Ward Character Sentiment Analysis

Aidan Klobuchar

December 23, 2018

1 Introduction

Ward is the wonderful fourth, in progress, web serial by the author Wildbow. It's a tale dealing with superpowers, post-apocalyptic rebuilding, and how of this relates to trauma and second chances. But more importantly, it's a LARGE collection of text, primarily told experienced through a first person perspective. This makes it an enticing dataset for messing around with some NLP. NLP, short for natural language processing, is a very popular, useful, and well-trodden area of machine learning and data science, specializing in the automatic analysis, summarization, tagging, etc. of human created text. One task that can be accomplished using NLP is known as *sentiment analysis*, which automatically determines how positive/neutral/negative a given sentence is. This can be used to analyze social media for brand value, analyze reviews, etc., and is generally accomplished using neural networks fit to large amounts of human-classified data. Given the first person nature of Ward, it is a great place to investigate sentiment analysis, as we can plot the sentiments the protagonist, Victoria Dallon, as a function of chapter, hopefully gaining some insight in how her opinion of certain characters change and accumulate over the course of the story.

Note: All associated code/files can be found on my github page.

2 Methodology

Performing any NLP work requires two foundational step; scarping/gathering the necessary text and cleaning/preparing the gathered text. To scrape the text from the web, I used the Python package 'Beautiful Soup', a package specializing in HTML parsing and web scraping. The text was scraped by gathering the appropriate links through the Ward table of contents page here. This was done somewhat messily, as the links containing posting dates were selected for, when it should probably just be a direct regular expression call. After the text was gathered, it was re-encoded into the utf-8 format (which seems to have some problems dealing with strange punctuation), in order to be compatible with the existing NLP software and the interlude chapters, which are from the perspective of different characters, were removed.

The encoded chapters were then parsed using the powerful NLP framework, 'spacy'. Feeding text to this method allows words to automatically tagged as named entities (NER), have

their part of speech recognized, have the separate words split up, and noting the base forms of all of the attached words (this transformation is known as *lemmatization*). This allows one to easily remove punctuation from the text, transform all words into their base forms (e.g. 'better' is turned into its base form, 'good'), and remove capitalization. However, while these transformations are useful for many NLP related tasks, such as fitting different models to the text, they are counterproductive when using one of the common prefit sentiment analyzers. As such, I skipped doing these tasks for this analysis, though you can see how to do them in commented out sections of the associated Python code. As mentioned above, spacy can also be used to automatically recognize proper noun using *named entity recognition*. While one *could* use this to pull out character names for our sentiment analysis, this isn't useful in this case, as I already know all of the key names/proper nouns, and NER is not perfect; it can output many false positives and will have trouble when a common word also serves as a name)such as with the character Rain) or a character is not referred to using a proper noun (a reference to a sister or mother, for relevant Ward related examples). So, again, I forgo performing NER in this analysis, though the methods to do so are also still present in the code.

I one thing I *do* do, is to automatically combine common phrases into a single word (e.g. the name 'Damsel of Distress' would become the single word 'Damsel_of_Distress'). This is accomplished using the package 'gensim's phrase modeler. This works statistically, noticing when words are adjacent to each other far more often than they would randomly, and then combining them. The modeler can be run multiple times to create larger phrases; one run will create two word bigrams, while a second run will create three word trigrams (and possibly some four word quadrams when two bigrams are combined). I ran the phrase modeler twice in order to create trigram phrases, which should be sufficient for this exercise.

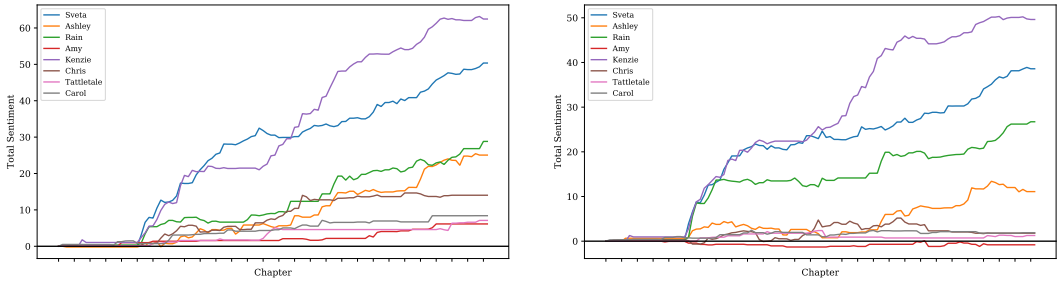
In order to perform the actual sentiment analysis, I decided to use pretrained models, as it takes a good deal of work, know-how, and computational power to create an accurate sentiment analyzer, though I may attempt to create a simple Naive Bayes model at a later date. The two premade options I found were the sentiment analyzer included as part of the TextBlob package and VADER sentiment analyzer that exists as part of the Natural Language Tool Kit (NLTK). In attempting to analyze the sentiments regarding specific characters, we have to select the appropriate sentences. I did this by searching every line throughout the entirety of Ward to see if it contained selected keywords and then checking, using spacy's part of speech feature, if that keyword was the subject in the sentence. The keywords associated with each character are collected in Table 1. Note that I did not investigate the sentiment involving the Vera twins, as the fact that Tristan and Byron share the 'Capricorn' superhero name would make things a bit tricky. The sentiment could then be computed as an average sentiment per opinionated sentence and as a running sum through out the entire work.

Table 1: Keywords Associated with each of the Selected Ward Characters

Character	Keywords
Ashley	Ashley, ashley, Ashley_Stillons, Damsel, Damsel_of_Distress, swansong, Swansong, Mangled_Wings
Rain	Rain, Rain_Frazier, Precipice, of5
Chris	Strange_Mammal, Chris, chris, Chris_Elman, Cryptid, creepy_kid
Sveta	Sveta, sveta, Tress, Garotte, Sveta_Karelina, Space_Squid, Kraken_in_a_Jar
Kenzie	Kenzie, kenzie, Kenzie_Martin, Lookout, Looksee, Kanzi, Heart_Shaped_Pupil
Tattletale	Tattletale, TT, Tats, Lisa
Amy	Amy, amy, Amy_Dallon, Amelia, Amelia_Dallon, Panacea, panacea, sister
Carol	Carol, Brandish, carol, Carol_Dallon, Mom, mom, mother, Mother

3 Results

When collecting the sentiment data, the first question I wanted to answer was to pick a sentiment analyzer between the TextBlob and VADER analyzers. To do so, I looked at the cumulative sentiment for every character, incremented each chapter. This is collected below as Fig. 1. Here we see that the analyzers return fairly similar results. However, the fact that the VADER analyzer exhibited less of a positive bias, with Amy having a negative overall sentiment and Chris dipping into the negative range early in the story made it more appealing to me, as those results better match my expectations. In addition, the VADER analyzer seems to favor Rain over Ashley as compared to TextBlob, and also sees mentions of Carol as about as positive as those of Chris, where TextBlob favors Chris.



(a) Cumulative Sentiment over Time Using the TextBlob Classifier

(b) Cumulative Sentiment over Time Using NLTK's VADER Classifier

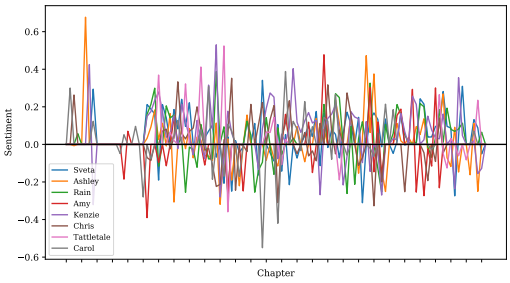
Figure 1: Comparison of the Cumulative Sentiment Total Given by Two Different Classifiers

The next question is how to best display the individual, chapter-by-chapter averaged sentiments. As can be seen from Fig. 2a, leaving the results in raw form leads to very noisy and illegible graph. Thus we can smooth out the results by using previous values to influence the current value. The first and simplest way to do this is to just compute a rolling average. In Fig. 2b this was done using a window size of 5, meaning that the fifth data point would be displayed as the average of datapoints 1-5. Earlier points were just interpolated from

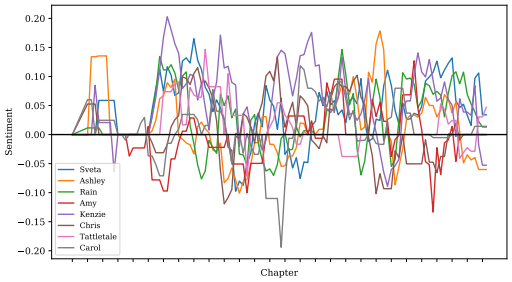
the first and fifth points. This gives a somewhat boxy but certainly more legible result. The second smoothing method we can try is *exponential smoothing*. This is given by the recursive formula

$$s_t = \alpha x_t + (1 - \alpha) s_{t-1},$$

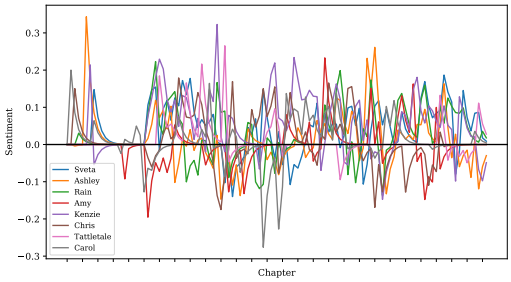
where α is an input smoothing factor ($\alpha = 0.5$ was used here) and t just keeps track of which datapoint we're talking about. Because the value of s_{t-1} is influenced by the previous smoothed points, it's easy to work out this formula's relation to an exponential, though that's neither here nor there. Looking at Fig. 2c, we see that this doesn't help much, do the large number of datapoints, the very noisy nature of the data, and the limited window that exponential smoothing looks at (though I suppose a very low α value may help; I haven't bothered to check yet). As such, I will use the rolling average method for the rest of this little investigation.



(a) Individual Chapter Normed Sentiment Scores



(b) Rolling Average Normed Sentiment Scores

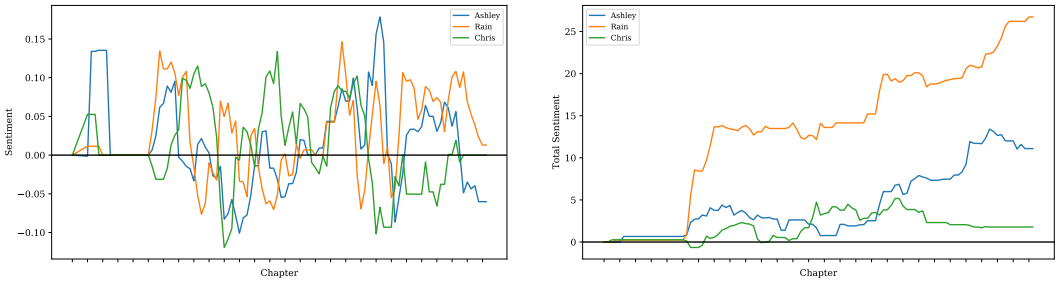


(c) Exponentially Smoothed Sentiment Scores

Figure 2: Comparison of Different Smoothing Methods for the Individual Chapter Scores

With the questions of methodology settled, let's next look at the results of smaller groups of characters, so that we can look at things more closely. We'll start by looking at the scores for Ashley, Rain, and Chris, members of Victoria's team, Breakthrough, whom she has somewhat mixed opinions on. Looking at the total scores first (Fig. 3b, we see that Rain was generally viewed in a positive light, while both Chris and Ashley were viewed in a more mixed light, with the sentiments around Ashley ending up being more positive, while those around Chris ended up being neutral to even negative, fitting given his recent traitorous turn (though this analysis only covers up to chapter 10.11). Looking at the more granular plot in Fig. 3a, we can get a better sense of how the sentiment evolved over time. All three had positive spikes early on, as those chapters were actually Glow-Worm, which especially

explains Ashley's likely Kenzie induced spike. For Rain, we see him dip during the Fallen raid arc, where his dark past is revealed and he has to go through A LOT of shit, and then trend positive for the rest of the story. Similarly, we see Ashley get described with mixed to negative sentiments early on, before Victoria figures her out and before Ashley mellows out. We also see a dip later on, likely due to the introduction of Ashley's clone sister, 'Ashley Black/Slashley'. Finally, Chris bounces all over the place for most of the story, fitting for his moniker of 'Cryptid'. At the end, when he decides to leave the team and join Amy on Earth Shin, we see the sentiment stay relatively negative.

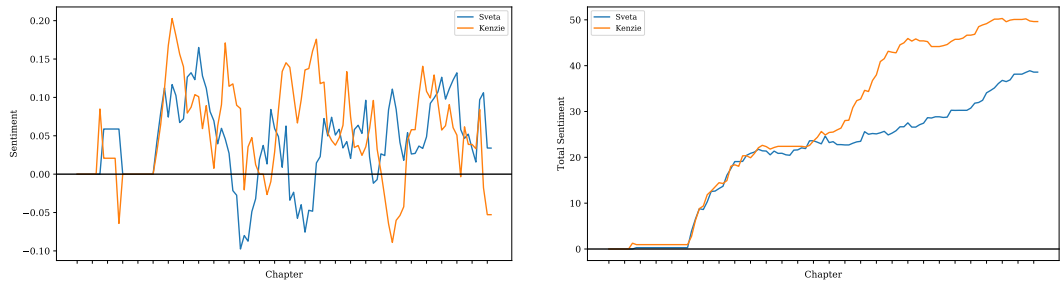


(a) Mean Smoothed Chapter Scores for Ashley/Rain/Chris

(b) Cumulative Sentiment Scores for Ashley/Rain/Chris

Figure 3: Sentiment Scores for Ashley/Rain/Chris

Next, we'll look at the two members of Breakthrough who Victoria holds in the most positive light; the kid tinker Kenzie and the best friend/'team mom' Sveta. Looking at Fig. 4b, we see that both are viewed in an overwhelmingly positive light. Looking at Fig 4a, we see that Kenzie is almost never described negatively, with only three real exceptions; early negative mentions during Glow-Worm, likely from Chris and/or Mayday, random points during the midpoint of the story, and when she was brainwashed by Goddess during the prison incident. Looking at Sveta's graph we see similar dips during the story's midsection; looking closer, these seem to be during the end part of the tense Fallen incident and during Breakthrough's rough TV appearance, where the hosts (and not the main character, Victoria) were negative towards pretty much the entire team.



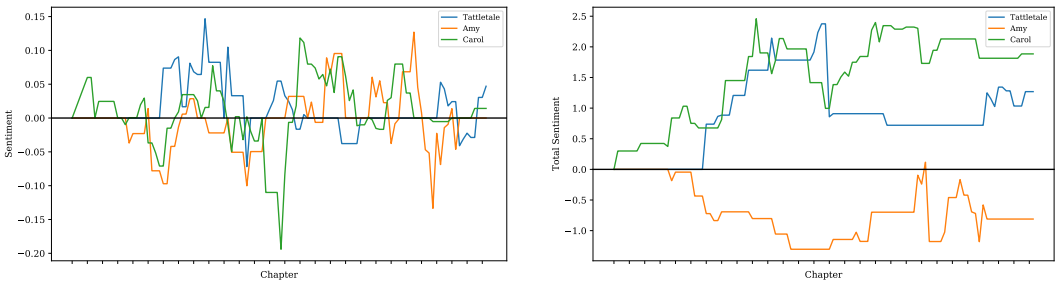
(a) Mean Smoothed Chapter Scores for Sveta/Kenzie

(b) Cumulative Sentiment Scores for Sveta/Kenzie

Figure 4: Sentiment Scores for Sveta/Kenzie

Finally, we move on to a trio of morally grey but antagonistic characters (especially in the

eyes of our main character); the villainous mastermind Tattletale, Victoria's mother (who needs to 'Mom the fuck up') Carol, and the source of most of Victoria's nightmares, her sister Amy. It's safe to say that these are the main characters three least favorite people, while also not being clearly evil or villains (despite this being Tattletale's technical position). As side characters, we expect to see less change among these three than the other five, as they aren't usually focused on, which is definitely true given the shape and magnitude of the sentiment totals, as seen in Fig. 5b. Unsurprisingly, Amy's sentiment is almost always below water, with the only exception being the chapter (8.12) where Victoria defends her during the national television incident (the other spike is due to others discussing Amy or Sveta referring to herself as a 'kid sister'). With regards to Tattletale, we that sentiments regarding her are actually fairly positive (best friends incoming?), though we do see a sharp positive increase followed by a precipitous drop near the middle of the graph; this is in the aftermath of the Fallen raid where first Tattletale offers to help our heroes, with the fall being when they actually talk to her (and roast her a bit) in person. We also see that Carol sits a bit above water too, partly due to Victoria's mixed but somewhat positive feelings towards her and the positive views held by other characters, namely Natalie. The large negative spikes, seen in Fig. 5a, are largely spurious, due to one off hand mention and the discussion of other mothers. This is the problem with detecting Carol by looking for the terms mom & mother; the terms are quite ambiguous.



(a) Mean Smoothed Chapter Scores for Amy/Carol/Tattletale

(b) Cumulative Sentiment Scores for Amy/Carol/Tattletale

Figure 5: Sentiment Scores for Amy/Carol/Tattletale

4 Conclusion

So what was the point of all of this? Well really, it was just to play around with NLP and sentiment analysis, to get a feel for some of what they can do and bit of how they work. We've definitely run into some limitations of the commonly available methods and of my own knowledge and abilities. One obvious issue is the ambiguity of the text. In general, the idea was to get the point of view character's perspective, but I've made no real effort to make sure only her thoughts and words were analyzed. Furthermore, Amy & Carol are often referred to by general nouns that may refer to others as well. Additionally, many references may be made using pronouns (he, him, she, her, etc.), and I have yet to really look into whether or not there is an easy way to automatically detect what a pronoun is pointing at. Outside of correct sentence selection, the sentiment analyzers are going to be

far from perfect; a quick Quora search shows that even custom models top out at 70 - 80 % accuracy and I'm using models trained on very different text sources. For fun, I may try to create my own simple Naive Bayes classifier to see how the results compare, but I doubt it will improve things on this front. And of course, there are always the usual odds and ends; cleaning up code, prettying graphs and their labels, etc. Regardless of all this, I hope whomever is reading this enjoyed it in some capacity, either by learning something or just enjoying some oscillating lines.