Social Network Analysis on IMDb
Team 2

Abhishek Mahadevan Raju *(1306162)*,
Jonne Deurloo *(1388797)*,
Natarajan Chidambaram *(1358111)*,
Qiang Fang *(0894902)*

1 Introduction

ANALYSIS of social networks has attracted considerable interest and curiosity in the field of sociology and behavioral science. It helps identifying the relationships and patterns between social entities. This is used in the field of mobility of people and their roots, social relationship, publications and topics in which a person has worked on, trade exploration, contacts maintained and suggested known people with second degree friend, third degree friend, and many other fields. Some of these topics also allow for weights in the connections (edges) to mark its importance. Applying graph theory on social network analysis makes the task more meaningful and helps in identifying the relations more clearly rather than the conventional approach of placing the points on a space and calculating the distance between them to get the degree of relation. This further helps in identifying the centrality through measure of adjacency. Identifying this point will make the searches even more optimized and takes less time to give the cardinality estimate.

This project uses the IMDb data set which has details on movies, actors and the crew involved in creating the movie. Possibilities with analyzing this social network can be to help identify the number of movies the actors and crew members are linked with, the movies that create connections between actors or the choice of actors, and crew to get the best possible movie. It can even help to identify the type of movie that needs to be created to get the maximum profit out of it.

But to capture some more meaningful hidden structures from the data, we need to perform some analysis using different algorithms. One of such algorithm is clustering. This helps by grouping similar data points and understand the relation between these nodes at a higher level. This further helps in analyzing the structural connectivity between two groups/social networks. On the other hand we can also form these clusters based on features or roles. Popular methods for role identification are RolX and RoleSim. These are used to identify the relation between nodes based on the similarity of the attribute on which they are connected. These completely depend on the degree of the node and ignore the attribute of the nodes. Other methods that help with meaningful observations when analyzing graphs are regular equivalence, or some basic measure-based methods such as PageRank, betweenness, or closeness.

Since the IMDb data set has millions of nodes and attributes, a subset of the data set is taken for better visualization, ease of calculations and explaining results. This project explores the data in a detailed way to obtain as much insights as possible.

The rest of the report is organized as follows, Section 2 explains the core approach RoleSim, Section 3 explains the additional approach RolX algorithm, Section 4 and 5 explain the observed roles from RoleSim and RolX respectively, Section 6 compares the result between the core and additional approach followed by the conclusion and future scope in Section 7 and Section 8 respectively.

2 RoleSim

2.1 Motivation

RoleSim [3] is a converging algorithm that attempts to determine the similarity values between nodes in a graph network, based on the connections between their neighbours. The problem of graph automorphism is considered to be an NP-hard problem (not formally proven to be), with no known polynomial solution, and thus previous approaches to the problem of role discovery have proven to be exponentially computationally expensive, so as to be prohibitive in large networks. Using a converging algorithm allows the algorithm to determine role similarities much faster, with a certain window for error. The converging algorithm proposed by the authors of RoleSim, generates an array of similarity values for every combination of 2 nodes in the graph matrix. The algorithm has proven to be faster than any approach proposed before.

2.2 Description

RoleSim is an iterative approach that calculates a similarity value between any 2 nodes of a graph network using a formula that primarily uses the maximal matching weight between the neighbours of the 2 nodes. The maximal matching is based on a subgraph, that is built as such that all nodes in N_u (neighbours of u) are connected to all the nodes in N_v (neighbours of v) with edges that have weights equal to the values of the similarity obtained from the previous iteration of the algorithm. The initialization process can be achieved with several approaches, but the one that we chose is the same one as the one proposed in the modification of the RoleSim algorithm - the Iceberg approach.

Through the Iceberg initialization algorithm, the authors propose a pruning algorithm, filtering nodes of the graph, based on whether they meet certain thresholds and rules, as defined. This leads to a reduction of the size that is orders of

magnitude smaller than the initial graph, and allows quicker convergence of the influential nodes. The Iceberg algorithm thus suggests the creation of a new hash-table format in which we put a value of the similarity calculated by an empirical formula given by the authors.

After the Iceberg initialization, we proceed with the normal iterative RoleSim algorithm that converges similarity values to a range of values from 0 to 1, between all the nodes that been selected after pruning, which gives us a bisimilarity matrix, in which every element is a similarity value between the nodes indicated by the indices. We then perform node-preprocessing by determining a certain number of roles and creating a histogram of nodes. This is then visualized and analyzed for the purpose of role discovery.

