Andrew Vande Guchte and Jack Lint

SI608

11/17/19

Networks Project Update

**Motivation and Research Question:**

Recommender systems have become commonplace, even necessary, in the modern marketplace. This is driven by the “long-tail” problem, where the advent of internet-based retailers has given shoppers access to seemingly infinite choices and products have become targeted to more and more niche customers.

A traditional approach to recommender systems is to use a collaborative filter which leverages similarities between how users rate items and how items are rated to construct bi-partite graphs and then create predictions on how much a user will like an item based off of those similarities. However, collaborative filters struggle when a user has rated few items or an item has few ratings. One method to avoid this issue is to build networks that connect users and items in other ways (such as by the characteristics of the users or items). In order to show their viability, we aim to build several network-based recommender systems in an attempt to beat the accuracy of a simple collaborative-filter baseline model.

**Related Work:**

Of course, our data comes from the Netflix challenge, so there are a number of documented methods that were used to approach building a recommender system for this dataset. Our goal, however, is not to improve upon those methods, but to build a recommender system from scratch using networks methods in order to identify which networks provide the most predictive power.

As such, much of the related work we will be relying upon describes the algorithms and methods we intend to use. One such algorithm is the ICA algorithm, described in Bhagat et al. 2011[[1]](#footnote-0). Another is an absorptive approach (where the random walk starts from an unlabelled node and continues until it reaches a labeled node, or “sink”) such as the LP-Zhu algorithm, but as a random walk (also described in Bhagat et al. 2011, but originally described in Zhu et. al (2003)[[2]](#footnote-1). If we have time, we would also like to implement the “Hitting Time” algorithm (Mei et al. 2008[[3]](#footnote-2)).

**Data Collection and Preparation:**

For this project we used data from two different sources. The first source being the original Netflix challenge data. This data set contains a list of movies, ratings that each of the movies have received from specific users. The other dataset that we used came from IMDB. By using the IMDB data we thus were able to get a significantly larger amount of information about each of the movies. Through IMDB we were able to find the actors, directors, genres, amongst other possibly useful information. Because the main part of our project revolves around using different types of networks to help with movie rating predictions, we needed to make sure that the movies in the netflix data could actually be matched with the IMDB data. Thus, the first step in preparing this data was to actually link the netflix data with the IMDB data, and that was done through creating a perfect on the title of the movie and the release year of the movie. We process the data even further so that in our case we would just be looking at films that were of the category ‘movie’ rather than including tv shows and other types of films. We made this choice as we felt that considering just movies will keep our networks with say actors more explicit because actors in tv shows may change over seasons. This left us with just under 8000 movies in the netflix dataset that had been successfully matched with its IMDB data. From their for speed purposes we decided that it might be best to subsample that dataset down even further, and as of now our initial findings are including ratings from 2500 movies in the netflix dataset.

Now that we have the data that we wanted and needed, we still had a lot to do to get it into a bunch of different networks. For our network generation we had two different paths taken, the first was having movies be nodes. The second was having the users be the nodes. The first type of networks, were created mainly by using the IMDB data about the movies. For instance we have thus far created 4 different networks with the nodes being movies.The first being movies linked by matching genres, a movie can have at most 3 genres listed, therefore the weights between links are based on the set of overlapping genres between movies. The second being a network of movies that are linked if they were directed by the same person. The third and fourth network are relatively similar in that they both involve connecting movies by actors/actresses. The first of these two networks though is created by looking at only actor’s most popular (well known) movies. This is done in the IMDB data by listing at most four movies for a given actor, in which they are most well known for. For instance, Brad Pitt is listed for being well known for four movies, but as we all know he has been in many more. The final network that we have generated thus far with nodes being the movies, is also connected by actors, but this time not looking at just the maximum of four movies that an actor can be known for. The other network that we made was using the users who have given ratings as the nodes. The connection between nodes being if they have rated at least one of the same movies, and the weight of the connection being the dot product between the two different users rating vectors.

<https://archive.org/details/nf_prize_dataset.tar>

<https://www.imdb.com/interfaces/>

**Exploratory Data Analysis:**

Our initial exploratory analysis has focused on learning features of the movie-based networks that we have constructed (the user-based network will be produced later). The four movie-based networks are surprisingly different from one another, as is summarized below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Network | % Largest Comp. (LC) | Diameter of LC | Avg Shortest Path of LC | Average Degree | Average Clust. Coeff. |
| Genre | 100.0% | 3 | 1.43 | 1408.90 | 0.81 |
| Director | 1.0% | 5 | 2.80 | 3.47 | 0.60 |
| shortList Actors | 99.9% | 6 | 2.39 | 73.69 | 0.17 |
| fullList Actors | 99.9% | 8 | 2.68 | 40.80 | 0.30 |

The Genre network is incredibly dense because many movies share genres. Thus, the diameter is extremely small. Correspondingly, the degree distribution is weighted towards higher values, with the average being 1408. Because of this, we expect the Genre network to actually perform poorly as part of a recommender system; there will be lots of noise since movies are very highly connected.

In contrast, the Director network is extremely sparse. There are 1045 components to the Director network, the largest of which only contains ~1% of the overall movies. This is likely due to the fact that most movies only have one or two directors, and therefore it is rare to have lots of movies connected by the same director. We expect the Director network to also have very little predictive power; this time because there are so few connections to exploit between movies.

The Actors networks appear to have hit the sweet spot. There is one large component for these two networks, and they are not as dense as the Genre network but considerably more dense than the Director network. In particular, the fullList Actor network has a clustering coefficient of 0.30 and an average degree of 40, which implies that distinct communities may have formed around certain types of movies.

**Baseline Modeling:**

As a baseline model, we built a recommender system that simply applies the average deviation of each user and each movie from the global mean rating. This baseline was able to produce a mean absolute error of 0.74 and an RMSE of 0.94, which means that even this baseline was able to predict each rating within “one star” on average and was only 0.09 RMSE away from the original winner of the Netflix Challenge (granted, we are only using a small subset of the Netflix Challenge data). Given the high results of this basic model, it may not be possible to improve it with our network-based approach, however, we may be able to combine the two approaches somehow in order to produce better results.

**Next Steps:**

The immediate next step is to build an absorptivity recommender system using the movie-based networks that we currently have. Then, we will attempt to implement the hitting time algorithm. Finally, we would like to apply these approaches to a user-based network as well and compare the results of all of our models to the baseline we created.

1. Bhagat S., Cormode G., Muthukrishnan S. (2011) Node Classification in Social Networks. In: Aggarwal C. (eds) Social Network Data Analytics. Springer, Boston, MA [↑](#footnote-ref-0)
2. Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In Proceedings of the Twentieth International Conference on International Conference on Machine Learning (ICML'03), Tom Fawcett and Nina Mishra (Eds.). AAAI Press 912-919. [↑](#footnote-ref-1)
3. Mei, Qiaozhu, et al. “Query Suggestion Using Hitting Time.” *Proceeding of the 17th ACM Conference on Information and Knowledge Mining - CIKM '08*, 2008, doi:10.1145/1458082.1458145. [↑](#footnote-ref-2)