PIC16 Book Final Project Report

Renee Hsu, April Guo

Abstract:

This project reads a novel and creates a network of the characters in the story, and an importance ranking of the characters.

Algorithm Breakdown:

1st step: Import all Needed Files

Import the book and a text file named “firstname.txt” that contains all the common first names in English language (downloaded online).
Then, in the “firstname txt”, we used regular expression to find all the first names.

2nd step: get_human_name Function

The function “get_human_name” has functions as such:

1. Tokenize each word in the book, add pos_tag on them, and loop through every word with “PERSON” tag.
2. If the word is a full name (which means it has length longer than 1), it will be included in the character name list.
3. If the input word is just a person’s first name, it will be checked against the name file, and then it will be included in the character list if it’s a legit first name.

3rd step: Put Book in the Function

Get the character names in the book using get_human_name function.

4th step: Removal of the Inaccurate Names

We manually removed some of the names in the character list to make it more accurate.

5th step: Dictionary of the character names

The names will be stored in an ordered dictionary, with each key being the full name and the value being its corresponding first name, last name and full name. This is done to make sure that the computer will still recognize that person’s name even when only his/her first name or last name is mentioned in the book.

6th step: Matrix Set-up

We created a n*n matrix of 0’s with n being the number of characters in the list.

7th step: Put Values into the Matrix

We created a nested for loop to put the values into the matrix. The algorithm is as such:

1. Iterate through all the words in the book. If that word is in the values of the character dictionary (meaning if it’s a person’s name), update the position in the book that we’ve already searched, and the position of the “last character” found in the dictionary.
2. for the next person’s name found, check if that person’s name is different as the last character’s name and if they have a distance smaller than 50 words in the book. If these two conditions are satisfied, update the value in the matrix by adding 1 to those two people’s connections. This is done to make sure that the distance between any two characters is close enough to be considered as “connections”. The positions values are updated as well.
3. in the case where the two character have the same name, meaning the same

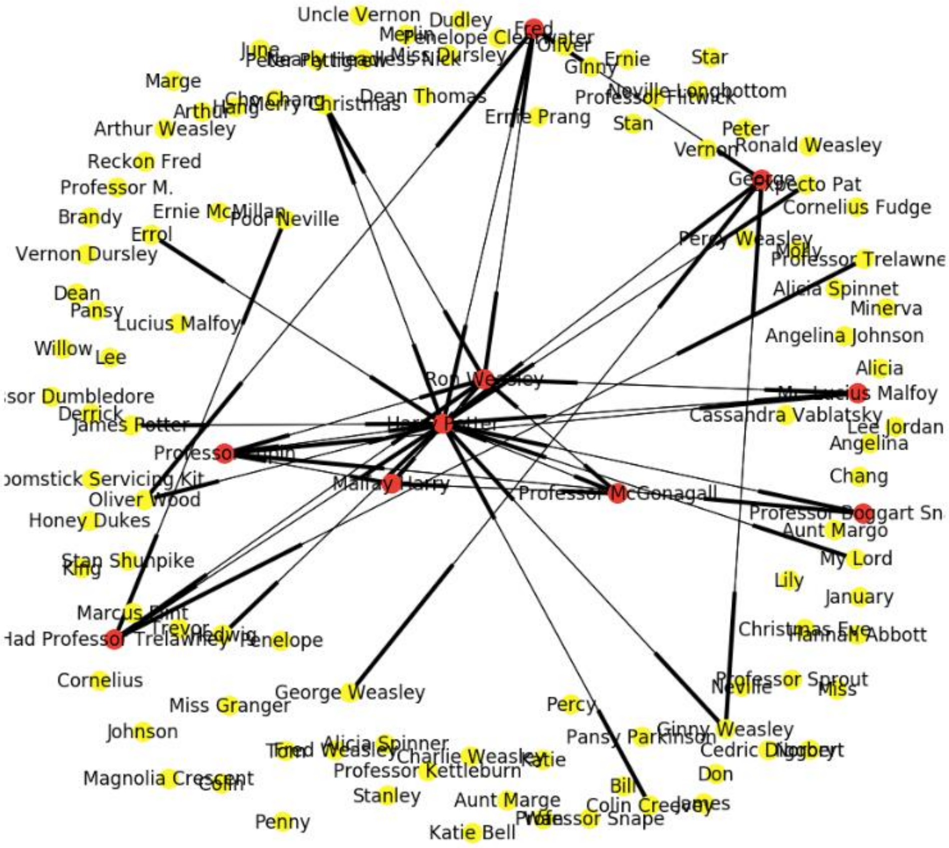
person is mentioned twice in the range of 50 words, update the positions value but the values in the matrix are not changed.

8th step: Plotting

Use networkx DiGraph to plot the graph. To make the graph more clean and easy-to-read, only those connections that have value larger than 5 are shown. Then sort the page rank value for each node and only picked out the nodes with top 40 page ranks (meaning if more than one node have the same Pagerank value, there could be more than 40 nodes presented) to be shown in the graph. Top 10 characters are shown in red and the rest characters are shown in yellow.

