Visualizing Social Groups in Literary Fiction

Ashoka Lella
School of Computer Science
University of Texas Dallas
avl160230@utdallas.edu

Abstract

Novels often contain characters that interact with each other or share common characteristics forming social groups e.g. people belonging to the same family or tribe or profession or close friends. This project focuses on visualizing these frequently occurring characters in prose using an embedding technique called t-SNE (t-Distributed Stochastic Neighbor Embedding). The main idea involves using a Named Entity Recognizer to extract character names from each line and build a feature vector for each character by observing entities that occur around them.

1 Introduction

Recent work in NLP has begun to exploit the potential of extracting social networks from text corpora (Nusratullah et al., 2017) which could be leveraged to build behavioral models; Bamman et al. (2013) explicitly learn character personas in a dataset of Wikipedia movie plot summaries. More recently Bamman et al. (2014) took this a step further by incorporating the influence of individual authors while inferring latent character types into modeling character personas that cut across different authors.

One commonality among all of these approaches to model characters is the procedure is fairly involved for a nascent area of research. Most often simple techniques are sufficient for modeling a system even though other techniques would help in more accurate interpretations. This project focuses on exploring a more elementary approach to model characters semantically and tests the model by visualizing characters. Although the model presented in this report focuses on literary fiction, it can be applied to other real-world textual content like news articles to build character (or entity) models to identify types of people, organizations, locations, etc.

2 Related work

Analysis of literature using more semantically oriented techniques has been rare, most likely because of the difficulty in automatically determining meaningful interpretations. Some exceptions include work on learning common event sequences in news stories (Chambers and Jurafsky, 2008), an approach based on statistical methods, the development of an event calculus for characterizing stories written by children (Halpin et al., 2004), a knowledge-based strategy.

While researchers have not attempted to construct a model to gauge character similarities in literature, people have tried to build social networks based on conversational interaction (Elson et al., 2010) and extract set of social events from news stories to build social networks from the text (Agarwal, 2011). The work in this project focuses on exploring an alternative way to build and visualize similarities among characters in literary fiction.

3 Approach

3.1 Character Identification

The first challenge was to identify character names by "chunking" names such as *Harry Potter* from the text. Each line from each novel has been processed by using Natural Language Tool Kit to extract Parts of Speech from the sentence (Bird et al., 2009) which further got processed by Stanford NER tagger (Finkel et al., 2005) to extract noun phrases that were categorized into entity types like Location, Person, Organization, etc. Normalizing partial names to correct full names (eg. Harry or Potter to Harry Potter) is a difficult problem as it is hard to distinguish full names among family members. Some other problems include names being mentioned together in sequence, titles in names (Sir, Mr, Aunt, Madam, Professor) and language slurs (eg. *Arry Otter*).

Although normalizing character names play an important role in feature building, it is not the primary objective of the project. So a simplistic approach was chosen to normalize names where a bigram model of all the frequently occurring names was built using NER tagger. If the word pair occurs more than a set threshold number of times, then its treated as a valid full name and any occurrence of his/her first name is replaced with the full name. The last names are never normalized. Here we are making an assumption that all characters have at most two words in their full name. As most of the characters do not break this assumption, this model performs reasonably.

3.2 Feature Building

For each character, a list of all other nouns occurring in the mentioned sentence along with their respective counts is stored. For each character, the most frequently occurring ten nouns are added into the feature set. Say, the book contains five characters, the size of the feature set may lie anywhere between ten and fifty as some of the frequently occurring nouns overlap.

3.3 Visualizations

The feature vector size for each character is somewhere in the range 230 to 280 depending upon the diversity of characters in the novel. Since it is not possible for humans to visualize more than 3 dimensions, the feature size is reduced to 2, using dimension reduction techniques such as PCA (Principal Component Analysis) or t-SNE (t-Distributed Stochastic Neighbor Embedding) (Maaten and Hinton, 2008). But PCA was not used because similar items cannot be plotted next to each other.

3.4 Clustering

Apart from plotting the results to manually identify clusters, alternative clustering algorithms were also applied. Most clustering algorithms require parameter tuning or specifying the number of clusters beforehand (k-NN). One such algorithm which does not require cluster count is affinity propagation (Frey & Dueck 2007). This clustering algorithm was applied to character vectors without dimensionality reduction for comparing manually extracted clusters and algorithmic clusters.

4 Data

All the analysis was done on Harry Potter series (books 1 to 5) by J.K. Rowling. Online stores publish books in ePub format compatible with electronic devices and suitable for consumption for general consumers. However, epub is not suitable for analysis so it is converted to a suitable ascii text format (a .txt file).