2.3 Implementation

We decided to use our own implementation of RoleSim, having found no satisfactory alternative openly available source code. The `networkx` graph library was predominantly used to store all the intermediate structures, inputs and output similarity graphs. Since RoleSim primarily works with degrees of nodes and not any features of the nodes themselves, the iterative algorithm uses graph structures instead of matrices to store the similarity values, in order to optimally save space. There are 4 hyperparameters to be set during the process of RoleSim, in both the Iceberg initialization step and the iterative algorithm itself, which are α (initialization parameter), β (decay rate), θ (pruning threshold) and δ (convergence parameter), and many trials were performed to determine the best configuration that would allow us the quickest and best convergence.

After obtaining a suitable similarity graph, role discovery is done by creating histogram bins and performing a binning operation using the digitise function of the `numpy` library. This allows us to determine individual roles for the selected nodes, and also provides statistics about the roles themselves. Visualization is then done using the `Plotly` library.

The pseudocode we used for this process was directly taken from the paper proposing the algorithm, and is presented below in Algorithm 1.

Algorithm 1 IcebergRoleSim($G(V, E)$, θ, β, α)

$H \leftarrow$ empty hash table indexed by node-pair ID (u, v);
$d(v) \leftarrow$ degree of v;
Sort vertices V by degree;
for each $v \in V$ **do**
  $D^v = \{d_1^v, d_2^v, \cdots, d_{d(v)}^v\} \leftarrow$ degrees of neighbors of v, sorted by increasing order;
end for
for each $u \in V$ **do**
  for each $v \in V, \theta' d(u) \leq d(v) \leq d(u)$ (Rule 1) **do**
    $m_{11} \leftarrow (1-\beta)\frac{min(d_1^u, d_1^v)}{max(d_1^u, d_1^v)} + \beta$;
    if $d_1^v \leq d_1^u$ and $N_v - 1 + M_{11} < \theta' N_u$ **then**
      Skip to the next v; (Rule 3)
    end if
    Compute maximal matching weight $w(\mathcal{M})$;
    if $w(\mathcal{M}) \geq \theta' d(u)$ (Rule 2) **then**
      Insert $H(u, v) \leftarrow (1-\beta)w(\mathcal{M})/d(u) + \beta$;
    end if
  end for
end for
Perform iterative RoleSim on H. For neighbor pairs $\notin H$, use $\tilde{R}(x, y) = \alpha(1-\beta)N_x/N_y + \beta$

3 RolX

3.1 Description

RolX [1] is a feature-based clustering method. It contains two main steps. First, it extracts the feature matrix from the networks by using ReFeX [2]. The next step is using non-negative matrix factorization to cluster and assign roles to each node. In this project, we used an existing implementation of RolX with C++ in Stanford Network Analysis Platform (SNAP)[1].

We also implemented a visualization program in Python with the Plotly[2] library to exhibit the actual results of the clustered roles. An example visualization can be seen in Figure 2 and can also be further explored as an interactive graph on https://plot.ly/~198684x/6/network-graph-made-with-python/. The details of the construction of this program are not included in this report.

3.2 Implementation

In the implementation, the are three important methods. They are feature extraction, non-negative matrix factorization, and role selection. The pseudo code is given in Algorithm 2.

The algorithm makes the number of edges within the egonet to be one column of features, and the number of edges entering and leaving the egonet to be another column of features. Once a feature is pruned, the algorithm does not use it to generate new features in the next iteration. This is a time-efficient choice. A conservative method is to use all features to generate the next round features. There exists cases where a pruned feature can generate new recursive features which are not correlated with any old features. In the algorithm, these useful features will never be generate. So it might omit some useful features. However, using all features to generate next round features will cause exponential increasing complexity. In a large scale graph, this is unacceptable.

The complexity of the algorithm is determined by the methods and we know that it is liner on the number of edges [1]. Let n be the number of nodes in the network, m be the number of edges, f be the number of features, and r be the number of roles. The feature extraction has complexity $O(f(m+nf))$ [1]. The non-negative matrix factorization has $O(nf)$. And role selection has $O(nfr)$. The algorithm has complexity $O(mf+nfr)$ [1]. The running time for 500 nodes with 662 edges and 10 roles is less than 3 minutes.

The algorithm can give the exact role assignment. In the role selection, the matrix has the same number of columns as

[1] https://github.com/snap-stanford/snap
[2] https://plot.ly/python/getting-started/

the roles and it selects the column with the max value in the row as the role for the node. Therefore, the algorithm always gives the exact role of each node.

4 Roles from RoleSim

Two subsets of data were considered from different parts of our original data set, since the vast size of the data set (millions of nodes) would be computationally impossible to achieve in Python on a normal laptop. The first data set was the set of movies from 2000-2001 (approx. 5000 nodes), that was chosen as a representative sample set of movies with no further reduction or filtering. Upon filtering through the Iceberg algorithm, the top 665 influential nodes were identified, and moved forward to the iterative RoleSim algorithm. This resulted in a set of roles being calculated, resulting in the graph that can be seen in Figure 1 and in the site https://plot.ly/~198684x/8/network-graph-made-with-python/.

