December 7, 2015

Extracting Social Network from Literature to Predict Antagonist and Protagonist

Matt Fernandez
matthew0

Ben Ulmer
ulmerb

Abstract
The study of communities in literature is a relatively unexplored space, due to the many layers of analysis required to extract information from a book to a network. This paper addresses that void by analyzing the general structure of graphs extracted from fairytales, as well as trying to predict the protagonist and antagonist of that story. Our initial system included general graph feature analysis including triad analysis. Beyond that, we also create three prediction algorithms for weighted and signed multimodal networks, to identify the protagonist and antagonist.

Introduction

We were inspired by David Jurgens, Hardik Vala, Andrew Piper, and Derek Ruths, who investigated the question of detecting characters in literary texts, to tackle a problem that built upon that task: analyzing the social graph of a network built from a story. In our analysis, we aim to extract several general features of the graph, and investigate the use of triads for those features, as well as create several algorithms for protagonist and antagonist identification.

We believe this question is interesting for several reasons. Not only is this area extraordinarily unexplored and interesting in and of itself, but it also has implications for future work on networks with multiple signed and weighted attributes. To begin with, while a substantial portion of modern research is focused on examining online social networks, such as Facebook, we feel that the study of social networks in literature is underrepresented. By creating algorithms for analyzing these literary social networks, we provide tools for others to use to understand material ranging from how literature changes over time to how genres are unique.

Furthermore, we approach the problem of identifying the protagonist and antagonist, which is useful for work currently being performed in the computer comprehension and question answering domains. However, although our analysis on protagonist and antagonist identification was conducted on literature, it is not limited to it. Our algorithms can be extended to identify the major positive and negative players any signed and weighted multimodal graph.

We hope our work provides a baseline for a system that transitions from literature to a network to character analysis.

Literature

As mentioned before, we referenced David Jurgens and company's paper "Mr. Bennet, his coachman, and the Archbishop walk into a bar but only one of them gets recognized: On The Difficulty of Detecting Characters in Literary Texts." Their paper describes an algorithm to detect characters in literary novels.

A core paper for analyzing social graphs came from Leskovec et al. and focused on "Signed Graphs in Social Media". This work highlighted several algorithms and network analysis techniques, which we implemented for fairy tale analysis. Most importantly this work contained new ideas about balance within signed networks as they relate to social graphs. Leskovec states that we expect typical social graphs to contain triads with 1 or 3 positive connections, which indicate a group of friends or a pair of friends plus common enemy respectively. We use this idea to define a strongly balanced social graph in our analysis

We also draw upon a paper titled “The Slashdot Zoo: Mining a Social Network with Negative Edges” by Kunegis et. al., which outlines a medley of analysis techniques for signed graphs ranging from clustering coefficient to popularity measures. We find the latter particularly interesting for protagonist and antagonist detection, although we extend his algorithm for use on weighted multimodal networks.

Finally, we were inspired by the Katz algorithm as explained by Kang et al. in their paper to the Society for Industrial and Applied Mathematics, “Centralities in Large Networks: Algorithms and Observations.” Although not many of the principles carry over to our algorithm, it provided insight, and was often the starting point for algorithm brainstorming.

Data Collection

We drew our data from literature made public on the internet. Books ranging from children’s stories to the Bible are available in plain text from sites like project Gutenberg (Gutenberg alone contains 50,000 free e-books) (Gutenberg).

Methods

We will now explore the major parts of our algorithm in detail and show the results of each individual component. Before we performed this algorithm, we stripped the story text of images, chapter headings, and anything else that was not the text of the story.

Social Network Extraction

Algorithm

Once the text is ready to be processed, we approach the problem of character detection and coreference, and then the question of sentiment analysis. For coreference, we follow a four step approach:

1. Run StanfordNLP to find coreferent mentions
2. Identify the mention chains that correspond to characters or groups of characters
3. Replace the original text
4. Post process to identify ambiguous reference chains such as first person pronouns

We then extract the relationships between characters and identify the sentiment of those relationships by identifying co-occurrence, extracting the verb phrase, and use a library called SentiWordNet to perform sentiment analysis on that verb phrase.

