprocated edges.
  - Choose node i according to the distribution $\{d_i^{\Rightarrow}\}$ and choose node j according to the distribution $\{d_i^{\Leftarrow}\}$. Then, add edges (i, j) to G, unless doing so creates multiedges or self loops. Repeat the above process for a total of $E - 2E_{\Leftrightarrow}$ single edges.

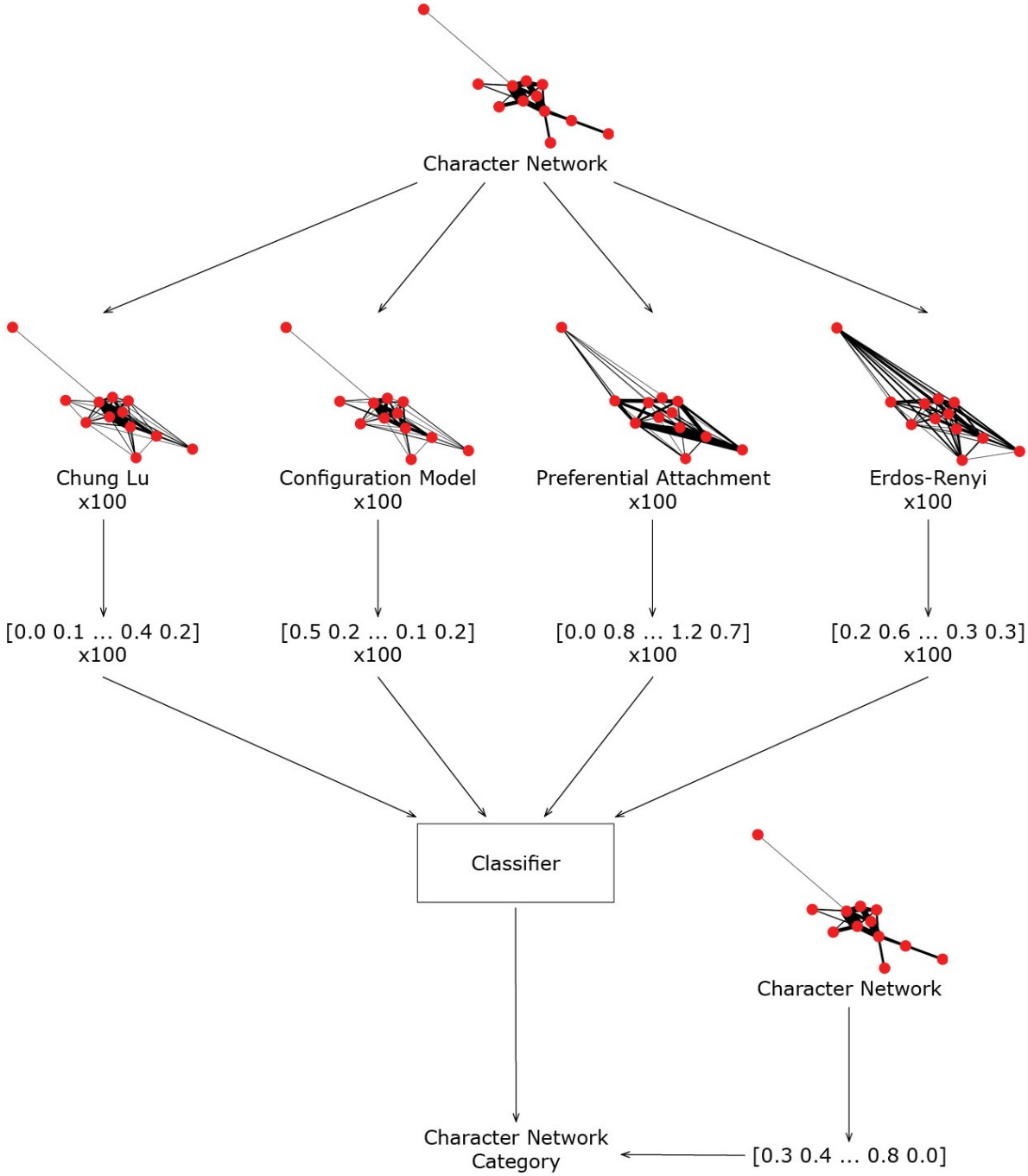
- Preferential Attachment: given the number of nodes V (each with outdegree d_i) and the number of edges E in the original graph, we create a new graph as follows:
  - Start with a graph with V nodes and zero edges.
  - For each node i, for $j \in \{0, 1, \ldots, d_i - 1\}$:
    - With probability $p = 0.2$, choose a node $k \neq i$ uniformly at random and add edge (i, k)
    - Otherwise, choose a node $k \neq i$ with probability proportional to the current indegree D_k

Training Graph Classifiers

We refer to Figure 4 for a visual summary of the classification process. For each of the 617 movies' character networks (either multi-graph or directed graph):

- We generate 100 samples of each of the four graph model candidates, as per the parameters of the original character network.
- We convert the 400 graphs into pairs (feature vector, random graph model label) and shuffle them.
- We then train a classifier on 80% of those graphs, using the remaining 20% to measure test accuracy.
- We finally feed the original character network's feature vector into the classifier

Figure 4: Visualization of Classification Process



Feature Choice

We must also choose which features to extract from a given graph, multi-edged or directed. Ideally, these features have high separation between different categories of random models. However, it is also important that we do not choose features that are trivially different between the categories of random models. For instance, using the degree distributions as features would cause the classifier to almost always predict the character network's category as Chung-Lu. This is because we construct the Chung-Lu samples *specifically from the degree distribution* of the original character network.