5 Experimental Results

Each book took about 10 to 15 hours to get processed. The bottleneck operation turned out to be POS and NER tagging. Due to the time-intensive nature of the process, only 5 books were analyzed. All the programs were run on *csgrads1* server. Due to the space constraints, only results for the first two books were shown in this report. All the annotations in the images (red lines) were done manually.

5.1 Book 1

Since Harry Potter was the lead character, he got positioned at the center of the embedding shown in Figure 1. All his neighboring characters include his friends and enemies. Interestingly, all of Harry's friends got positioned towards his left and enemies towards the right. The school houses got clustered together. There are a couple of clusters associated with peoples profession. All Professors at Hogwarts got clustered together at the bottom and shop owners got clustered together. Another, type of cluster is a location-based cluster - all people living near

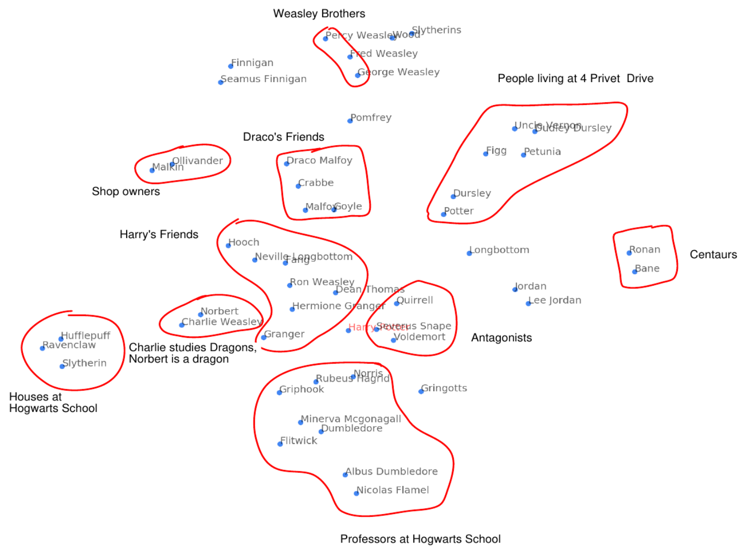

Figure 1: Book 1 - The Sorcerer's Stone

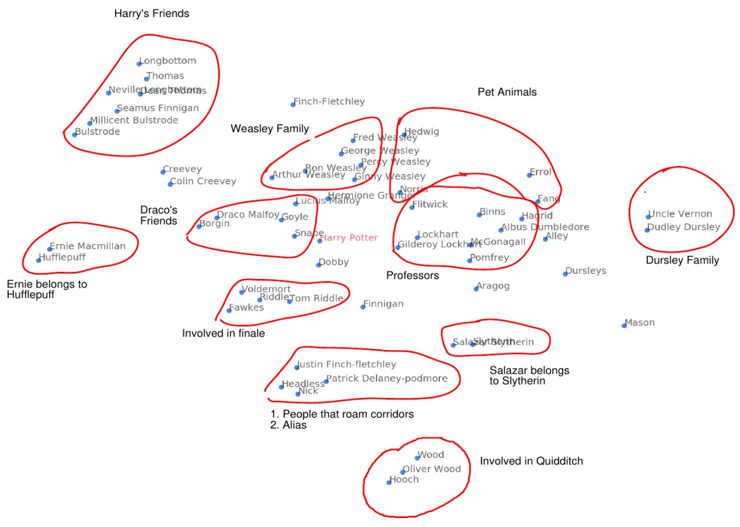

Figure 2: Book 2 - The Chamber of Secrets

Table 1: Book 1 - The Sorcerer's Stone

Cluster	People in cluster
0	Ravenclaw,Slytherin,Hufflepuff,Hooch
1	Nicolas Flamel,Albus Dumbledore
2	Norris,Seamus Finnigan,Finnigan
3	Ronan,Bane,Griphook,Malkin
4	Granger,Charlie Weasley,Voldemort,Harry Potter,Severus Snape, Quirrell,Gringotts,Dumbledore,Rubeus Hagrid,Minerva Mcgonagall,Flitwick
5	Slytherins,Percy Weasley,Fred Weasley,Wood,George Weasley
6	Petunia,Uncle Vernon,Dudley Dursley,Figg
7	Goyle,Crabbe,Malfoy,Pomfrey,Draco Malfoy
8	Fang,Neville Longbottom,Norbert,Dean Thomas,Ron Weasley,Hermione Granger
9	Potter,Longbottom,Dursley,Ollivander
10	Lee Jordan,Jordan

Table 2: Book 2 - The Chamber of Secrets

Cluster	People in cluster
0	Seamus Finnigan,Thomas,Finnigan,Bulstrode,Millicent Bulstrode, Longbottom,Neville Longbottom,Dean Thomas,Errol
1	Ginny Weasley,Percy Weasley,George Weasley,Fred Weasley,Ron Weasley,Hedwig
2	Headless,Nick,Justin Finch-fletchley,Patrick Delaney-podmore
3	Norris,Fang,Albus Dumbledore,Alley,Aragog,Hagrid,Slytherin,Pomfrey, McGonagall,Dursleys,Salazar Slytherin,Binns,Finch-Fletchley
4	Gilderoy Lockhart,Lockhart,Harry Potter,Snape,Flitwick
5	Voldemort,Riddle,Tom Riddle,Fawkes
6	Colin Creevey,Creevey
7	Mason,Uncle Vernon,Sprout,Dudley Dursley
8	Ernie Macmillan,Hufflepuff
9	Arthur Weasley,Borgin,Goyle,Hermione Granger,Lucius Malfoy,Dobby,Draco Malfoy
10	Wood,Oliver Wood,Hooch

Harry's house (4 Privet Drive) got positioned together. Creatures belonging to the same race (Centaurs) got positioned at the right end.

5.2 Book 2

Again similar to book 1, Harry gets placed at the center in Figure 2. Similar to book 1, members of Weasley family get placed together, Draco and his friends get placed together, professors placed together and Harry's friends get placed together. Interestingly, people who roam corridors at night get placed together. Corridor roaming plays an important role in book 2 as people doing so get petrified, which is the central theme in the story. Also, a new *belongs to* relationship can be observed with Ernie Macmillan, Hufflepuff (Ernie belongs to Hufflepuff house) and Salazar Slytherin, Slytherin (Salazar Slytherin belongs to Slytherin house). There is also a new temporal cluster involving Tom Riddle, Voldemort, and Fawkes all of which are involved in the story's finale.

6 Conclusion

While the model does a decent job of placing similar characters together, some neighboring characters are not closely related. This problem can be clearly be observed with frequently co-occurring characters in the story. For example, *Harry* and *Snape* (Harry's Professor) are not closely related but, Snape turns out to be the closest neighbor in both books.

Sometimes there is an imbalance in the number of times a name is presented in the text. This problem can be clearly noticed in Hogwarts houses. While the school contains a total of four houses only three of them got detected (the house of Gryffindor has a low word count in the book).

One way in which the effectiveness of the model could be greatly improved is by using proper noun coreference resolution as approximately 74% of references to characters in books come from pronouns (Bamman et al. 2014). This was not pursued due to lack of time and lack of good programming libraries that do a good job at this task.

Name resolution was one of the hardest problems. Some characters are mentioned by their last names eg. Lee Jordan gets mentioned as Jordan. The last name Jordan could not be resolved to Lee Jordan because there could be other characters who share the same family name.

In conclusion, this work provides a solid foundation to analyze character similarities providing a more semantic understanding of characters. This model can be applied to real-world text corpuses such as news articles to recognize similarities between types people (CEOs, politicians, athletes etc) and organizations.

7 References

Nusratullah, K., Shah, A., Akram, M. U., & Khan, S. A. (2018). Detecting Change from Social Networks using Temporal Analysis of Email Data. In *Information Technology-New Generations*, Springer, Cham, pages 297-304.

David Bamman, Brendan O'Connor, and Noah A. Smith. 2013. Learning latent personas of film characters. *Proc. of ACL.*

Bamman, D., Underwood, T., & Smith, N. A. (2014). A bayesian mixed effects model of literary character. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages. 370-379.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics (ACL-08)*, pages 789–797, Columbus, Ohio.

Harry Halpin, Johanna D. Moore, and Judy Robertson. 2004. Automatic analysis of plot for story rewriting. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'04)*, Barcelona.

Elson, D. K., Dames, N., & McKeown, K. R. (2010, July). Extracting social networks from literary fiction. In *Proceedings of the 48th annual meeting of the Association for Computational Linguistics (ACL 2010)*, pages 138-147.

Agarwal, A. (2011, June). Social network extraction from texts: A thesis proposal. In *Proceedings of the ACL 2011 Student Session*, pages. 111-116.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 363–370.

Maaten, L. V. D., & Hinton, G. (2008). Visualizing data using t-SNE. In *Journal of machine learning research*, 9(Nov), 2579-2605.

Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. *science*, 315(5814), 972-976.