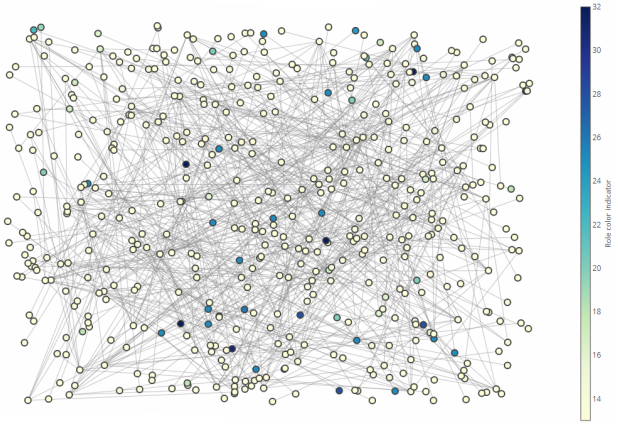


Fig. 1: Visualization of a subset of nodes, their connections, and a color coding corresponding to their role

Statistics of the roles are described in Table I. Not all roles are displayed in this table since setting a higher number of roles allows finer role-separation in the histogram. While setting many different numbers of bins in the algorithms, we found that higher granularity tended to coalesce our roles into a single role.

Table I: Amount of nodes per role

Role	13	14	15	16	17
Nodes	532	65	11	8	6
Role	18	19	25	28	32
Nodes	4	2	19	3	5

The second data set that was considered were the first 10,000 nodes from the Movie to Actor data set, and many parameters were attempted to prune out nodes from the algorithm, in order to achieve a usable processing time for the RoleSim operation. However, in spite of setting the threshold to extreme values, it was impossible to find a pruned sub-graph that had less than 5000 nodes, suggesting that any further processing would remove vital information about similarities from the network. With such a large sub-graph, processing of

Algorithm 2 RolX

```
G ← input network
procedure ExtratFeature(G)
    F ← empty features
    procedure AddNeighborhoodFeatures(G, F)
        procedure AddLocalFeatures(G, F)
            for each node in G do
                F ← add IN degree
        procedure AddEgonetFeatures(G, F)
            for each node in G do
                F ← add numbers of edges within egonet
                F ← add numbers of edges entering and
                      leavingegonet
        procedure AddRecursiveFeatures(G, F)
            t ← 0, similarity threshold
            rF ← F, retained features
            while size of rF > 0 do
                newF ← recursive features from rF
                rF ← pruned features using t, newF
                F ← add rF
                t ← t + 1
    return F

v ← convert F to matrix
r ← input number of roles
w, h ← empty matrix
t ← threshold value

procedure NonNegativeMatrixFactorize(
            v, r, w, h, t)
    cost ← 100
    newCost ← 0
    row ← number of nodes in v
    col ← number of features in v
    w ← random matrix with row, r
    h ← random matrix with r, col

    while |(newCost − cost)/cost| > t do
        cost ← newCost
        newCost ← 0
        update newCost by calculating v
        update w by calculating v
        update h by calculating v
    return w, h

procedure FindRoles(w)
    R ← empty sets
    m ← min value
    for each row in w do
        role ← −1
        for each column in w do
            if cell value > m then
                m ← cell value
                role ← index of column
        R ← add node and role
    return R
```

the nodes took extremely long, in the order of hours per epoch, and we did not have the memory to proceed.

The processing of the first sub-graph took nearly 11 minutes per iteration, and it ran for approximately 40 epochs before converging to our final similarity values.

In both the graphs, the Iceberg pruning step was not significantly long, and ran for approximately 5 to 20 minutes.

5 Roles from RolX

5.1 Data Preparation

When trying to extract roles with RolX, the first 10,000 nodes were selected from the Movie to Actor data set. The resulting network contains 13,119 edges. Because the Movie to Actor network was used, this subset of the network contains nodes that have both the movie and the person types. For the extraction of roles we decided to separate the movies into 10 different roles. Calculating this took about 46 minutes for the algorithm to finish and to assign roles for all the different nodes. The node ID and the number of roles per group are shown in Table II. A visualization of the a subset of the nodes can be seen in Figure 2.