Currently, we make use of the normalized Laplacian matrix. Given a graph G with n vertices, we define the matrix L_G to be:

$$(L_G)_{ij} = \begin{cases} 1 & \text{if } i = j \text{ and OutDegree}(i) > 0 \\ -\frac{N_{ij}}{\text{OutDegree(i)}} & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases}$$

where N_{ij} is the number of edges (i, j). In the undirected multi-graph case, the eigenvalues are all real because the Laplacian is symmetric. However, L_G is not necessarily symmetric in the directed case; still, the eigenvalues do still serve as useful features of the graph. For instance, the number of zero eigenvalues corresponds to the number of strongly connected components. We choose to use the normalized Laplacian, whose eigenvalues have real parts with absolute value at most 2. We can bin the real parts of the eigenvalues of a Laplacian into equally-sized intervals (empirically, the imaginary parts of the eigenvalues are quite small in comparison, so using the magnitudes would yield similar results). Thus, if we have b buckets, we receive a b-dimensional feature vector, which can also be concatenated with other feature vectors. We choose to not use the eigenvalues themselves as features, as the feature vector should not depend on the order of the eigenvalues (ex. attempting to feed the sorted eigenvalues as features should not intuitively be mapped to a space with independent basis vectors).

In addition to the above eigenvalue histogram features, we also use betweenness centrality (on nodes) as a source of features. Betweenness centrality is a metric on nodes and edges; intuitively, it measures how often the node or edge lies on shortest paths in the graph. Formally, if σ_{CD} is the number of shortest paths from node C to node D and $\sigma_{CD}(A)$ is the number of such paths that also pass through A, then the betweenness centrality is:

$$B(A) = \sum_{A \neq C,D} \frac{\sigma_{CD}(A)}{\sigma_{CD}}$$

This metric is used in community detection algorithms such as the Girvan–Newman algorithm. Again, since we are interested in the relative centralities across different nodes, we first normalize the centrality values to take range $(0.0, 1.0)$. Then we compile the resulting normalized betweenness centralities into a histogram, which is used as part of the feature vector for each graph.

We also analyze *graphlets / graph profiles* as sources of features. We examine every group of $k = 3$ nodes and find the distribution over all possible graph topologies (i.e. whether the group of nodes has zero edges, two edges connecting the same two nodes, etc.) However, this approach is only applicable to the directed graph case, as pairs of nodes can contain arbitrarily high numbers of edges in the multigraph case. In experimentation, however, this approach to feature extraction yields poor results.

Results and Analysis

We measure the mean accuracies of the multi-graph and directed graph classifiers, as these accuracies reflect our confidence in the predictions for the original character networks' categories. The mean accuracies are taken over all 617 classifier models, one for each character network.

Figure 4 contains the mean accuracies for the multi-graph classifiers; we test with four different multi-label classifier algorithms. Figure 5 contains the same information for the directed graph classifiers. We can see that in both cases, the K-Nearest Neighbors classifier performs best.

Finally, we feed each original network into its corresponding classifier, which yields the random graph model that the network most closely resembles. Figure 6 shows the distribution of predictions for the multi-graph networks, and Figure 7 shows the distribution for the directed networks.

Figure 5: Multi-Graph Classifier Accuracies

Classifier Algorithm	**Mean Training Accuracy**	**Mean Test Accuracy**
Support Vector Classifier	49.36%	46.69%
AdaBoost	66.28%	61.88%
K-Nearest Neighbors	**82.35%**	**72.67%**
Stochastic Gradient Descent	69.81%	67.38%

Figure 6: Directed Graph Classifier Accuracies

Classifier Algorithm	**Mean Training Accuracy**	**Mean Test Accuracy**
Support Vector Classifier	57.27%	51.13%
AdaBoost	67.25%	64.38%
K-Nearest Neighbors	**90.11%**	**84.81%**
Stochastic Gradient Descent	81.67%	77.50%

Figure 7: Multi-Graph Character Network Labels

Graph Model	**Proportion of Networks**
Multi-Graph Erdos-Renyi	1.13%
Multi-Graph Configuration Model	35.66%
Multi-Graph Preferential Attachment	**48.95%**
Multi-Graph Chung Lu	14.26%

Analysis and Future Directions

We can see that while no graph model is consistently the chosen label for the original character network, Preferential Attachment is the most frequent label, with roughly half of the 617 final labels as Preferential Attachment. This result matches our intuition about the general structure of a film's characters; there are often a few main characters that serve as primary protagonists or primary antagonists. Most of the movies' conversations likely involve at least one of these main characters in order to give the audience context for those conversations, meaning that most of the character networks' edges likely connect to a few nodes representing these main characters. This pattern is common in preferential attachment models.

