Heroes and Villains: What A.I. Can Tell Us About Movies

Brahm Capoor
Department of Symbolic Systems
brahm@stanford.edu

Michael Troute
Department of Computer Science
mtroute@stanford.edu

Varun Nambikrishnan
Department of Computer Science
varun14@stanford.edu

Abstract

TODO

1 Introduction

When faced with a new movie, an audience asks itself several questions. Who is the protagonist? Who is the antagonist? How are the supporting characters aligned? Does the plot develop according to a certain archetype? These questions are often answered to the tune of raging debate, coloured by the biases of the audience and how they responded to the movie.

Sharing a common love of film, we were interested in bringing A.I. techniques to bear on these quests and examining – given the screenplay of a film – what sorts of insights we might be able to glean into how the script works.

We use distinct techniques to answer three questions. First, we use supervised learning to try and identify the Protagonist and Antagonist of a film. Next, we cluster the characters of a film based on various sentiment scores and their relationship to the protagonist to identify different factions within the film. Finally, we find the emotional trajectories of films as the plot develops and cluster those trajectories to identify archetypical stories.

2 Literature Review

3 Task Definition

As the introduction outlines, we tackle 3 distinct problems.

3.1 Detecting Protagonists and Antagonists

First, given a film's screenplay as input, we apply supervised learning and sentiment analysis to accurately predict the identity of both the protagonist and the antagonist. We define a film's protagonist loosely as the character around whom the film revolves, and with whom the audience most closely identifies; we define the antagonist as the character who stands in greatest opposition to the protagonist's goals.

3.2 Character Clustering

Second, we apply unsupervised learning and sentiment analysis to (a) group characters within the same film into certain factions or aligned groups, and (b) group characters across our entire dataset to identify common roles roles in film (e.g. mentor or main relationship character).

3.3 Shapes of Stories

Third and finally, we apply sentiment analysis to determine emotional trajectories for each of the films in our dataset; given these trajectories, we apply unsupervised learning to determine a set of archetypical stories. Once we've determined these archetypes, given a previously unseen film, we can quickly determine which archetype we believe it belongs too.

4 Data and Infrastructure

We use the Cornell Movie-Dialogs Corpus, [1] which contains 220,579 lines of dialogue spoken between 9,035 characters in 617 movies. The corpus also includes metadata such as film genre, character gender, and character position in movie credits.

We hand label protagonists and antagonists for a subset of the films in the dataset.

[1] available at https://www.cs.cornell.edu/~cristian/Cornell_Movie-Dialogs_Corpus.html

5 Approach

5.1 Feature Extraction

In order to generate numerical features from our dataset, we built two feature extractors that comb through our dataset to make a feature vector for a particular character.

5.2 Detecting Protagonists and Antagonists

As indicated by our feature extractors, we consider the identification of the protagonist and antagonist of a film separate problems.

5.2.1 Oracle and Baseline

Our Oracle is simply a human identifying both the protagonist and antagonist of a film, and gets 100% accuracy in both protagonist and antagonist identification.

Our baseline algorithm counts the number of times each character speaks in the film, and then considers the character with the most lines to be the protagonist, and the character with the second most lines to be the antagonist. Surprisingly, this approach is reasonably successful at identifying protagonists. It is able to identify the antagonist for 88% of films, failing mostly when the protagonist is mostly silent or the protagonists are a group of individuals, none of whom alone speaks a lot. However, the algorithm is far less successful at identifying antagonists of films, and is only correct 28% of the time. In comparison to our oracle, our baseline only gets 24% of protagonist/antagonist predictions correct.

This suggests to us that identifying antagonists is a significantly more difficult task than identifying protagonists. Given our domain knowledge, this makes sense: it is generally much easier to universally agree on the protagonist of a movie than the antagonist.

5.2.2 Protagonist Feature Extractor

Our feature extractor used when identifying protagonists of films finds the following features for every character in a film:

1. Number of lines spoken
2. Number of words spoken
3. Number of times spoken to
4. Whether any part of the character's name is in the title of the film
5. The position of the character in the film's credits
6. Whether the character is male
7. Whether the character is female

These features were fairly low-level (i.e. without taking into account semantic or sentimental knowledge of the film), but we found they led to high performance in our prediction. Note that Features 4, 6 and 7 are indicator features that are 1 when their corresponding condition is true and 0 otherwise. Note as well that features 6 and 7 are separated to accommodate characters that are not human and have no gender.

5.2.3 Antagonist Feature Extractor

In order to extract features for antagonist detection, we use all the same features as we do for protagonist detection, but given our identification of a protagonist, we are also able to use two higher-level features, namely:

1. Average sentiment of the lines they speak
2. Average sentiment of the lines the protagonist speaks towards them.

Sentiment information is found using NLTK's pre-trained 'Vader' sentiment analysis module.

5.2.4 Detecting Protagonists

To identify protagonists, we use a training set consisting of $< x, y >$ pairs such that each x is the feature vector produced by the Protagonist Feature Extractor (see section 5.2.1) for a particular character and y is an indicator variable indicating whether the character is the protagonist of their film. The training set represents characters from 75% of the films in our corpus. We then train a Logistic Regression model on this set. To identify the protagonist of a film in our testing set, we input feature vectors of all the characters in the film, and the model outputs the probability that each character is the protagonist. The character with the highest probability is selected as the protagonist.

5.2.5 Detecting Antagonists

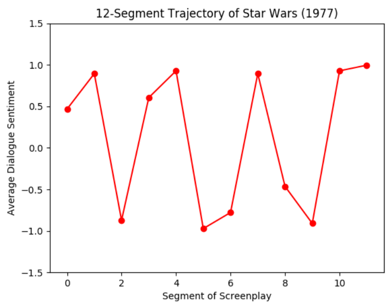
To identify antagonists we use a training set consisting of $< x, y >$ pairs such that each x is the feature vector produced by the Antagonist Feature Extractor (see section 5.2.2) for a particular character and y is an indicator variable indicating whether the character is the antagonist of their film. To find high-level features in the Antagonist Feature Extractor (specifically, the sentiment of the protagonist towards a particular character), we use the protagonist identified previously in section 5.3.1. This training set also represents characters from 75% of the films in our corpus. Similarly to Protagonist Detection, we then train a Logistic Regression Model on our training set and identify antagonists of films in the test set by selecting the character with the highest probability of being an antagonist.

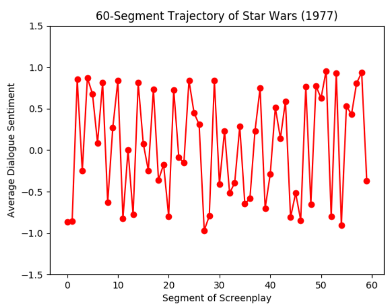
5.3 Character Clustering

5.4 Shapes of Stories

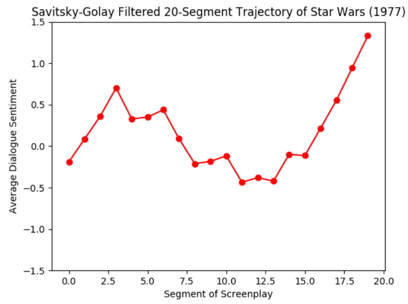
In the third phase of our project, we use unsupervised learning to find patterns in the emotional trajectories of the scripts in our dataset.

To obtain an emotional trajectory for a film, we first concatenate all the tokens of all the lines spoken in the film into a single ordered list. We then split this list into N equal-length "segments". Using NLTK's pre-trained 'Vader' sentiment analysis module, we can then determine a sentiment score for each of these segments; taken together, these scores define a rough emotional trajectory for the film. The following plots show the emotional trajectory of the original *Star Wars* (1977) for 12 and 60 segments respectively.

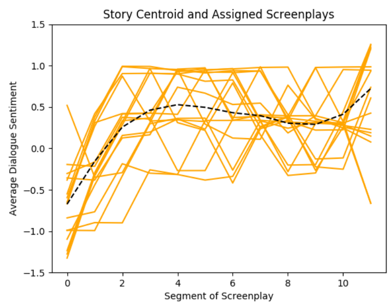


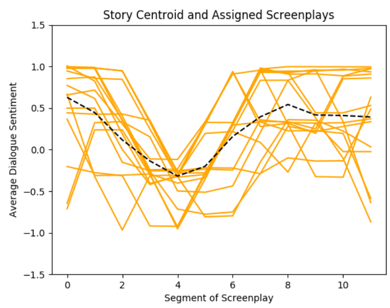


It is immediately clear that both of these figures, especially the second, are extremely noisy; as N increases, individual segments are become smaller and smaller, and tend to produce highly volatile trajectories. To address this, we apply a Savitsky-Golay filter to the trajectories. This produce the following smoothed trajectories when applied to *Star Wars*' 20-segment trajectory.



Once we have obtained smoothed trajectories for each of the films in our dataset, we can perform K-means clustering on the trajectories (as points in N-dimensional space) in an attempt to cluster stories that follow the same arc or have the same general shape. The following figures show two of the centroids, and the stories assigned to them, learned with $K = 7$ and $N = 12$.





We call the first of these story archetypes the Hero's Journey, and the second the Redemption Story. It's easy to read such meaning into the story centroids produced by any combination of K and N, so choosing these values carefully is paramount, as will be discussed further in the next section.

6 Experiments and Error Analysis

6.1 Shapes of Stories

Distance between circles is topological similarity