Table II: Amount of nodes per role

Role	0	1	2	3	4
Nodes	747	1742	306	178	209
Role	5	6	7	8	9
Nodes	3871	446	1372	910	219

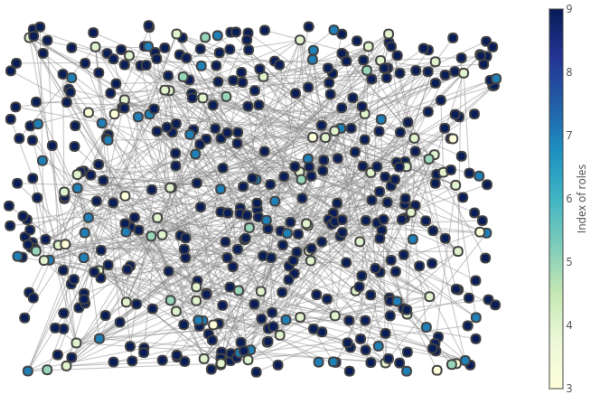


Fig. 2: Visualization of a subset of nodes, their connections, and a color coding corresponding to their role

5.2 Interpretation

The roles defined by RolX will be manually looked at to verify the performance. A few roles will be highlighted in this section.

5.2.1 Role 2: This role contains a lot of action and crime movies that originate in China/ Hong Kong. There are some movies in there which do not correspond to the role, but the vast majority belong to movies with a Chinese background. An example list of of movies from this role can be seen in Table III.

Table III: Movie titles and genres from role 2, where the majority of movies come from China

Movie Title	Genre
Long teng si hai	Action,Comedy,Crime
Nu er dang zi qiang	Action,Crime,Drama
Wong Kok cha 'fit' yan	Action
Jin bang ti ming	Crime
Bok geuk cha lou	\N
Wu ting	Action,Crime
Mongkok Story	Action,Drama
Jing hua rou bo jiang jian dang	Crime

5.2.2 Role 6: This grouping of movie first looks like a grouping of Chinese gongfu movies, but looking further into the list, most of the titles are in some form of Spanish. Looking at the origin of theses movies shows that they originate from Mexico. Some of these titles can be seen in Table IV.

Table IV: Movie titles and genres from role 6, where the majority of movies come from Mexico

Movie Title	Genre
El investigador Capulina	Comedy, Crime, Mystery
El guía de las turistas	Adventure, Comedy
El circo de Capulina	Adventure, Comedy, Family
Dios los cría	Adventure, Comedy, Romance
El ganador	Action, Drama
El sátiro,	Comedy
Satanico Pandemonium	Horror, Mystery, Thriller
El compadre más padre	Comedy
De Cocula es el mariachi	Comedy, Drama, Music

5.2.3 Role 7: One of the more interesting roles is role 7. In Section 5.1 it was mentioned that the data set contains movies and actors. However, most roles only contain movies, sporadically containing an actor or director. This is not the case for role 7, here it is the other way around. Almost all of the items that where grouped into role 7 are people nodes. This means that the way people make connections to movies is different from the way that movies are linked to people and they can therefore be separated from the other nodes as a separate group. An example list of some of these people is given in Table V.

Table V: Names and work from role 7, where the majority is people

Sarah Miles	Actress
Eduardo Bea	Actor
Stelvio Cipriani	Composer
Pedro del Rey	Editor
Robert Redford	Producer, Actor
Joel L. Freedman	Producer, Director

5.2.4 The rest: Role 0 contains a combination of Indian Bollywood movies, followed by French drama movies. Then in role 1, 3 and 4 it looks like a random combination of movies from a lot of different places. Role 5 is the biggest group of the 10. It mostly contains American/ English movies but is also filled with some movies originating from China, India and the south of Europe. Role 8 again did not make much sense when looking for roles. Role 9 is mostly adult movies.

6 Comparison

By comparing the two methods, it is obvious that RolX has less computation complexity and better running time. Unlike RoleSim, it does not create the entire nodes by nodes matrix. Instead, it uses the given number of roles to create and calculate the feature matrix. With small number of roles, it is very efficient for large networks. But the weakness is also obvious. It is hard to determine what number of roles is appropriate for the network. If the number was too small, the algorithm might not be able to discover all the different roles. On the other hands, if the number was too large, despite the running time, the roles might loss generality and the nodes were clustered into too many small groups.

7 Conclusions and Recommendations

In this project, we have implemented and applied two methods, namely RoleSim and RolX, to extract possible roles from the IMDb movie data set. The roles are successfully extracted by both methods with different interpretation. This is reasonable since the feature matrices for the role assignment in both the methods are different. In the visualization, the differences in role detection become visible by showing the same nodes and positions but with the different extract roles. The roles extracted by the algorithms do bear some meaning, showing that there are indeed roles existing within the network. Based on the methods we have studied, it is obvious that a good method to extract roles is essential for the social network analysis. Since the roles are present in the network, it is important to identify and study them in order to have a better understanding of the whole network.