The results for directed graphs are also consistent with previous literature and intuition. (Our results for directed networks are actually more conclusive than those for multi-edged networks, as the mean accuracies are higher for the directed graph classifiers.) Empirically, the directed graphs resemble undirected graphs, meaning that the motivation behind Bonato et. al.'s approach likely applies here. Indeed, more than 40% of the character network labels are Chung Lu. This model captures information about each node's degree distribution (both indegree and outdegree) that a null model (such as directed Erdos-Renyi) cannot. In addition, directed preferential attachment does not yield high reciprocity of edges; several nodes may have high indegree, but they may not be guaranteed to have similar outdegree.

Figure 8: Directed Character Network Labels

Graph Model	Proportion of Networks
Directed Erdos-Renyi	3.24%
Fast Directed Reciprocal	**49.92%**
Directed Preferential Attachment	5.51%
Directed Chung Lu	**41.33%**

However, Bonato et. al. did not study the Fast Directed Reciprocal model, which theoretically better matches the high reciprocality of the character networks' edges. While generating directed Chung Lu graphs requires only information about the degree distributions, generating FRD graphs also requires information about which nodes have reciprocated edges.

The directed character networks we use are based on conversation rather than co-occurrence (unlike Bonato et al). However, their topological resemblance to Bonato et al's undirected networks still means that our results mostly serve to reinforce Bonato's discoveries. That said, our findings do show that for other types of directed character networks, we can repeat the above classification process (with the same features).

For future work, we could construct and test additional features of our random graphs, especially the generated multigraphs. Finding the appropriate features would allow our classifiers to increase in mean accuracy. This would let us be more confident in our categorical predictions of the original character networks.

We also consider adapting our classification architecture for dynamic character networks i.e. networks that change over time. Aspects of the characters' social interactions may not be apparent in a static character network, which means that analyzing possible random dynamic graph models could yield additional insight into the movies' social interactions.

Special Thanks

We would like to thank Poorvi Bhargava for providing feedback on our initial proposal and for offering useful advice for this project's direction. We would also like Jure Leskovec and the CS 224W TA staff for putting together the class and providing the course material to motivate this project.

References

[1] Agaev, R., & Chebotarev, P. (2005). "On the spectra of nonsymmetric Laplacian matrices." *Linear Algebra and its Applications*, 399, 157-168.

[2] Aparício, D., Ribeiro, P., & Silva, F. (2015). "Network comparison using directed graphlets." arXiv preprint arXiv:1511.01964.

[3] Bauer, F. (2012). "Normalized graph Laplacians for directed graphs." *Linear Algebra and its Applications*, 436(11), 4193-4222.

[4] Bonato, A., D'Angelo, D. R., Elenberg, E. R., Gleich, D. F., & Hou, Y. (2016). "Mining and modeling character networks." In *Algorithms and Models for the Web Graph*: 13th International Workshop, WAW 2016, Montreal, QC, Canada, December 14–15, 2016, Proceedings 13 (pp. 100-114). Springer International Publishing

[5] Burstein, D. (2017). "Asymptotics of the spectral radius for directed Chung-Lu random graphs with community structure." arXiv preprint arXiv:1705.10893.

[6] Chung, F. (2006). "The diameter and Laplacian eigenvalues of directed graphs." *The Electronic Journal of Combinatorics*, 13(1), N4. Chicago.

[7] Danescu-Niculescu-Mizil, C. (2011) "Movie Dialog Corpus." [Dataset]. Retrieved from Kaggle.

[8] Durak, N., Kolda, T. G., Pinar, A., & Seshadhri, C. (2013, April). "A scalable null model for directed graphs matching all degree distributions: In, out, and reciprocal." In *Network Science Workshop (NSW)*, 2013 IEEE 2nd (pp. 23-30). IEEE.

[9] Elenberg, E. R., Shanmugam, K., Borokhovich, M., & Dimakis, A. G. (2015, August). "Beyond triangles: A distributed framework for estimating 3-profiles of large graphs." In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 229-238). ACM.

[10] Ráth, B., & Szakács, L. (2012). "Multigraph limit of the dense configuration model and the preferential attachment graph." *Acta Mathematica Hungarica*, 136(3), 196-221. Chicago

[11] Shafie, T. (2015). "A multigraph approach to social network analysis." *Journal of Social Structure*, 16, 0_-1.

[12] Roughgarten, T, and Valiant, G. (2017). "Spectral Graph Theory." *The Modern Algorithmic Toolbox*, Stanford University.