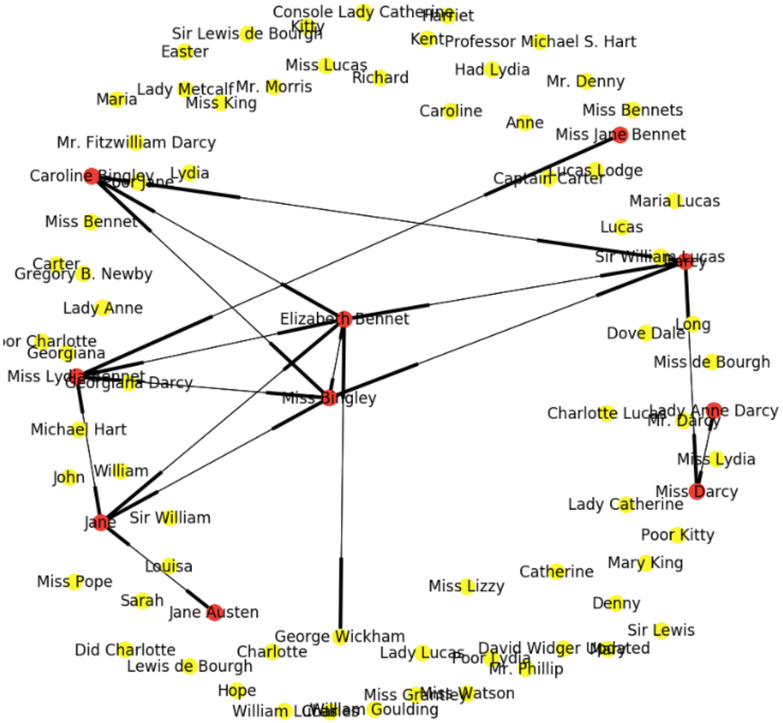
Discussion and Results:

1. Harry Potter 3



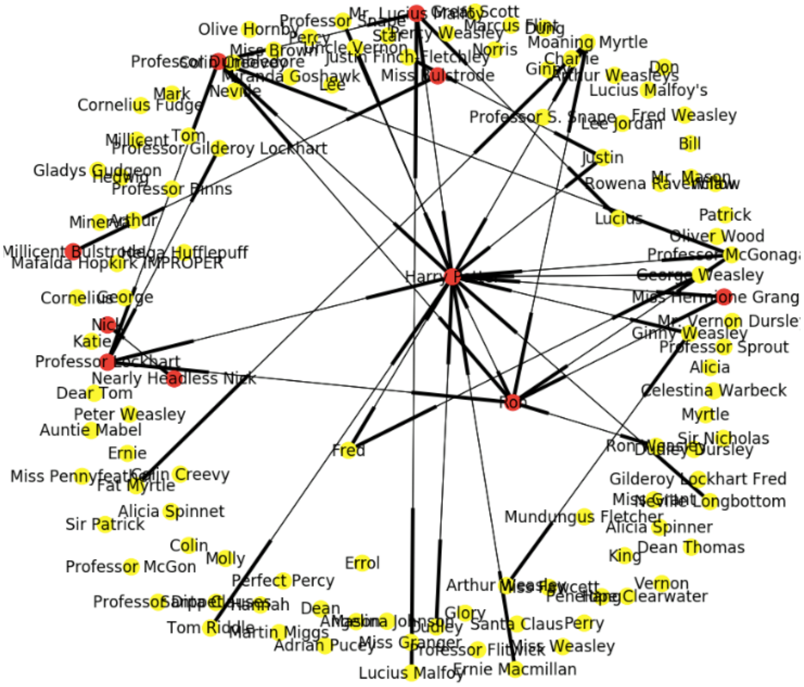
From this graph, we can see that Harry Potter lies in the center in the graph, meaning that he has most connections with all the other characters in the book. Other characters shown in red are also important in this network: Professor Lupin, Malfay Harry, Professor Snape, George, Fred, Ron Wesley, Professor Helawney and Professor McGogan.

2. Pride and Prejudice



Elizabeth Bennet, the protagonist in the book “Pride and Prejudice”, lies in the center of our network graph.

3. Harry Potter 2



From this graph, Harry Potter, the protagonist also lies in the center. The other important characters are: Fob, Mios Herroione Grang, Miss Sulctrode, Mr. Lucius and

Headless Nick.

To Be Resolved:

1. We still have to manually pick out the wrongly captured character names due to the incomprehensive model. To refine the model, we are thinking of incorporating machine learning algorithm, such as training the model to find out which words are normally after a person’s name, to be used as a cross-reference with our existing model.
2. We weren’t able to figure out how to eliminate those special characters (such as €,™,∞) in most of the book text files we found online. This makes our model only restricted to only those text files without any special characters.