In terms of operation, RoleSim is a far less less optimal algorithm, since it computes a similarity graph between every single node, having an exponential effect on the matrix size. This is detrimental to the efficiency, as we discovered in further papers, where it has been observed that more than 72 hours were required to run the algorithm on a larger data set. This constrained us to sub-graphs which may not capture the full complexity of the similarities. We attempted multiple computational methods t

8 Further Investigation

For RoleSim, the biggest problem is the running time and the memory usage. Since it creates the matrix by nodes number. It is interesting to explore in future research whether it is possible to modify the algorithm in such a way that it can have a better and faster performance. Parallelization on GPU's might help reduce the time taken, since millions of operations are involved in the calculation of each iteration, and it expands exponentially. Different role-identification algorithms might help in identifying a better bin for each node, as an alternative to the histogram approach.

For RolX, the choice of roles is an interesting direction to explore. Instead of giving the number of roles, it might be possible to let the algorithm determine the best number of roles. And if that is the case, what would be the underlying principle and how could this be implemented. More research and investigation surrounding this topic could be done in future.

References

[1] K. Henderson et al., *Rolx: structural role extraction & mining in large graphs*. Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2012.

[2] K. Henderson et al., *It's who you know: graph mining using recursive structural features*. Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2011.

[3] R. Jin, V. E. Lee, and H. Hong, *Axiomatic Ranking of Network Role Similarity*. Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2011.

Appendix A
Poster

IMPLEMENTATION OF
SOCIAL NETWORK ANALYSIS ON IMDB
TEAM2

NATARAJAN CHIDAMBARAM, JONNE DEURLOO, QIANG FANG AND ABHISHEK MAHADEVAN RAJU

DESCRIPTION

Introduction

The IMDb dataset contains a variety of data on movies such as actors, directors, co-actors etc.

From this data, we mine the principal actors that are in the cast of multiple movies, and identify the actors who have multiple movies in common.

Social Network1: Movie to Actor
Social Network2: Actor to Actor
Social Network3: Movie to Movie

Description

The dataset is filtered to reduce the total number of nodes and edges.

Movies released after 1975 and average rating > 5.0
Categories Archive Footage and Archive sound are dropped

Social Network1: 687,181 nodes, 1,227,536 edges
Social Network2: 545,880 nodes and 5,053,843 edges
Social Network3: 141,301 nodes and 5,071,004 edges

CORE APPROACH

RoleSim

- Using an iterative approach to converge to a solution for the graph automorphism problem
- Using Iceberg RoleSim algorithm to initialize similarity matrix, along with pruning nodes.

```
(3418, 3418)
[[0.    1.    0.235 ... 0.64  0.64  0.64 ]
 [1.    0.    0.235 ... 0.64  0.64  0.64 ]
 [0.235 0.235 0.    ... 0.46  0.46  0.46 ]
 ...
 [0.64  0.64  0.46  ... 0.    1.    1.   ]
 [0.64  0.64  0.46  ... 1.    0.    0.19 ]
 [0.64  0.64  0.46  ... 1.    0.19  0.   ]]
```

Similarity matrix for 3418 nodes

Results

- Producing a similarity matrix for the nodes that haven't been pruned out by the previous step. (reduced 10000 to 3418 nodes)
- Used histogram for post-processing

Roles interpretation:

- Movies: Action movies predominant in a specific bin.

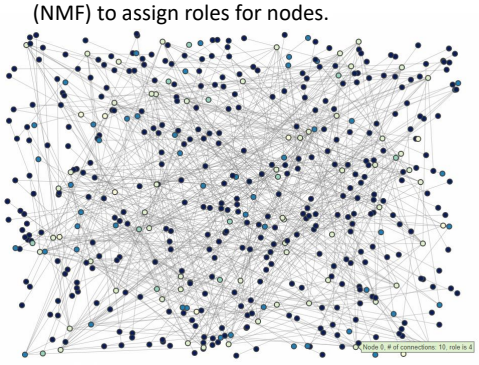
```
Truant Hero ---- The Last Shot
Killer from Above ---- Sang gong lau yin
Bloody Brotherhood ---- Ban ye xiao zi
Yin yang xie di zi ---- The Vengeful Beauty
Biao qi fei yang ---- The Eight Escorts
Run Papa Run ---- The Hunting Party
Lotte in Weimar ---- B. Monkey
Invincible Swordswoman ---- Jue dou zhe de sheng ming
The Magic Blade ---- Bruce, King of Kung Fu
Super Dragon ---- The Double Crossers
Super Dragon ---- Warriors
The Jade Fox ---- Born Invincible
Long feng zhi duo xing ---- Crash Pad
Jian ren shi jia ---- Failure to Launch
```

Action movies clustering in bin 2 after convergence

ADDITIONAL APPROACH

RolX

- Using Recursive Feature eXtraction (ReFex) to extract feature matrix.
- Using Non-negative Matrix Factorization (NMF) to assign roles for nodes.



