Journal Report 4
9/23/19-9/29/19
Abby Kim
Computer Systems Research Lab
Period 2, White

Daily Log

Monday September 9

Looked at how the tokens file tags people and how the book.html file lists different characters. trying to see how to standardize the names of different characters and sort them by gender.

Tuesday September 10

Did not feel well in the morning :(. Felt better later though and ended up tracing the people tags through with the token numbers and references, so I kind of understand the way words are connected to people specifically.

Thursday September 12

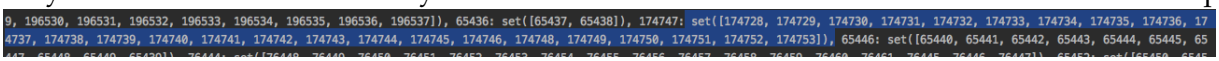
Continuation of Tuesday. Traced Oliver Twist and Rose a couple times. Started coding the algorithm I (kind of) figured out.

Timeline

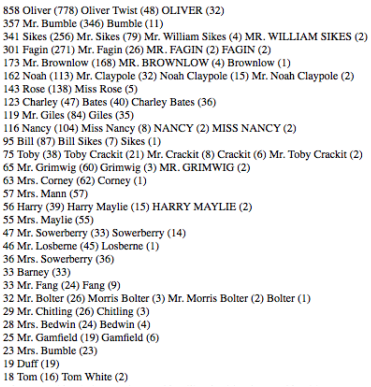
Date	Goal	Met
Two Weeks Ago	"Implement the algorithm I drew up. I'm kind of bad at coding so there will probably be a lot of bugs so this might take a lot of time."	done (toit)
Last week	Figure out the next part of the algorithm regarding filtering out characters by gender	yesh shir
This week	Implement that algorithm	N/A
Today plus 1 week	Graph the data points generated from the previous week and see what it looks like. (This seems simple but I'm also really bad at math so this will probably take a week)	N/A
Today plus 2 weeks	Figure out how to run other books with the pipeline	N/A

Reflection

Pretty cute week not going to lie. So the next part I want to code up this week has to do with extracting words that are associated with different people. So right now I have the grouped words with other words associated them by using their token id. Right now it's a giant dictionary with a root head as the key and a set of all words connected to it as the value. Example:



I need to start by importing the character tags from the html file that looks like this:


858 Oliver (778) Oliver Twist (48) OLIVER (32)
357 Mr. Bumble (346) Bumble (11)
341 Sikes (256) Mr. Sikes (79) Mr. William Sikes (4) MR. WILLIAM SIKES (2)
301 Fagin (271) Mr. Fagin (26) MR. FAGIN (2) FAGIN (2)
173 Mr. Brownlow (168) MR. BROWNLOW (4) Brownlow (1)
162 Noah (113) Mr. Claypole (32) Noah Claypole (15) Mr. Noah Claypole (2)
143 Rose (138) Miss Rose (5)
123 Charley (47) Bates (40) Charley Bates (36)
119 Mr. Giles (84) Giles (35)
116 Nancy (104) Miss Nancy (8) NANCY (2) MISS NANCY (2)
95 Bill (87) Bill Sikes (7) Sikes (1)
75 Toby (38) Toby Crackit (21) Mr. Crackit (8) Crackit (6) Mr. Toby Crackit (2)
65 Mr. Grimwig (60) Grimwig (3) MR. GRIMWIG (2)
63 Mrs. Corney (62) Corney (1)
57 Mrs. Mann (57)
56 Harry (39) Harry Maylie (15) HARRY MAYLIE (2)
55 Mrs. Maylie (55)
47 Mr. Sowerberry (33) Sowerberry (14)
46 Mr. Losberne (45) Losberne (1)
36 Mrs. Sowerberry (36)
33 Barney (33)
33 Mr. Fang (24) Fang (9)
32 Mr. Bolter (26) Morris Bolter (3) Mr. Morris Bolter (2) Bolter (1)
29 Mr. Chitling (26) Chitling (3)
28 Mrs. Bedwin (24) Bedwin (4)
25 Mr. Gamfield (19) Gamfield (6)
23 Mrs. Bumble (23)
19 Duff (19)
18 Tom (16) Tom White (2)

I'm going to try two ways of filtering the people out. The first is just looking at the root head and determining if it's lemma is in the list of names imported. If it is keep it and add it to a new bucket of words for that character, if not ignore it.

The second way involves looking at every token, not just the root heads. The html tags whether a character is the agent of patient, so I would count how many times they are each of those things then bucket words accordingly.