A NETWORK TOUR OF DATA SCIENCE

A Netflix Tour of Data Science Film suggestion by diffusion on graphs

Dataset : Kaggle Dataset - Films and Crew

Students – Team 17:

AVIGNON Edwige
FOURCADE Pierre
NGUYEN Kenneth

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Introduction

Nowadays, most of the streaming platforms propose a targeted service that suggests you the films/series that you might enjoy according to the films/series that you have already watched. For instance Netflix proposes this kind of service and their proposals are often very accurate.
Imagine now that you are a modest student, you cannot afford those services but you do not want to spend your Saturday evening looking for a film instead of watching it. However, you have followed a course in Data Science and you wonder if you can finally get rid of this time loss by imagining a system that would target for you the films you would like.
One may think that the scenario, the performance of the actors and the depth of the story are important criteria to know if someone is going to enjoy the film or not... but then, realizing that "*Furious 7*" was ranked 7, "*Transformers: Dark of the Moon*" was ranked 20 in a list of highest-grossing films, we wondered if simpler criteria (such as the cast, the crew, the genre, the budget, the advertising...) were not more critical in the valuation of the film. The main idea that we are going to use here is that if you liked (resp. disliked) a film with a particular cast, crew, genre, you might also like (resp. dislike) a film presenting similar features.

To get an idea of how are the similarities between every films we can use graphs. Indeed when we are building graphs we are making strong, weak or no connections between nodes to represent how similar two nodes on the graph are. Those connections are the edges and the force of those connections are the weights. Thus we can build several adjacency matrices representing the similarities of the films (nodes) for different features. From here we want to analyse and predict the likings of someone called "user". A user can only inform us if he liked or not a film with a vote between 0 and 10 (10 being that he loved the film). Then to predict the likings of this user, our idea is to combine the graph with the signal representing the average of the vote given by all users, the signal representing our particular user votes. Our user has only seen a few films and his votes propagate to the closest neighbours using diffusion with a heat-kernel slightly modified to make sense in this subject. This would create a personalized signal for this user based on the average rating of the others (because it would takes too long for the users to watch enough films to have relevant proposals).

I - Adjacency

To diffuse vote on the graphs we need to create the graphs and their corresponding adjacency matrices.
With the different data we have with the Kaggle dataset for the films and the crews, we have selected 4 features to build 4 adjacency matrices: the cast, the first role (first actor mentioned in the cast), the genres and the crew.
Those matrices must represent correctly the similarities of the films according to the features we are considering. Building those matrices is essential and more precisely the definition of the weights is crucial. Indeed as our goal is to use diffusion to influence the similar films, if the measurement of similarity is not done with meaning the results will be pointless.

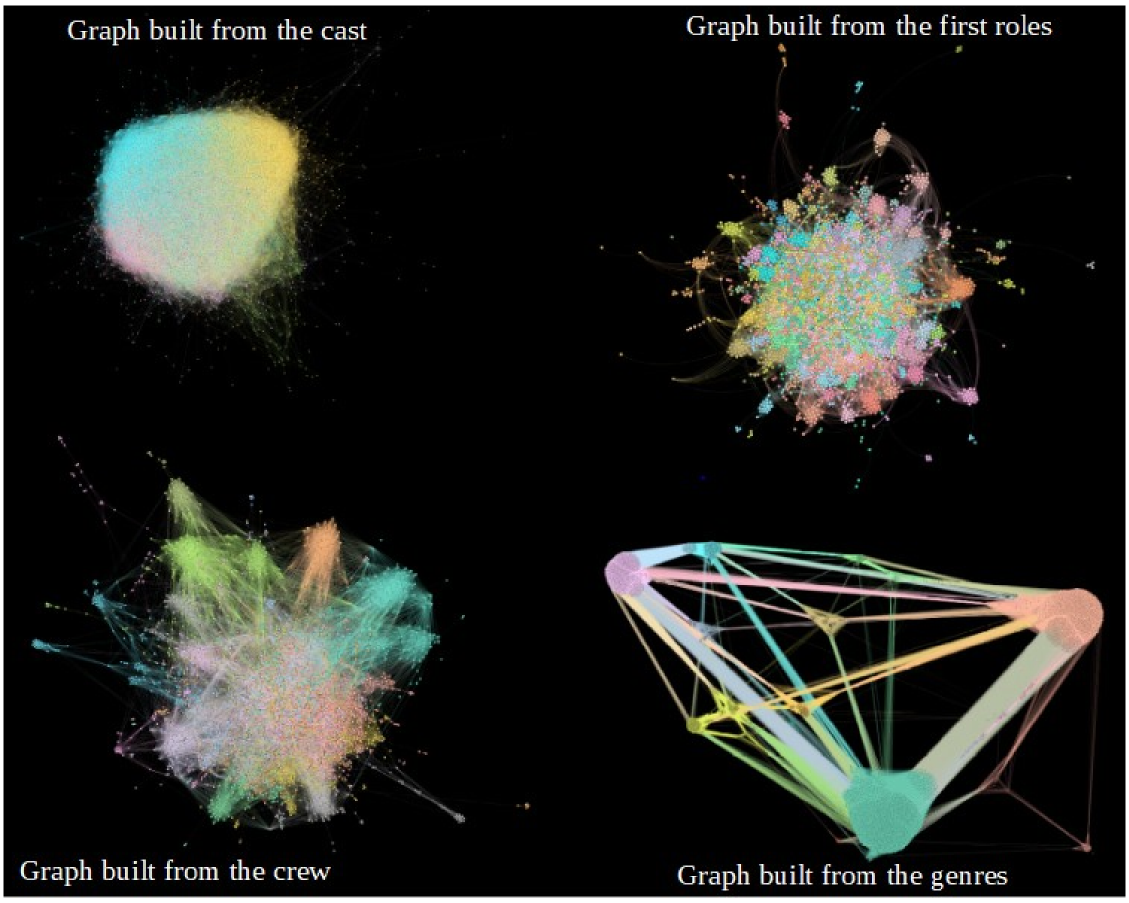
First of all, as we are not working with meaningful numerical values for the different features: id of the actors, id of the crew or id of the genres, the similarity between two films comes from a personal interpretation. In our case for the matrices, for each features, we have considered the number of elements in common and the position of the elements. Indeed, if we take the genres for

instance, they are mentioned in a specific order: the first genres mentioned correspond to the most emblematic genre of the film and then comes non-predominant genres.
The important point is to define levels of similarity. The idea of those levels is that if a node A is connected to two different nodes B and C, belonging to different levels of similarity in regards of the first node A, the gap between between B and C must be noticable.

Let’s take an example: the adjacency built with the genres.

- If the first genre of the film considered does not match the first or the second genre of the other film: the nodes are not connected
- If the first genre of the film considered matches the second genre of the other film and if the second genre of the film considered matches the first genre of the other film: unit weight
- If only the first genre of the two films match: 2 x (unit weight)
- If the first and the second genres match one another: 4 x (unit weight)
- If the two genre lists contain more than two genres and are strictly identical: 8 x (unit weight)

The rules taken to build the different matrices are different for each one but the core idea is the same. All the matrices built are undirected in our case.


Graph built from the cast
Graph built from the first roles
Graph built from the crew
Graph built from the genres
Graph of the four adjacency matrices represented with Gephi and coloured by modularity

We can see here that depending on the feature we have chosen and the rules we have taken the graphs can differ a lot: for instance we can see more easily some groups in the graphs made from

the first roles, the crew and especially from the graph made from the genres; whereas for the cast we have something more like a single giant component.
However thanks to the modularity computed by Gephi we can see in all the graphs some communities and confirm that the similarity between the nodes is varying.

<u>Note</u>: The graph made from the genres is actually reduced in this representation. Indeed we had too many edges for Gephi. Thus we have reduced the number of parallel edges. However it is only for representation, otherwise the full adjacency is used.

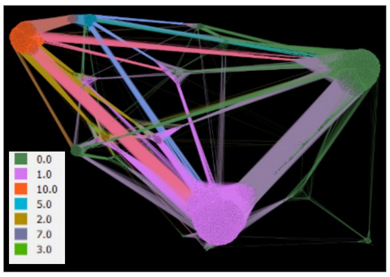
II – Adapted diffusion on the graph

To influence the other nodes and to make a graph specific to one user we want to use diffusion and more precisely the heat-kernel. We have built graphs with the goal to make similar films closed to each other. Then, we want to create a diffusion by closest neighbours, just like the heat diffusion.

To diffuse a single vote on a graph we have to consider a Dirac of this vote and then apply the heat-kernel to this Dirac (in the spectral domain). The problem is that we are considering a base signal: the average vote. This signal is the starting point, it is the notes we take into account until the user modifies them. It wouldn't be interesting to build a signal from the start in our case. But this means that we cannot just change the vote of one node on the signal and apply the heat-kernel: this would just diffuse every node and only make the graph more uniform, it would be meaningless.
As a result we need to take into account the diffusion by the heat-kernel of a single node in a base signal already present. For that we need to adapt the diffusion.

Let's take an example: the user rates 10 for a film. The node associated is considered as a heat source in the graph. Then we apply the following procedure:

- We create a Dirac for this node and diffuse it in the graph with the heat-kernel. Then we have a profile of how the node diffuses in the graph.
- We multiply the values of this diffused Dirac so that the highest value is equal to 10.
- To take into account the base signal and the diffused Dirac we create a new signal which is the mean terms by terms of the base signal and the diffused Dirac.
- However, in the case of films, placing a heat source cannot cool the other nodes (if we like a type of films it does not mean we will dislike the other types) so the new signal only take the mean if it is superior to the vote of the base signal.
- Finally, the votes of the user are fixed and cannot be changed unlike in classical heat diffusion where the heat source would have become colder as it gave its heat to its neighbours.



Diffusion of a Dirac 10 in the genre graph

If the vote is considered as a cold source then we take the same procedure but modified to cool down the graph.
This adapted diffusion can be used for all the possible votes the user can make if we consider correctly the type of source it represents.

III – Application and results

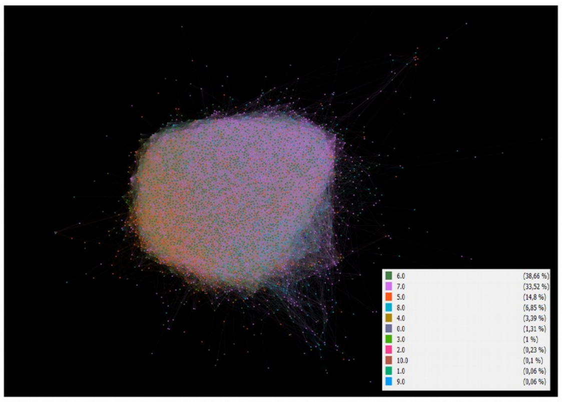
Now we are going to apply the adapted diffusion we have defined to our graphs. For convenience we have considered that the user is quite simple: he likes a film so he gives it 10 or he dislikes and gives a 0.

To test our model we have defined a series of films that a user has voted for. We have chosen to consider superheroes films as they represent a well defined kind of films. The cast and the crew can vary but the genres should be quite similar.

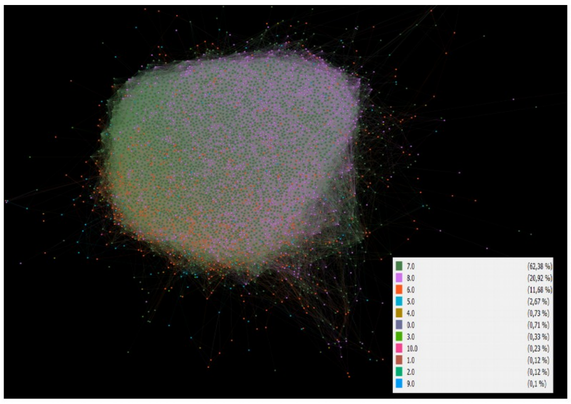
Then we have found 6 random superheroes films (quite different one another) that the user likes.

After the observation it, as expected, only shows an obvious change on the genres graph. We also have an influence on the other graphs but not as important. This seems logical as we have chosen a strong genre similarity in the films.

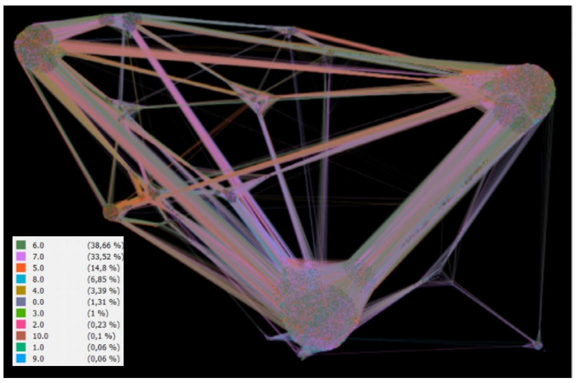
Let’s see two graphs with the base the signal and with the signal after taking into account the 6 votes of the user:



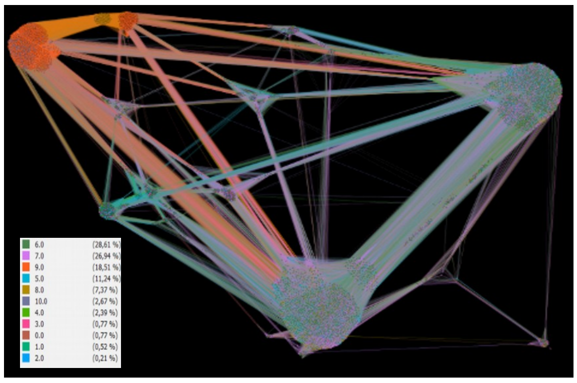
Graph built from the cast with the base signal (vote average)



Graph built from the cast with the signal taking into account the votes of the user



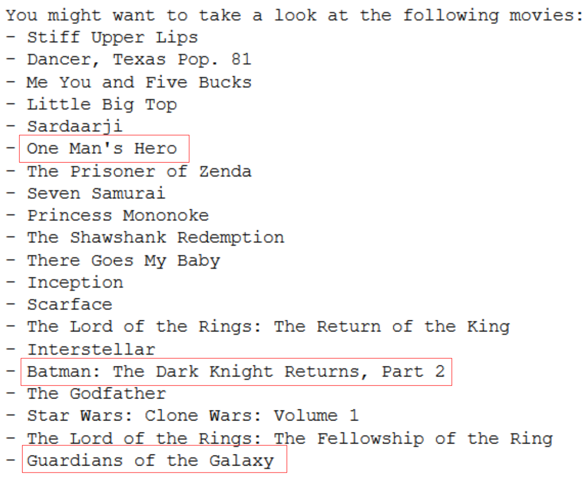
Graph built from the genres with the base signal (vote average)



Graph built from the genres with the signal taking into account the votes of the user

As we can see from the graphs our model has its influence and the influence doesn't seem to be illogical. We can clearly see that a well defined area of the graph (the graph from the genre more obviously) have been changed with our diffusion. As the user liked this type of films the films considered as similar got their vote increased to 9 if we look at the legend.

Then we have created a proposal function, which selects for us films that we might like according to our preferences. We give this function the signal combined from the four graphs: the mean terms by terms of the four signals.
Among the suggestions of this model we can still find the preferred films of the majority, which makes sense as we haven't placed a cold source to reduce their vote. But more importantly we can find some superheroes films.



```
You might want to take a look at the following movies:
- Stiff Upper Lips
- Dancer, Texas Pop. 81
- Me You and Five Bucks
- Little Big Top
- Sardaarji
- One Man's Hero
- The Prisoner of Zenda
- Seven Samurai
- Princess Mononoke
- The Shawshank Redemption
- There Goes My Baby
- Inception
- Scarface
- The Lord of the Rings: The Return of the King
- Interstellar
- Batman: The Dark Knight Returns, Part 2
- The Godfather
- Star Wars: Clone Wars: Volume 1
- The Lord of the Rings: The Fellowship of the Ring
- Guardians of the Galaxy
```

IV – Conclusion

However they are some parts of the process that needs to be clarified and completed. For instance it could be interesting to add other graphs built from new features. We could also search for other algorithms for our diffusion. We based it on the mean between a base signal and a diffused signal but there might be a more accurate way to see the influence of our vote in the base signal.
Moreover as we spoke about Netflix we could also try to had a collaborative filtering algorithm to this method to increase the number of vote to diffuse in a the graphs. It would take into account preferences of users that have the same taste as you.
As a conclusion for this project, it is possible to use diffusion on graphs to perform film suggestion.