Visualization with first 500 nodes

Results

Roles interpretation:

- Movies: clustering algorithm
- Person: clustering around movies

Role 7: Bollywood		Role 4: Chinese Action		Role 9: Adult	
Shakti	Action,Crime,Drama	Hua xin ye mei gui	\N	Neon Nights	Adult,Mystery
Mashaal	Action,Drama,Family	Fists of Fury 2	Action,Drama	800 Fantasy Lane	Adult,Comedy
Karma	Action,Adventure,Comedy	Tiaohui	Crime,Thriller	Amanda by Night	Adult,Crime,Drama
Kanoon Apna Apna	Action,Crime,Drama	Bruce Li the Invincible Chinato..	Action,Adventure	Ultra Flesh	Adult,Comedy,Sci-Fi
Saudagar	Action,Drama,Family	Po jie	Action,Drama	The Ecstasy Girls	Adult,Comedy,Drama
Mazdoor	Drama,Romance	She xing zui bu	\N	Society Affairs	Adult,Comedy,Crime
Dharm Adhikari	Action,Comedy,Drama	Cold Blooded Murder	Action	Unthinkable	Adult
Kranti	Action,Adventure,Drama	Chak wong ji wong	Comedy	Private Teacher	Adult,Comedy
Qila	Drama,Family,Music	Mercenaries from Hong Kong	Action,Drama	Crazy with the Heat	Adult
Duniya	Action,Drama	Shuang long tu zhu	Action,Comedy	Ecstasy Girls II	Adult
Bairaag	Drama,Family	Hak do fuk sing	Action	Indecent Exposure	Adult,Comedy
Faraar	Action,Crime,Drama	Bloody Brotherhood	Action	Showgirls	Adult
Hamare Tumhare	Family	Shao Lin sha jie	Action,Drama	Sex Play	Adult

Clusters around role 4, 7 and 9

FUTURE PLAN FOR COMPLETION

- Extend the implemented algorithm to other social networks to analyze links between them
- Try Fitting the algorithm on the full data collection rather than a sub-set

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

APPENDIX B
PREVIOUS REPORT

- See next page -

Social Network Analysis on IMDb

Preparation and Planning - Team2

Abhishek Mahadevan Raju *(1306162)*,
Jonne Deurloo *(1388797)*,
Natarajan Chidambaram *(1358111)*,
Qiang Fang *(0894902)*

***Abstract*—In this report, we introduce a publicly-available dataset- the IMDb dataset, and identify some social networks that we have extracted from it. We explain the 3 different social networks and provide some key statistics regarding them. The process that we followed during the construction of the social networks is then explained, and links are provided to access the social networks in the CSV format. Finally, we provide some preliminary hypothesis regarding the roles that we may be able to determine for the various nodes in the social network.**

1 Data Source

THE Internet Movie Database, better known as IMDb, is a large, online database with information regarding movies, TV series, actors, and much more related information. IMDb provides this information as raw data sets on their own website. Information about the data sets can be found on https://www.imdb.com/interfaces/ and the data sets themselves are hosted on https://datasets.imdbws.com/. As stated on their website, this complete data set is provided under non-commercial licensing and can therefore be used for educational purposes.

The information in this data set is distributed over seven different files.

1) *title.akas*: Contains information about movie aliases.
2) *title.basics*: Contains basic movie information such as name, year and genres.
3) *title.crew*: Contains information about the writers and directors of the movies.
4) *title.episode*: Contains information about specific TV show episodes.
5) *title.principals*: Contains information about the main cast and crew of a movie.
6) *title.ratings*: Contains the rating for items in the data set.
7) *name.basics*: Contains basic actor information such as name, birth year and professions.

From the data set, we find that there are approximately 33 million roles and 5.8 million different titles, starting from the year 1874. This is an enormous data set, consisting of millions of movies, short films, documentaries, TV series and video games. This means that we have to consider a smaller subset for computational ease. The constraints that we decided upon were movies that had been released after 1975, with an average rating greater than 5. It seems reasonable to assume that movies in this period might have a significant correlation and a largely connected social network, being a period of innovation for Hollywood.

2 Social Network

The final reduced data set that will be used in this project has a total of 687,181 nodes and 11,352,383 edges. We consider 3 social network formations from this data.