Results for Character Detection

Our character detection algorithm achieved an F1 score of .815, with an average precision of .75 and an average recall of .933. This compares very favorably with the average F1 of .5713 score reported by Vala et. al. in their paper. Part of this discrepancy lies with the fact that Vala et. al. were analyzing much more complex novels, such as *Sherlock Holmes* and *Pride and Prejudice*, as opposed to fairy tales such as Hansel and Gretel, which have much longer casts of characters and more intricate writing, but even still, our results are impressive.

Results For Sentiment Analysis
To test our sentiment analysis system, we check whether we correctly classified the relationships between characters. We tested on five fairy tales, for which we hand-parsed the gold solutions. Summing together all examples, both positive and negative, we find a sentiment analysis accuracy of 71%. Considering that fairy tales often involve significant amounts of deception -- Little Red-Cap, Hansel and Gretel, and The Goose-Girl all involve one character pretending to be another -- it can be very difficult to classify these relationships accurately, and we are pleased with our result.

Network Analysis
General Graph Analysis
Our network analysis algorithms take character relationship data and plot a graph that includes information about the positivity, negativity, and significance of each character and their relationships. From this graph we set out to understand the complexity and bipartiteness of the graph.

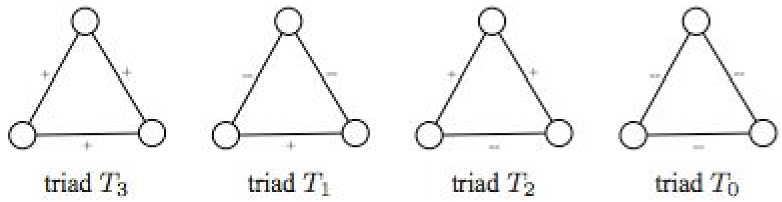
Our processed data from the character identification and sentiment extraction is stored as a list of connections and the positive and negative weights corresponding to that connection. This data allows us to construct a network using SNAP and to create multigraphs for positive and negative categories.

Our analysis starts with simpler layers of information and trends towards more complex measurements. The first batch of features (which takes into account positive and negative weights) include:

1. Individuals with the largest degree
2. Individuals with the largest value for sum of all edge weights
3. Largest difference between positive and negative weights to determine the most positive characters
4. Least difference (can be negative) between positive and negative weights to determine the most negative characters

This set of features provided insight about the structure of the graphs, but didn't lead to significant results for predicting information about the story.

We then explored graph analysis through triad analysis on the social networks. By examining the triads that exist in a social graph we are able to find two core pieces of information about the story. First, we can identify if the graph is strongly, weakly or not structurally balanced. This information allows us to identify stories that are more complex and are more likely to be cases of deception or complicated characters because the relationships triads are unbalanced. The



Triad Types - From Leskovec

second piece of information gained from triad analysis is an understanding of relationships . For examples characters at the negative end of T1 triads are much more likely to be the antagonist and the protagonist will often be in T3 triads with other allies or positive characters.

This triad analysis, in addition to other features that aren't related to triads, but add intuitive information about character's emotional valence in the story, were focused into a simple but effective rule-based classifier for identifying protagonist and antagonist. The other features are:

1. Number of times appeared in the story overall
2. Degree of the individual
3. Value for sum of all edge weights in the positive / negative direction

Primary Results

Standard version for Protagonist and Antagonist with no thresholding

Fairy Tale	Protagonist ID / Error	Protagonist Error %	Antagonist ID / Error	Antagonist Error %	Balanced ID / actual	Balanced Triads %
Hansel and Gretel	correct / 0	0%	incorrect / 4	75.8%	No / No	18%
Rapunzel	incorrect / 1	13.6%	incorrect / 4	170.5%	Weak / Strong	50%
The Goose-Girl	incorrect / 2	1.7%	incorrect / 1	2.2%	No / No	40%
Little Red-Cap	correct / 0	0%	correct / 0	0%	Weak / Strong	40%
The Frog Prince	correct / 0	0%	None*	None*	Strong / Strong	100%
Overall	3/5	3.6%	1/4	62.3%	3 / 5	-

* There is no villain in this story

Key:

1. Protagonist Error: Number of characters ranks away from correct prediction
2. Protagonist Error Percentage: Increasing the correct protagonist by this amount will lead to a correct prediction

3. Antagonist Error: Number of characters ranks away from correct prediction
4. Antagonist Error Percentage: Increasing the correct antagonist by this amount will lead to a correct prediction
5. Balanced: Strong = only T3 and T1 triads; Weak = T3, T1, and T0 triads; No = at least one T2 triad (We calculate the triads in two ways so we use the lowest standard)
6. Balanced Triads Percentage: Ratio of fully balanced triads. For most stories the goal should be greater than 80% because few of the triads are unbalanced in gold graphs

Error analysis

Our system shows a lot of promise for predicting protagonist, antagonist and story complexity based on social network analysis. For 3 of the 5 stories, we were correctly able to identify the protagonist of the story and for the other 2 we were off by only a small margin. By expanding and intelligently training our feature weights, we could likely identify the correct protagonist and antagonist for the vast majority of stories.

Our antagonist identification accuracy was lower because in both “Hansel and Gretel” and “Rapunzel” the villain is a complex character that has a positive relationship that turns sour. These types of relationships aren’t accurately shown in our data because we ignore any temporal aspect of the the relationships. Another issue is simply that the antagonist doesn’t always form lots of enemies (as in Little Red-Cap), so their character may not be as central which makes it harder to identify as the villain.

Because of the variety of moralities and complexities of the fairy tales analyzed its difficult to gain too much from the balance of the graph. Our system correctly identified 3 out of 5 of the balance identifications, which indicates that although our network edges aren’t perfect, the overall relationship network is often correct. One simple, but valuable, conclusion is that the frog-prince is a positive and uncomplicated relationship story because it is strongly balanced. This feature is important for our extension into other types of stories that may have similar character archetypes. This aspect of our research is somewhat weakened by accuracy difficulties at the coreference and sentiment layers, even though our accuracies for those tasks are comparable to or exceed current research for the same task. By staying true to a “raw text to network” model, we accept a hit in our confidence level for network analysis.

However, in the future, with added machine learning algorithms and further statistics beyond positivity and negativity pulled from the data, we could greatly increase the accuracy and significance of those results. Additionally because we focused on fairy tales, we didn’t explore variations in types of stories and differences in authors.

Expanded Protagonist and Antagonist Detection

Summary

We now contract our focus from general analysis to the process of identifying the protagonist and antagonist of a story. We present three novel methods:

Method 1: Iterative Neighbor Algorithm

The following is our iterative asymptotic algorithm which takes into account neighbor scores and the corresponding edge weights:

$$score_{n,i} = score_{n,\,i-1} + \left(\tfrac{K}{i} * \sum_{neighbor\, j} score_{j,\,i-1} * edge_{j,n} \right)$$

$$n = current\ node,\ i = round\ number,\ K = update\ factor$$

We initialize the score to be the result of our rule based classifier above. The intuition behind this algorithm is that protagonists will have positive multigraph components with positive neighbors, and negative ones negative neighbors. Overall, the system is able to use neighboring information to gain further clarity about the protagonist and antagonist.

Method 2: Modified Signed and Weighted Spectral Ranking Algorithm

We adapted the pagerank algorithm to adapt to a multi-modal signed and weighted graph in order to identify the protagonist and antagonist of the tale. In place of the teleportation probability, we made use of a normalized count for a node i :

$$TeleportationProb_i = \frac{i_{count}}{\sum_{node\ j} j_{count}}$$

This matches the intuitive idea that the results should be skewed towards characters that appear often, as they are often the protagonist and antagonist. To have the “random surfer” follow an edge, we select an attribute α of an outgoing edge γ from node η with probability:

$$EdgeModeProb_{\gamma_\alpha} = \frac{|\gamma_\alpha|}{\sum_{edge\ \delta\ from\ \eta} \sum_{attribute\ \beta\ from\ \delta} |\delta_\beta|}$$

This rule enforces the concept that our “random surfer” should follow stronger edges attributes over weaker ones. Having selected an edge attribute from above, we have the update rule:

$$Weight_\eta \coloneqq weight_\eta + \gamma_\alpha$$

which intuitively follows from the fact that a negative relationship is indicative of a villain, whereas a positive relationship is an indicator of a protagonist.

Method 3: Modified Katz Centrality Algorithm

Our next algorithm is inspired by the Katz Centrality Algorithm, an algorithm used for estimating the influence of a node:

$$\overrightarrow{C}_{\mathrm{Katz}} = ((I - \alpha A^T)^{-1} - I)\overrightarrow{I}$$

Where *A* represents an adjacency matrix of connections, and α is an attenuation factor. The Katz algorithm intuitively measures the number of walks between a pair of nodes. We used this principle as the basis of an algorithm to reinterpret multimodal edge weight as a function of possible walks:

1. Create a series of adjacency matrices for each attribute, where the values of the matrices represent the normalized unsigned values of the connection
2. For node a in the graph:
    a. For node b in the graph:

b. Set $weight_{ab} := 0$

i. For node c in the graph:

1. If the nodes are distinct,

$$weight_{ab} := normalizedWeight_{ac} * normalizedWeight_{cb}$$

3. Assign to each node a centrality score that is their weighted degree in $weights$
4. We then assign to each node n a “protagonist score” that is:

$$protagonistScore_n = centrality_n * \sum_{node\ i} \sum_{\substack{Adjacency\ Graph\ A \\ for\ Attribute\ a}} A_{ni}$$

This algorithm is based off of the assumption that the protagonist is a central character of the story, and, more importantly, the protagonist is often the character that links different communities. By skewing centrality towards characters that are part of multiple communities, we are more likely to identify the protagonist. Once we have identified the protagonist, we have a similar method of determining an “antagonist score”:

$$antagonistScore_n = centrality_n * \left(1 - \sum_{node\ i} \sum_{\substack{Adjacency \\ Graph\ A \\ for\ Attribute\ a}} A_{ni}\right) + \sum_{node\ i} \sum_{\substack{Adjacency \\ Graph\ A \\ for\ Attribute\ a}} - protagonistScore_i * A_{ni}$$

The difference from the protagonist calculation is that these algorithms are performed sequentially, so once the protagonist score is calculated, we are able to add a correction factor attempts to identify enemies of the protagonist as the antagonist.

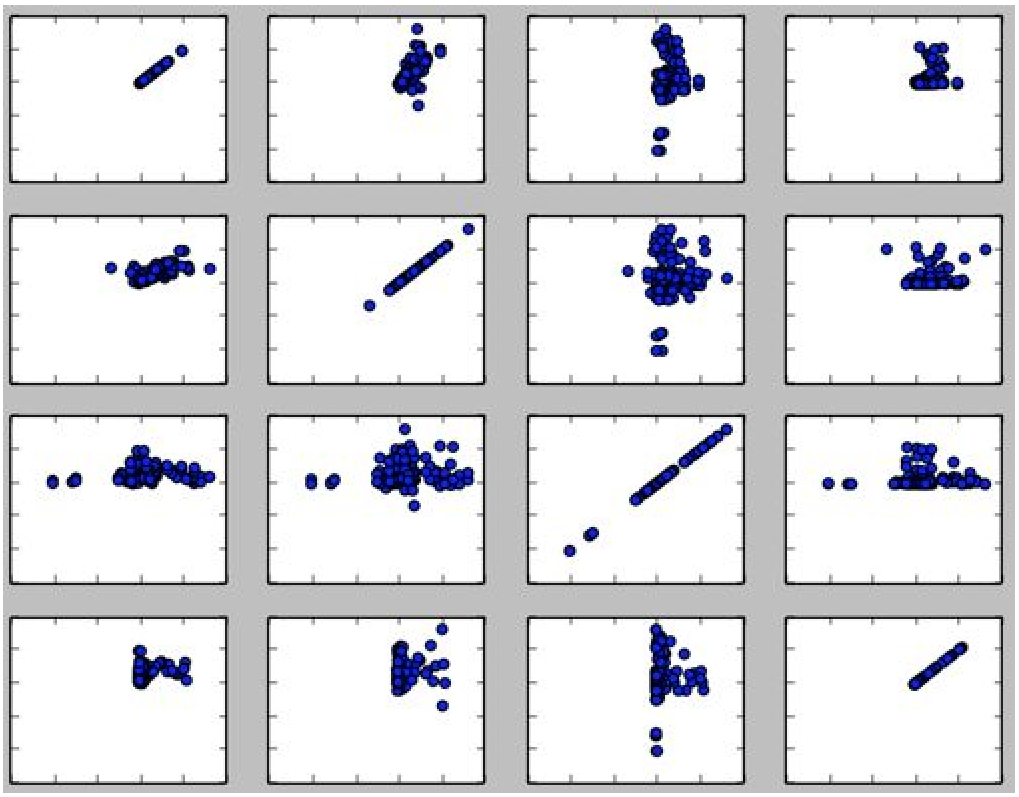
Results of Protagonist and Antagonist Detection And Discussion:

	Protagonists correct	**Protagonist distance**	**Antagonists correct**	**Antagonist distance**
Triad	.636	0.877	.166	0.447
Iterative Neighbors	.416	0.745	.363	0.503
Spectral Ranking	.333	0.729	.250	0.579
Modified Katz	.416	0.804	.083	0.584

Above is the table reporting the results of the various techniques as performed on 12 fairy tales: Little Red-Cap, Hansel And Gretel, The Frog Prince, The Goose-Girl, Rapunzel, Cat And Mouse In Partnership, Mother Holle, Old Sultan, The Adventures Of Chanticleer And Partlet, The Fisherman and his Wife, The Valiant Little Tailor, and Briar Rose. Our methods return a ranking of protagonists and antagonists, and the “protagonists correct” and “antagonists correct” column correspond to the average percentage of times a method correctly identified the protagonist or antagonist as being the top of that respective ranking. We also report the average distance down the ranking that a method's prediction was. For example, if our prediction of protagonist is ranked second in a group of 10 characters, the protagonist distance would be .8, as it identifies the true protagonist to be a more likely protagonist than 8 of the 10 characters.

The naive triad method is the best at correctly identifying protagonists, and the iterative neighbor approach is best at correctly identifying villains. In spite of its low percentage of antagonists correct, the modified Kantz method has the best average distance.

Method Similarity Measure:



The four protagonist and antagonist methods plotted against one another. From top/left to bottom/right, the figures show triads, iterative neighbor, spectral ranking, Katz centrality.

We see that there is a positive correlation between our iterative neighbor and triad models, which is understandable, as the triad score is used as a seed value for our iterative neighbor model. We can also tell that some models, such as the modified Katz model, output a smaller range of values However, beyond that, what is notable is the lack of correlation between models. This is in part due to the fact that the models are built upon very different foundations - a spectral ranking system, an algorithm that modifies edge weights based upon potential walks, and a rule based classifier. Their performance is dictated by the properties of the networks they analyze. In the future, we could explore an ensemble method that combines the three, with hyperparameters based off of the characteristics of the networks to adjust the influence of a particular algorithm.

Conclusion

Our paper appears to be the first -- or at least one of the very first -- to dive into not only extracting characters from fairy tales, but also identifying each character's social network and the sentiment of these relationships. Given the lack of available training data, we believe our system performed very well, with an overall coreference F1 score of 0.815, sentiment analysis score of 71%, best protagonist score of 64%, and best antagonist score of 36%. To improve the coreference and sentiment analysis we may do hand-labeling of data to allow for training classifiers, attempting to notice deception, identify specific attributes of relationships in social networks (e.g. best friends, lovers, family, etc.) or more.

By conducting triad analysis and general graph analysis, we are able to conclude information about the story including the existence of deception and complicated characters. This is shown most prominently in Little Red-Cap where the wolf pretends to be the grandmother, which predictably correlates with non-balanced triads involving the grandmother. Our system correctly identifies the protagonist and antagonist indicating that our system has potential for some stories. Additionally, by implementing our own modified versions of a spectral ranking algorithm and Katz Centrality, we are able to identify and understand more about the centrality of characters. This does not always lead to the highest results for protagonist and antagonist, but provides a strong basis for future research.

Because this is the first project of our knowledge to start with raw text and create social networks without supervision, it's difficult to asses our results relative to other methods. Because we use the latest in algorithms from Leskovec's triad analysis and adapted centrality measures to our signed multimodal graphs, we believe to have staked a claim as a solid baseline for similar tasks in the research community.

Author Note

This project was shared with CS224N: Natural Language Processing. The network analysis represented the bulk of our research and that is for this class

Works Referenced

Borodin, A., Roberts, G., Rosenthal, J., & Tsaparas, P. (n.d.). Finding authorities and hubs from link structures on the World Wide Web. *Proceedings of the Tenth International Conference on World Wide Web - WWW '01*.

Elson, D. (2010). Extracting Social Networks from Literary Fiction. *ACL2010*. Retrieved November 17, 2015, from http://www1.cs.columbia.edu/~delson/pubs/ACL2010-ElsonDamesMcKeown.pdf

Esuli, Andrea, and Fabrizio Sebastiani. "SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining." LREC 2006 (2006). Web. 8 Dec. 2015.

Finn, A., & Kushmerick, N. (n.d.). Learning to classify documents according to genre. *J. Am. Soc. Inf. Sci. Journal of the American Society for Information Science and Technology,* 1506-1518.

Fowler, J. (2006). Connecting the Congress: A Study of Cosponsorship Networks. *Political Analysis,* 456-487.

Free ebooks by Project Gutenberg - Gutenberg. (n.d.). Retrieved November 17, 2015, from http://www.gutenberg.org/

Kang et. al., (2011). Centralities in Large Networks: Algorithms and Observations. Retrieved December 5, 2015, from http://www.cs.cmu.edu/~ukang/papers/CentralitySDM2011.pdf

Jerome Kunegis et. al. The Slashdot Zoo: Mining a Social Network with Negative Edges 2009. ACM WWW 2009, Madrid, Spain

Leskovec, Jure, Daniel Huttenlocher, and Jon Kleinberg. "Signed Networks in Social Media." Proceedings of the 28th International Conference on Human Factors in Computing Systems - CHI '10 (2010). Print.

Manning, Christopher D., Surdeanu, Mihai, Bauer, John, Finkel, Jenny, Bethard, Steven J., and McClosky, David. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-60.

Newman, M. (n.d.). Analysis of weighted networks. *Physical Review E Phys. Rev. E*.

Vala, H., & Jurgens, D. (2015). Mr. Bennet, his coachman, and the Archbishop walk into a bar but only one of them gets recognized: On The Difficulty of Detecting Characters in Literary Texts.