

# Project B: Book Analysis

---

INFO-I 535 BIG DATA

Project B

Data Science Residential

Fall 2017

Indiana University

Bloomington, IN, USA

Jinju(Hellen) Jiang

2000276677

[heljiang@iu.edu](mailto:heljiang@iu.edu)

Nov 16, 2017

## Contents

1. Abstract .....	3
2. Introduction.....	3
2.1 Virtual Machine in Jetstream.....	3
3. Data pipeline.....	5
3.1 Data Pipeline Diagram .....	5
3.2 Running Data Pipeline .....	5
I. ETL .....	5
II. Process and clean data .....	6
III. Analysis .....	8
IV. Export.....	9
V. Visualization.....	9
4. Questions for previous work .....	12
5. Analyzing Other Texts(Jane Eyre by by Charlotte Brontë) .....	14
6. Further work.....	15
7. References.....	15


## 1. Abstract

this project I created a database of books and analyzed and visualized their text. By using natural language processing (NLP) to automatically discover the characters in a book, quantify the strength of relationship between them, find the important characters, identify groupings between them, and finally, created a visualization of the analyses..

## 2. Introduction

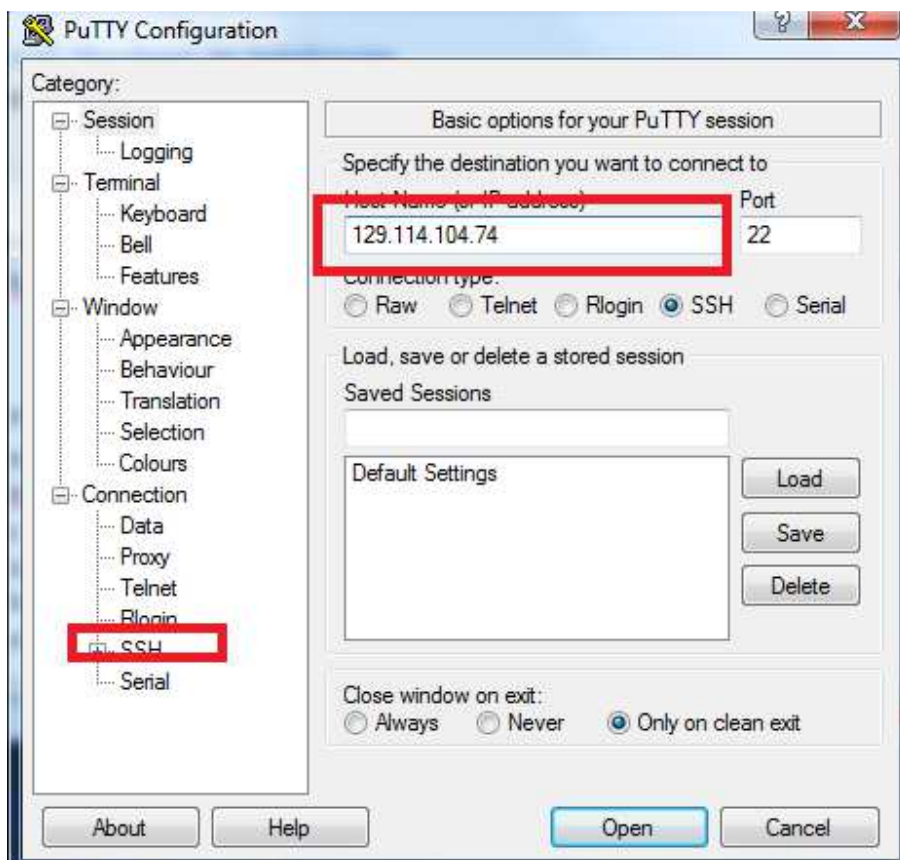
### 2.1 Virtual Machine in Jetstream

#### 1. Tiny Machine in Jetstream

 I535-I435-B669 Project A_hellen Jiang	<span style="color: green;">●</span> Active	N/A	129.114.17.25	M1.Tiny	Jetstream - TACC
---	---	-----	---------------	---------	------------------

#### 2. Setup VM

- ❖ Connect via PuTTY by typing IP address and upload private key. Then type username and password in jetstream.



- ❖ Setup directories and download code:

```
$ mkdir ~/Projects
$ cd ~/Projects
$ git clone https://github.com/dimitargnikolov/book-project.git
$ cd book-project
```

```
heljiang@js-104-74:~$ mkdir ~/Projects
heljiang@js-104-74:~$ cd ~/Projects
heljiang@js-104-74:~/Projects$ git clone https://github.com/dimitargnikolov/book-project.git
Cloning into 'book-project'...
remote: Counting objects: 219, done.
remote: Total 219 (delta 0), reused 0 (delta 0), pack-reused 219
Receiving objects: 100% (219/219), 1.94 MiB | 0 bytes/s, done.
Resolving deltas: 100% (76/76), done.
Checking connectivity... done.
heljiang@js-104-74:~/Projects$ cd book-project
heljiang@js-104-74:~/Projects/book-project$
```

Created a start-up script called .bashrc and installed pymongo, an extension to Python: Now the environment is ready.

```
heljiang@js-104-74: ~/Projects/book-project
python-35.pyc
creating build/bdist.linux-x86_64/egg/EGG-INFO
copying pymongo.egg-info/PKG-INFO -> build/bdist.linux-x86_64/egg/EGG-INFO
copying pymongo.egg-info/SOURCES.txt -> build/bdist.linux-x86_64/egg/EGG-INFO
copying pymongo.egg-info/dependency_links.txt -> build/bdist.linux-x86_64/egg/EGG-INFO
copying pymongo.egg-info/requires.txt -> build/bdist.linux-x86_64/egg/EGG-INFO
copying pymongo.egg-info/top_level.txt -> build/bdist.linux-x86_64/egg/EGG-INFO
writing build/bdist.linux-x86_64/egg/EGG-INFO/native_libs.txt
zip_safe flag not set; analyzing archive contents...
pymongo.__pycache__._cmessages.cpython-35: module references __file__
bson.__pycache__._cbson.cpython-35: module references __file__
creating dist
creating 'dist/pymongo-3.6.0.dev0-py3.5-linux-x86_64.egg' and adding 'build/bdist.linux-x86_64/egg' to it
removing 'build/bdist.linux-x86_64/egg' (and everything under it)
Processing pymongo-3.6.0.dev0-py3.5-linux-x86_64.egg
creating /opt/anaconda3/lib/python3.5/site-packages/pymongo-3.6.0.dev0-py3.5-linux-x86_64.egg
Extracting pymongo-3.6.0.dev0-py3.5-linux-x86_64.egg to /opt/anaconda3/lib/python3.5/site-packages
Adding pymongo 3.6.0.dev0 to easy-install.pth file

Installed /opt/anaconda3/lib/python3.5/site-packages/pymongo-3.6.0.dev0-py3.5-linux-x86_64.egg
Processing dependencies for pymongo==3.6.0.dev0
Finished processing dependencies for pymongo==3.6.0.dev0
heljiang@js-104-74:~/Downloads/pymongo$ cd ~/Projects/book-project
heljiang@js-104-74:~/Projects/book-project$
```

### 3. Data pipeline

#### 3.1 Data Pipeline Diagram

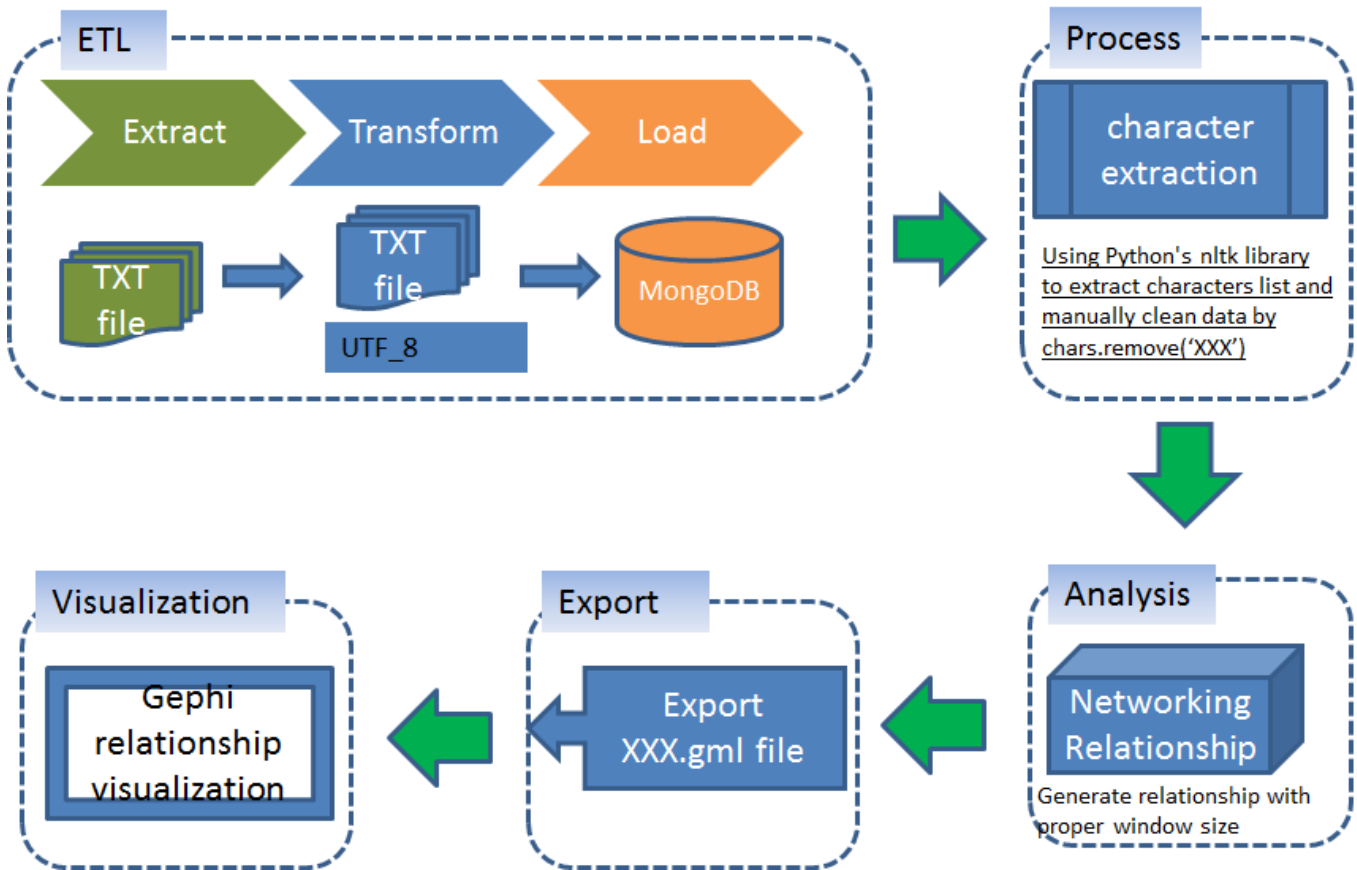


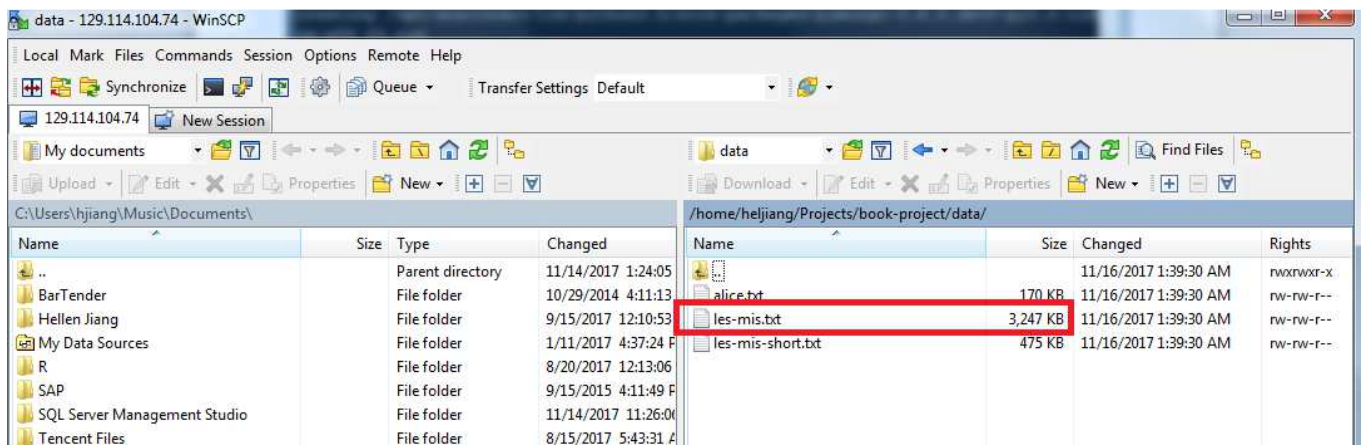
Figure 1: Data pipeline diagram

#### 3.2 Running Data Pipeline

##### I. ETL

**Step 1:** Download book and add book to MongoDB:

Downloaded book- Les Miserables from Project Gutenberg website (plain text UTF-8 format) into my local harddrive and then copy to project folder:



The other way is to download the book from website and save it to project folder by the following command:

```
$ wget https://www.gutenberg.org/files/135/135-0.txt (Links to an external site.)Links to an external site.-O
~/Projects/book-project/data/les-mis.txt
```

**Step 2:** Insert the content of book into MongoDB via pymongo:

```
heljiang@js-104-74:~/Projects/book-project$ cd ~/Projects/book-project
heljiang@js-104-74:~/Projects/book-project$ python
Python 3.5.2 |Anaconda 4.2.0 (64-bit)| (default, Jul 2 2016, 17:53:06)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pymongo
>>> from pymongo import MongoClient
>>> mongodb = MongoClient()
>>> db = mongodb.projectB
>>> with open('data/les-mis.txt', 'r') as f: text = f.read()
...
>>> db.books.insert({'author': 'Victor Hugo', 'title': 'Les Miserables', 'text':
text})
ObjectId('5a0c81f58bc181191213de99')
>>>
```

Now we can see book contents is in MongoDB:

```
>>> text[:500]
'\ufeffThe Project Gutenberg EBook of Les Misérables, by Victor Hugo\n\nThis eBook is for the
use of anyone anywhere at no cost and with almost\nno restrictions whatsoever. You may copy it
, give it away or re-use\nit under the terms of the Project Gutenberg License included with th
is\neBook or online at www.gutenberg.org\n\nTitle: Les Misérables\nComplete in Five V
olumes\n\nAuthor: Victor Hugo\nTranslator: Isabel F. Hapgood\nRelease Date: June 22, 2008
[EBook #135]\nLast Updated: January 18, 2016\n\nLangua'
```

## II. Process and clean data

**Step 1:** Extracting the Characters from a Book:



```
>>> import pymongo
>>> from pymongo import MongoClient
>>> mongoddb = MongoClient()
>>> db = mongoddb.projectB
>>> mongo_results = db.books.find({'title': 'Les Miserables'})
>>> from lib import *
>>> tagged_texts = tag_texts(mongo_results)
>>> chars = find_people(tagged_texts)
>>> chars
```

We can see a lot of people name:



```
heljiang@js-104-74: ~/Projects/book-project
eu', 'Dupuytren', 'Jesus', 'Flora', 'Aristotle', 'Philemon', 'Rio Maior Marques', 'Auvergene',
'Mutor', 'Les Marguerites', 'Reason', 'Time', 'Delphos', 'Alix', 'Pisa', 'Meaux', 'Legros', '
Sorsum', 'Mademoiselle Dog-lack-name', 'Râpée', 'Jacquerie', 'Jeanne', 'Hurry', 'Gaspard Bès',
'Varus Vibiscus', 'Flavius Josephus', 'Griserait Monsieur Orfila', 'Monmol', 'Mother Annoncia
tion', 'Enghien', 'Mademoiselle', 'Sabran', 'Inspector Javert', 'Feuilly', 'Venus', 'Madeira',
'Merci', 'Mademoiselle Lanoire', 'Théodule', 'Julia Alpinula', 'Bombarda', 'Farmer Mabeuf', '
Maître Corbeau', 'Paulin Musebois', 'Grace', 'Platon', 'Bacheux', 'Henri IV', 'Chaume', 'Genli
s', 'Curé', 'Foudre', 'Tyranny', 'Mousqueton Beauty', 'Miguel', 'Blank', 'Aller', 'Charles Nod
ier', 'Twilight', 'Jonah', 'Renard', 'Whither', 'Mirliton', 'Carthage', 'Little Gavroche', 'En
large Enlarge', 'Manon Lescaut', 'Plutarque', 'Tutu', 'Renée', 'Nîmois', 'Christmas', 'Richeli
eu', 'Zéphine', 'Lolotte', 'Martyrdom', 'Genappe', 'Shakspeare', 'Merlonus Horstius', 'Exodus'
, 'Bicêtre', 'Dom Luc', 'Van Kluze', 'Madam Magloire', 'Atala Avec', 'Charles IX', 'Angoulême'
, 'Vaugirard', 'Pope Eugene', 'Mr. Montparnasse', 'Labarre', 'Chevalier', 'Brevet', 'Fierce',
'Zelma', 'Whether', 'Passy', 'Lévi', 'Nous', 'Babet', 'Bootkick', 'Faton', 'Jean Bart', 'Ramus
', 'Stephen', 'Naviguer', 'Haxo', 'Pascal', 'Stupid', 'Virgil', 'Jean Valjean', 'Coligny', 'Ec
ce', 'Ezekiel', 'Jérôme', 'Pierre de Bruys', 'Prouvaire', 'Mercy', 'Jacobinism', 'Hugo', 'Mira
cles', 'George', 'Laffitte', 'Alava', 'Montals', 'Equal Workingmen', 'Lynch', 'Paul', 'Cambyse
', 'Folard', 'Monsieur Tranchelevent', 'Halles', 'Saint Sacrament', 'Listolier', 'Hugomons', '
Harmodius', 'Jausion', 'Bonaparte Scapin', 'Les Quadrins Historiques', 'David Widger LES', "Va
ljean's", 'Cain', 'Baccalauréat', 'Tartar', 'Campan', 'Coësre', 'Carry', 'Burtot', 'Opheltios'
, 'Sakoski', 'Bienvenu', 'Lacoste', 'Arostogeiton', 'Prince Aldobrandini', 'Bibere Tiberim', '
Marcos Obrégon', 'Tyre', 'Villon', 'Chloe', 'God', 'Clinton', 'Gula', 'Carrousel', 'Massieu',
'Danton', 'Europe France', 'Gaillon', 'Toulon', 'Gueulemer', 'Marseilles', 'Calais', 'Monsieur
Gillenormand', 'Sister Simplicie', 'Louis XVI.', 'Pépin', 'Gruchera', 'Baptistine', 'Mother A
scension', 'Salaberry', 'Nero', 'Blücher', 'Citizen', 'Petrarch', 'Pope Alexander VII', 'Henry
IV.', 'Himself', 'Dictées', 'Marsan', 'Patron Minette', 'Madame Pabourgeot', 'Avec', 'Letelli
er', 'Henri II', 'Master Scaufflaer', 'Papelotte', 'Casimir Périer', 'Monsieur Babet', 'Mademo
iselle Fauchelevent', 'Dædalus', 'Barge', 'Monsieur Mabeuf', 'Sabot', 'Jean Girin', 'Jacobin',
'Philip V.', 'Miles', 'Minerva', 'Madame de Léon', 'Lent', 'Sir Hudson Lowe', 'Treat', 'Volta
ire', 'Ben Lothian', 'Pan', 'Scotch Grays', 'Pollux', 'Poor Marius', 'Lord', 'Mr. Lawyer', 'Ma
gloire', 'Marcognet', 'Caracalla', 'Mirabelle', 'Martyrs', 'Foy', 'Delancey', 'Aigle', 'Alcipp
e', 'Mademoiselle Bigottini', 'Siam', 'Rue Cassette', 'Domon', 'Io', 'Marnix', 'Captain Louis
Hugo', 'Immense', 'Ditch', 'Somerset', 'Corinthe', 'Mademoiselle Cosette', 'Barague', 'Gourgau
d', 'Bernis', 'Argus', 'Je', 'Wilda', 'Tzar', 'Virgin Mary', 'Master Bourgaillard', 'Carignan'
, 'Cifuentes', 'Galileo', 'Desrues', 'Bavoux', 'Night', 'Providence', 'Lutetia'
>>>
```

## Step 2: clean data manually

The results are not perfect -- there are some characters that shouldn't be there like A, Madame, Paris and so on. This is in part due to the fact that the text we are working with is not entirely clean. In part, this is due to the entity-extraction algorithm not being perfect and getting fooled by non-traditional capitalization in the book.

This is another illustration of the need for setting up a data pipeline to clean your data before it is analyzed. In this case, we will clean the data manually: below is only some examples for the removing nodes. What I did is to copy all charts into excel file and then sorter and filter to find unlikely character names and then remove them in chars.

```
>>> chars.remove('A')
>>> chars.remove('Madame')
>>> chars.remove('Paris')
>>> chars.remove('English')
>>> chars.remove('Did')
>>> chars.remove('Roman')
>>> chars.remove('Sir')
>>> chars.remove('Son')
>>> chars.remove('Change')
>>> chars.remove('N')
>>> chars.remove('V')
>>> chars.remove('Doctor')
>>> chars.remove('Extra')
>>> chars.remove('Authority')
>>> chars.remove('Order')
>>> chars.remove('School')
>>> chars.remove('Which')
>>> chars.remove('Sad')
>>> chars.remove('Le')
>>> chars.remove('Follow')
>>> chars.remove('January')
```

### III. Analysis

Now that we have a list of characters we are interested in analyzing, we want to infer how closely related they are to each other. What we mean by that is that characters who appear in the same scenes or talk to each other often should be considered more closely related than characters who don't. We can represent the character relationships as a network where each node is a character, and each edge denotes the strength of relatedness between two nodes. Using a network representation like this is very powerful since it will later allow us to leverage a lot of knowledge about network analysis ([Links to an external site.](#))[Links to an external site.](#) and apply it directly to our problem in this project.

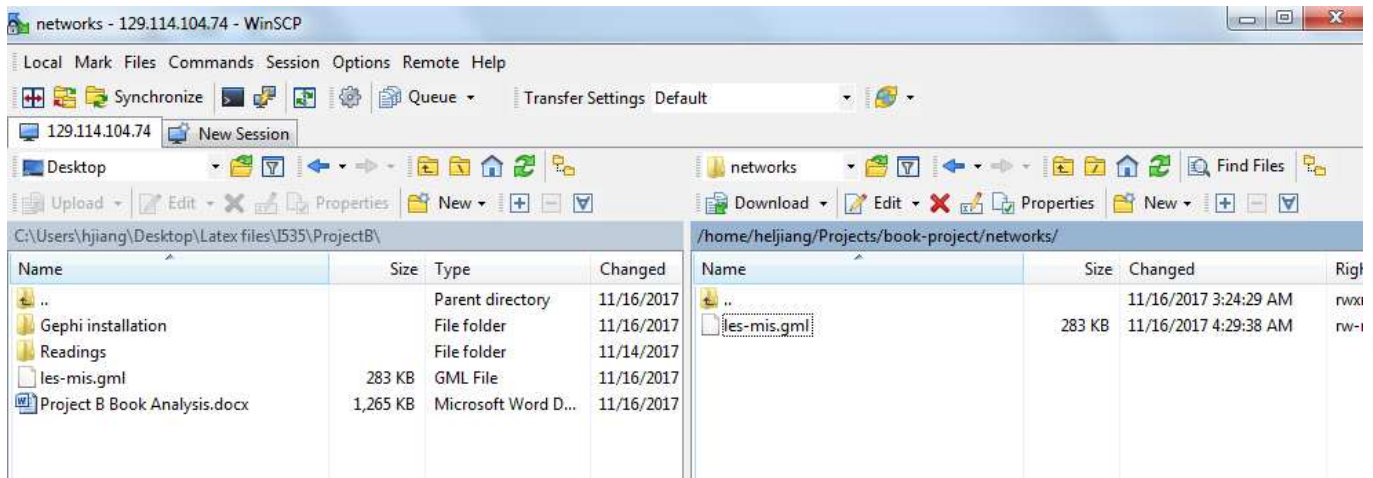
Now we use the available package lib.py to create relation nodes for all those characters.

```
>>> network = create_network(tagged_texts, chars, N=15)
>>> import networkx as nx
>>> os.makedirs('networks')
>>> nx.write_gml(network, os.path.join('networks', 'les-mis.gml'))
```

Now we can find there is one more folder-networks created and one more file for nodes relationship: copy it to local machine for further analysis and visualization.



## IV. Export



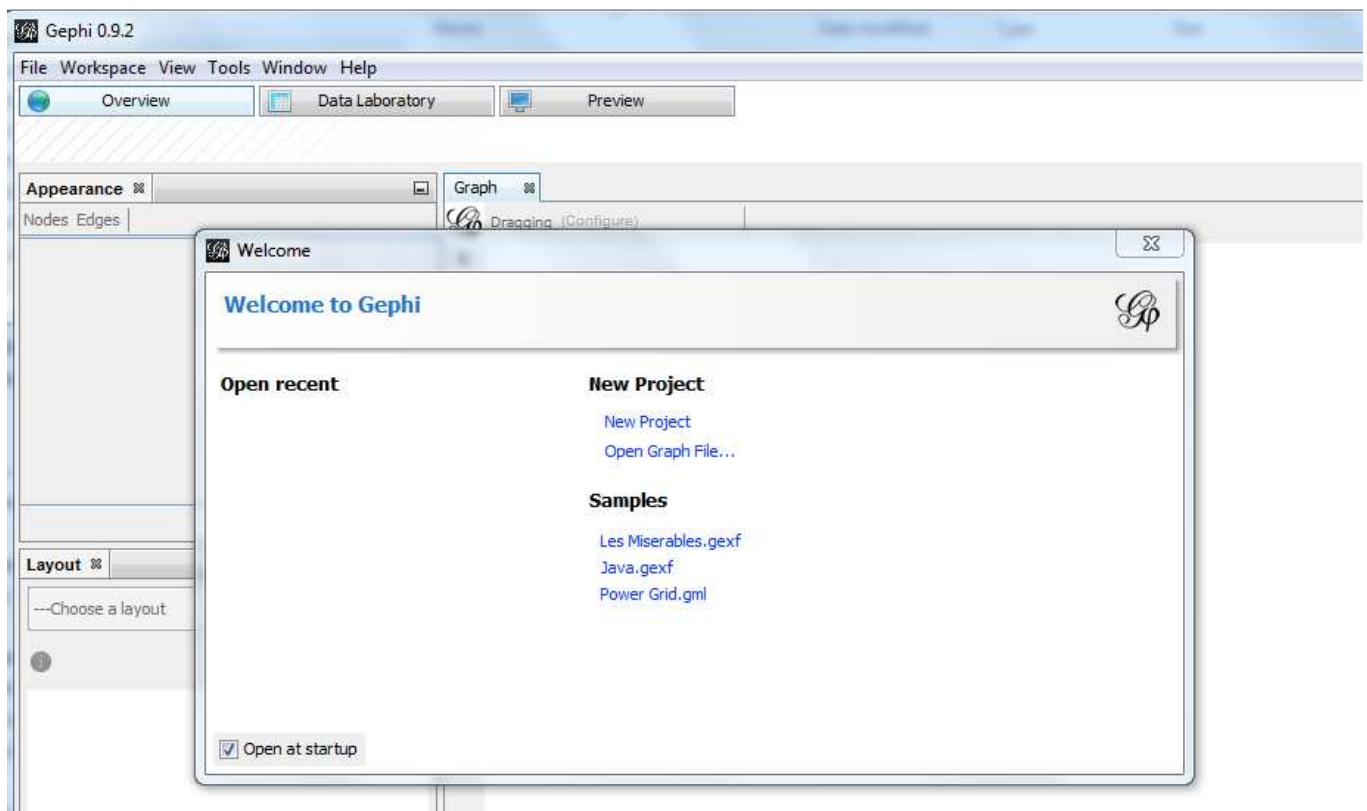
Here it is gml file looks like:



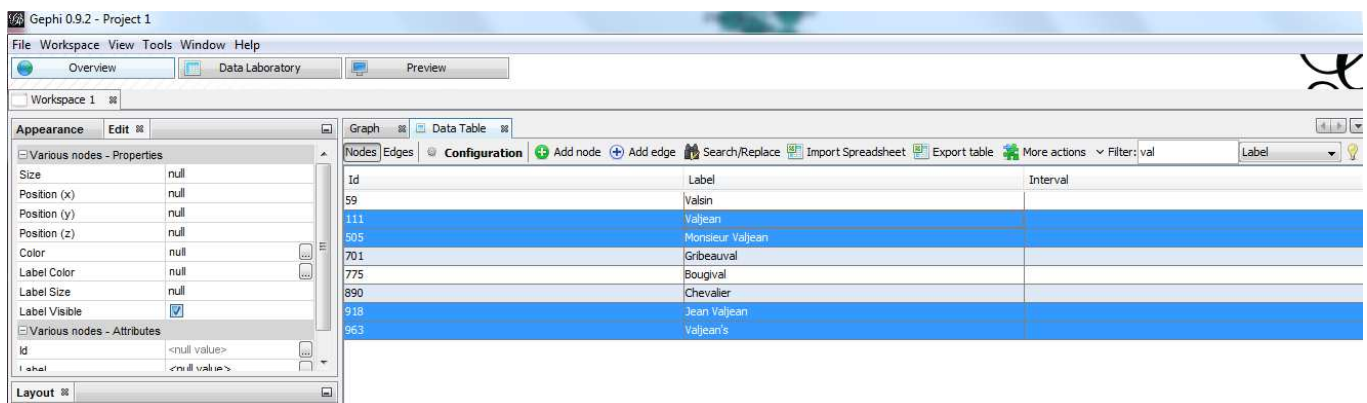
## V. Visualization

To analyze and visualize this network, we will use the Gephi graph analysis and visualization platform.

**Step1:** Install Gephi and JDE8.



**Step 2:** open gml file and open data laboratory to do further data clean: because there are some names that obviously are for same character. Here it is one example:



**Step 3:** Use statistics to calculate centrality and modularity, and use forceatlas 2 layout, got the following raw image:



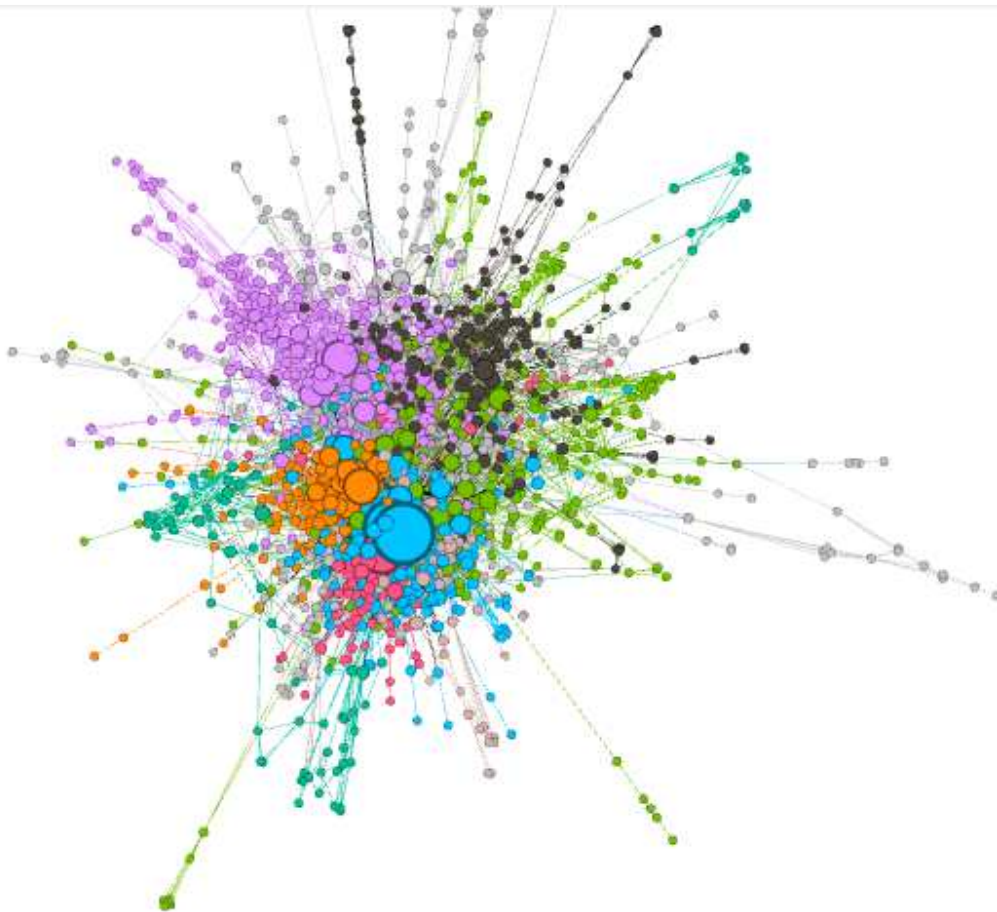
#### 4. Questions for previous work

1. Is a window of size 15 a good window size for the characters that you think are related? Why and why not?

Answer:

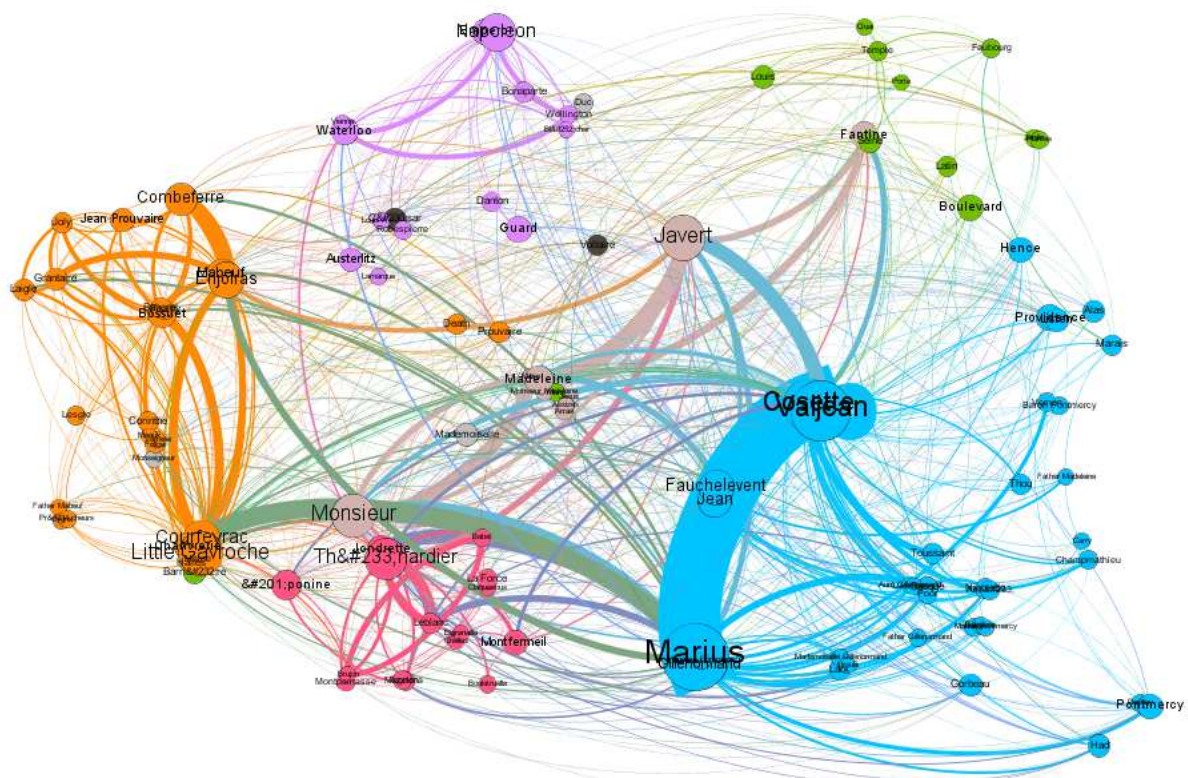
Yes, it is related, with 15 words I could create beautiful networking graph as shown above. And the average sentence length is 15-20 words, So it is make sense to set window size to 15.

I tried the networking with window size of 30(which was very long, and it took more time to create networking nodes, and it created more edges than 15 windows size). Here it is the raw relationship between them(window size for 30):



After improvement, we can find there are a lot of relationships among nodes: looks like the major are edges instead of nodes, which is not a good networking graph.





2. What are the strengths and weaknesses of a larger window size? Give an example of a relationship that was missed because of a window size of  $N=15$ .

Answer:

**Strengths:** larger windows size will scan more words, which will not miss relationship, especially for long sentence. If most of the sentences in the document is long sentence, for example, the average words for each sentences are 25 words, then it is better to setup window size to 25. If the window size is too small, it might miss some relationship. Here it is one example that the relationship that was missed because of a window size of  $N=15$  (from "Gone with the wind"): Scarlett's face did not change but her lips went white—like a person who has received a stunning blow without warning and who, in the first moments of shock, does not realize what has happened. So still was her face as she stared at Stuart that he, never analytic, took it for granted that she was merely surprised and very interested. The relationship between Scarlett and Stuart will be missed if the window size is 15.

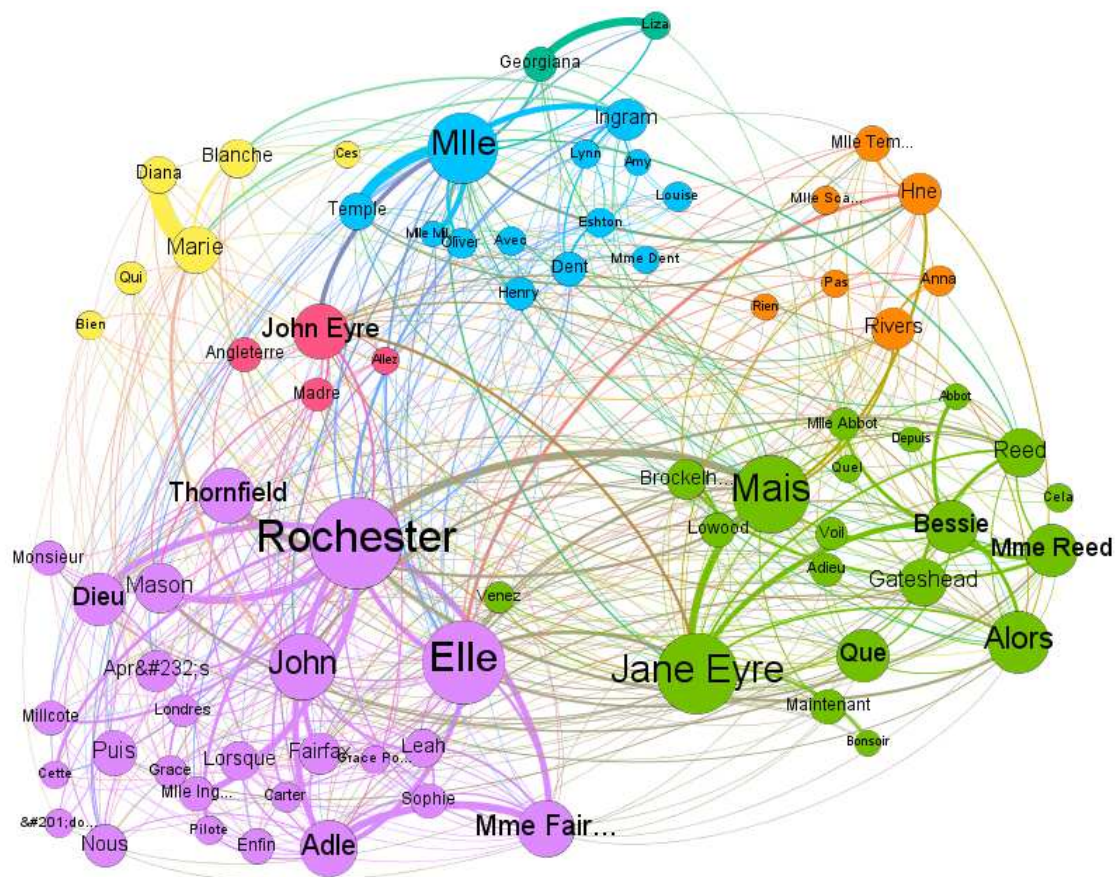
**Weakness:** The weakness is that it may create extra relationship which might do not exist. For example, it might cover 2 or 3 sentences to create relationship. For example: The professor Yun canceled the class today so I went back to home to do my homework. My sister Susan was very happy when she saw me back. If the window size is 30, then it will create relationship between professor Yun and my sister Susan, however actually there is no relationship between them. In this case the proper window size is 15.

## 5. Analyzing Other Texts(Jane Eyre by by Charlotte Brontë)

**Step 1:** Load book(Jane Eyre) into Mongoddb and extra chars, remove unlikely chars manually, and then create networking nodes with window size=15:

```
>>> import pymongo
>>> from pymongo import MongoClient
>>> mongoddb = MongoClient()
>>> db = mongoddb.projectB
>>> with open('data/JaneEyre.txt', 'r') as f: text = f.read()
>>> ...
>>> db.books.insert({'author': 'Charlotte Brontë', 'title': 'Jane Eyre', 'text':text})
ObjectId('5a0e0ecd5869c660dfef91')
>>> text[:200]
'\uffeffThe Project Gutenberg EBook of Jane Eyre, by Charlotte Brontë\n\nThis eBook is for the use of an
yone anywhere at no cost and with\nalmost no restrictions whatsoever. You may copy it, give it away or
\nre'
>>> mongo_results = db.books.find({'title': 'Jane Eyre'})
>>> from lib import *
>>> tagged_texts = tag_texts(mongo_results)
>>> chars = find_people(tagged_texts)
```

**Step 2:** Data visualization in Gephi



**Data visualization findings:** In my opinion, the most important character should be Jane Eyre. But from the relationship graphs, it looks like that Rochester is the most important character in this book.



## 6. Further work

The most time consuming part is data cleaning: remove noise chars, and then merge the nodes which are for the same characters(for example, Jane, Jane Eyre, Jeanne Eyre should be the same character Jane Eyre). The further work is to find easy and efficient way to clean data. And the other one is that how to process unrecognized letters, now I also corrected them manually. Need to find the automatic way.

## 7. References

- [1] Instruction from IU Canvas
- [2] <https://player.vimeo.com/video/9726202>
- [3] <https://www.gutenberg.org/>
- [4] Jane Eyre by Charlotte Brontë
- [5] <http://www.nltk.org/>