1) **MovieToPerson network** - The first network concentrates on the relation between the movies and the people associated with them.
   - There are 687,181 nodes and 1,227,536 edges.
   - These nodes contain 141,301 unique movies and 545,880 unique people.
   - Movie nodes describe the unique movies present in the data set and the people nodes describe the unique people present in the data set.
   - Movie nodes and people nodes have different attributes. Each node has its own identifier that can be used to identify it and its connections.
   - The movie nodes contribute to 20.5% of the total nodes, the majority are people nodes. The people associated with the movies are the people who played an important role in the movies like directors, producers, actors, actresses, music composers, writers, editors and some other categories.
   - There are 1,227,536 distinct edges and have, by default, a weight of 1.
   - This is mainly used for analysis of the connections between the movies and people and has the default attribute ”is connected” and has no other explicit property.
2) **PersonToPerson network** - The second network is a subset of the first network and focuses on the relationship between the people who contribute to various movies. Thus, in the first social network, if there are two people who have contributed to a single movie, an edge is made in the second social network between those two people with edge weight 1. If an edge already exists with some edge weight, then the weight is incremented by 1. Thus, in the new social network, the weight of an edge between two people corresponds to the total number of movies that are in common between them.
   - There are 545,880 nodes and 5,053,843 edges.
   - The number of distinct nodes is the same as the number of people nodes from the previous network.
   - The properties associated with these nodes are category, job, characters, primaryName, birthYear,

deathYear, primaryProfession, knownForTitles.
- The nodes have the following types, actors 25.95%, actress 15.93%, producer 12.07%, director 13.53%, self 8.53% and the remaining are music composers, writers, editor, cinematographer and other professions.
- There are 5,053,843 distinct edges with weights on them. The highest weight is 200 (the two people have worked together in 200 movies) and the lowest is 1. This highest weight corresponds to an Indian music composer duo **Laxmikant - Pyarelal**. According to Wikipedia, this duo has worked on nearly 800 movies together in a career spanning 35 years.
- All the edges have same property "is connected" with a weight and a unique identifier.

3) **MovieToMovie network** - The third network is similar to the second network and focuses instead on the relationship between the various movies based on the people that they have in common. In the first social network, if there are two movies that have a single person in common, then an edge is created in the new social network between the two movies with edge weight 1. If an edge already exists between those two movies with some edge weight, then the edge weight of this edge is incremented by 1. Thus, in the third social network, the weight of an edge between two movies is the number of people that they have in common.
   - There are 141,301 unique nodes and 5,071,004 edges.
   - The number of distinct nodes is the same as the number of movie nodes from the first network.
   - The properties associated with these nodes are titleType, primaryTitle, originalTitle, isAdult, startYear, endYear, runtimeMinutes, genres, averageRating, numVotes, ordering.
   - The nodes do not have a classifiable distinction, since they are all movie nodes with the same attributes.
   - The edges have a unique identifier and a weight. The maximum weight is 10 (10 people have worked together in two movies).

3 Construction

The data set is available from the IMDb website in the form of TSV files. It is a dynamic data set, meaning that the files get updated on a daily basis. We thus froze a single day of data, and will work with this during the analysis. The TSV files were imported into our workspace in the form of a pandas dataframe, using the tab character as the delimiter. The unique identifier of the titles in the data set is 'tconst'. We perform some initial filtering to create one large data set using a combination of all the data sets. This will enable simpler filtering and data access.

The rows of the table_basics data set are initially filtered based on the titleType to select only titles of type 'movie'. This significantly reduced the titles list to a sixth of the original size. We then use 'tconst' to perform a join with the title_ratings data set and the title_principals data set.

From this combination, we reached a data set with 21 columns and began filtering further. The categories 'archive_footage' and 'archive_sound' were dropped, as they did not seem to be nodes of people and also did not make up a significant share of all the principals. Further, the movies which did not have a startYear (release year) and which were released before 1975 were filtered out. At this point, we had a row size of 2 million, and decided to prune further by considering only the movies that received an average rating greater than 5. We then found some noise in the data set where some 'tconst'-'nconst' combinations were present multiple times, which we fixed by dropping duplicates on the combination.

To get the individual details of the people (principals), we performed a join on the name_basics data set, using the person (principal) identifier 'nconst'. Having reached a reasonable size of 1.2 million unique rows, this final table was exported in the pickle format to be used for our network construction.

All the above operations were performed using the **Pandas** library, and our final data set format was a dataframe with 1.2 million rows and 21 columns of data. The **pickle** library was used to import and export the dataframe for transferring.

We then subdivided the data set into three social networks to have a detailed view and study of the relations between nodes.

For the first network, constructing the nodes was simple - we found the unique 'tconst' and 'nconst' values and isolated their attributes. For the edges, every single combination of 'tconst' and 'nconst' in the network was a viable edge, and we extracted only these columns.

For the second and third networks (people and movie inter-relationships), the numbers of nodes were simply the unique people and movie nodes individually. For the edges, we used the **itertools** module to determine the possible combinations of size 2 between the nodes that were connected either a person or a movie node. This gave us the list of all the edges denoting all the inter-relationships between people or movies with a node in common.

Further analysis will be undertaken in the next part of the project.

4 Links

The generated CSV files can be downloaded from the following URL:

- **Movie to Person Network:** https://drive.google.com/file/d/15htQlA7dU4FJDMZB7nBvD_Mmj-3li3ad/view?usp=sharing
- **Person to Person Network:** https://drive.google.com/file/d/12wlfsiTm5KL9qWYeO7lMmR6ZJyfIgqEw/view?usp=sharing
- **Movie to Movie Network:** https://drive.google.com/file/d/1KtYZetgrDvP4u61y29VisnOEoneGZukH/view?usp=sharing

5 Hypotheses

The idea of different roles in our network is a bit tricky since the data sets already have a role such as actor, producer, or director for each person. By investigating the graph, the role hypotheses are based on the relations of the nodes and the movie's popularity and quality. The idea is to decide the influence of nodes for a movie. For instance, different actors or directors can be excellent, very good, good, mediocre, or bad for a movie. And a movie can be excellent, very good, good, mediocre, or bad by checking its actors and director. We will be analyzing the characteristics of a particular combination and to further analyze if a particular combination works out good all the time. This gives insight in getting the best combination and also helps in understanding the links and the connections that are established in the movie industry which could be used for prediction in future.

6 Conclusion

In the report, we introduced the IMDb data set, and explained the rationale behind constraining the data set to a reasonable size for computational ease and relevance. We then declared the various social networks that we will be using to analyze in the next part of the assignment, and provided some key statistics and interesting facts about the data set. We then discussed the procedure that we used for the data set, and declared the various libraries used to form the edges and nodes. Finally, we provided a hypothesis regarding the possible roles that we might observe from the role simulation algorithms.

Appendix A
Poster

PROPOSAL FOR THE IMPLEMENTATION OF

SOCIAL NETWORK ANALYSIS ON IMDB

TEAM2

NATARAJAN CHIDAMBARAM, JONNE DEURLOO, QIANG FANG AND ABHISHEK MAHADEVAN RAJU

DESCRIPTION

Introduction

For the social network analysis, we have chosen the well-established IMDb movie dataset. The IMDb dataset contains a variety of data regarding movies, short films, documentaries and other film-media.

From this data, we mine the principal actors that are in the cast of multiple movies, and identify the actors which have multiple movies in common, thus helping to identify the most connected actors.

Attributes of importance

MoviesBasics
titleID : Primary Key identifier
language : Language of the title
isAdult : 0 for non-adult; 1 for adult

MoviePrincipal
actorName : Identifier of primary actor
genres(array): Maximum 3 genres

Rating
rating : Weighted average of all ratings

Actors
actorName : Identifier for actor
BirthYear : YYYY
DeathYear : YYYY or '\N'
Primary profession : String array
Titles known for: String array

The data is in the format of TSV (Tab-separated-values) files, which are available at the following URLs:
https://www.imdb.com/interfaces

https://datasets.imdbws.com

MINED INFORMATION

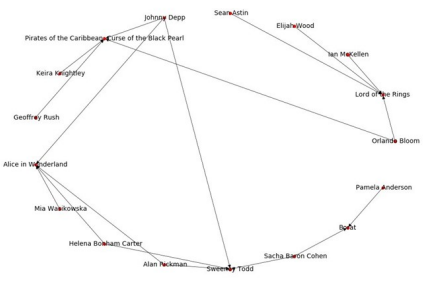
There are 33,745,716 roles in all kinds of movies, which number 5,872,369 unique entities. There are 10 categories of entities, from which we focus upon movies.
There are 518,264 movies in the IMDb database. The oldest is from 1874, a short film called 'Passage de Vanus'

In terms of actors and actresses, we have 13,745,628 of them throughout all the different entities! Just the movies have 1,649,199 actors and actresses in the entire history of film-making.

Kevin Bacon has become synonymous with graph networks, due to the fact that his prolific film career keeps him no further than 6 steps away from any actor or actress that ever worked in film history.

Kevin Bacon has played 482 roles across all the different entities, since he was born in 1958.
He has been a actor in 146 movies, and in the rest, he plays either himself, or undertakes other roles.

SAMPLE SUBGRAPH



This subgraph of our dataset shows 4 different movies:

- Pirates of the Caribbean
- Lord of the Rings
- Sweeney Todd
- Borat

-Here we can clearly see some overlapping actors in different movies
-These interactions will be interesting when analyzing roles in this dataset.

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY