Natural Script Processing: An attempt to automate greenlighting movie scripts for production and character casting

Avijeet Kartikay, Sean Pili and Pedro Uria Rodriguez

Natural Language Processing

The George Washington University

Natural Script Processing: An attempt to automate greenlighting movie scripts for production and character casting

Introduction

Motivation. Roughly 50,000 screenplays are registered with the Writer's Guild of America each year. Based on their experience of producing movies that are a hit, producers and movie studios manually read through scripts that they come across themselves or by word of mouth of their colleagues. A fair portion of the registered scripts rots away without ever being read. Reading approximately 50,000 scripts, handpicking a good story, producing the film in time to meet the seasonal box office demands is quite the arduous task and an interesting problem to solve. While the financial success of a film cannot be determined solely by the script, there are many stages that need to be optimized. Once a script is selected, the creative decisions of cast and crew, the trailers, the awareness generated around the film and many other such factors fuel the financial success of a film. These factors can only be addressed once a few quality scripts are selected. As a primary filter to predicting the success, we will attempt to predict the success of a film using NLP techniques.

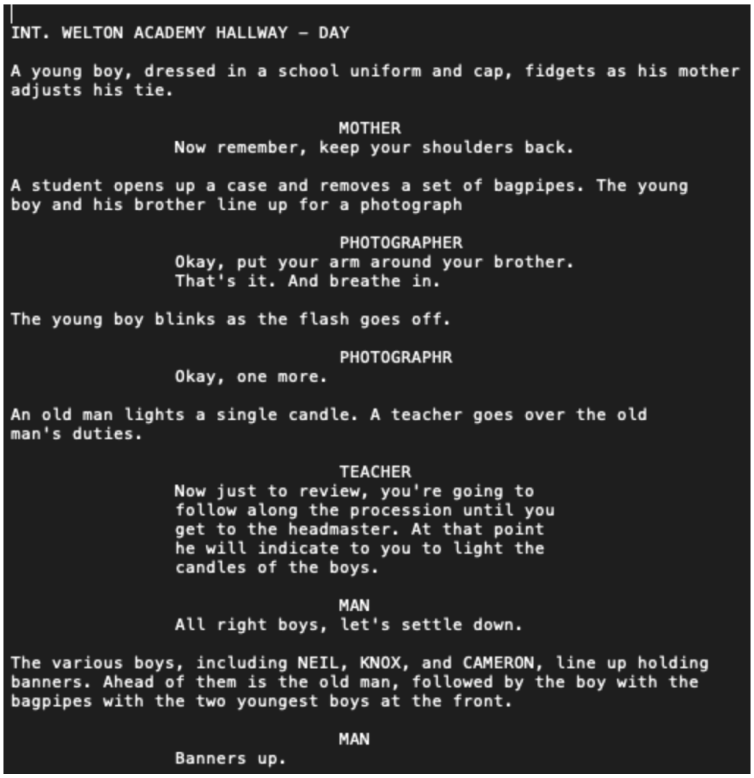
Upon investigating the dataset, we uncovered another interesting problem that can be solved: the problem of casting. We will create clusters of character types using their individual dialogues. Casting the right character for the right role is another crucial problem we can aim to address with the data available to us. An actor is picked based on his ability to capture the characteristics of the character he plays. If the performance of the actors is incoherent to the story their characters explore, the audience is left cinematically unsatisfied and thereby rendering the film a commercial failure.

Previous Work. To predict movie profitability, Jehoshua Eliashberg, Sam K. Hui, and Z. John Zhang trained a model using features extracted from a corpus of 281 spoilers, or fleshed-out plot summaries in their paper: *From Storyline to Box Office: A New approach for Green-LIghting Movie Scripts*. According to an approach proposed by Starling David Hunter, Susan Smith, and Saba Singh, the opening week box office is predicted using data extracted from the text. Using data readily on the Box office Mojo website, the financial performance of the movie is used in tandem with multi-morphemic compounds (MMC) feature extracted from text data. These features capture the relationships between characters and use the richness of these relationships the target variable is predicted. To generate more believable character dialogue for video-games, Marilyn A. Walker, Grace I. Lin and Jennifer E. Sawyer used a variety sophisticated textual features to represent characters from certain genres of film which, when fed into dialogue generator was able to convince judges that the dialogue generated sounded like it came from a particular fictional character (e.g. Indiana Jones) in their paper: An Annotated Corpus of Film

Dialogue for Learning and Characterizing Character Style.

Data

Source. A dataset is not readily available to us on Kaggle or other such data repositories. All the movie scripts were downloaded from https://www.imsdb.com in text format and the movie budge data is available on https://www.the-numbers.com/movie/budgets/all. Once downloaded, the scripts more or less follow a standard format as shown on the next page. The character name is in capitals indented to the centre of the page, and the text is also centered below the character name. We aim to extract feature from the script per character and store it in a JSON format file.



```
INT. WELTON ACADEMY HALLWAY - DAY

A young boy, dressed in a school uniform and cap, fidgets as his mother
adjusts his tie.

                                MOTHER
                Now remember, keep your shoulders back.

A student opens up a case and removes a set of bagpipes. The young
boy and his brother line up for a photograph

                                PHOTOGRAPHER
                Okay, put your arm around your brother.
                That's it. And breathe in.

The young boy blinks as the flash goes off.

                                PHOTOGRAPHR
                Okay, one more.

An old man lights a single candle. A teacher goes over the old
man's duties.

                                TEACHER
                Now just to review, you're going to
                follow along the procession until you
                get to the headmaster. At that point
                he will indicate to you to light the
                candles of the boys.

                                MAN
                All right boys, let's settle down.

The various boys, including NEIL, KNOX, and CAMERON, line up holding
banners. Ahead of them is the old man, followed by the boy with the
bagpipes with the two youngest boys at the front.

                                MAN
                Banners up.
```

Processing.

- Most of our scripts followed the standard format. Those who did not were identified and removed through various methods.
- Identify characters names:
  - Count the frequency of each unique sentence.
  - Get a list of potential characters: frequency > 5.

  - Select the top 5 characters with the most dialogue (need next bullet point for this final step).

- Extract dialogue for each potential character:
  - Loop over each sentence.
  - If the whole sentence is uppercase (filtering some stuff).
    * We get all the sentences below it with the same level of indentation.
    * This level is identified as the indentation of the first sentence that does not contain "(<action description>)".

Evaluation.

- We compared our list of characters with the ones listed on IMDb and obtained 0.82 accuracy. This is under evaluating as some names might not match.
- While performing clustering and extracting different features, we identified edge cases that were accounted for in the code.
- We also identified most of the movies that did not follow the standard format and removed them from our dataset (about 40/700).
- We saved the dialogues for each movie with .json format: https://raw.githubusercontent.com/PedroUria/NLP-Movie_Scripts/master/diag_jsons/Big-Lebowski%2C-The_script.json
- We also distinguished lines by keeping a count [i].

Feature Extraction. After getting the characters, we extracted many features for the top 5 characters:

- Each character's overall polarity
- Cosine similarity between each of the top 5 characters in a film
- Number of times characters mention each other (and polarity of sentences where they are mentioned)
- Percentage of sentences a character speaks in the passive voice
- Fleisch Kincaid Readability Score
- Percentage of sentences in which a character uses a curse word.

- Average number of unique words used per sentence
- Percentage of sentences in which a character uses a stop word
- Percentage of sentences in which a character uses emphasizer or softener hedges (maybe, definitely, etc.)
- Percentage of sentences in which a character uses a word that may indicate if they are giving an opinion (i.e "in my opinion", "to me", "I think" etc.)

We also aggregated some of these features and extracted others more relevant for clustering, which will be discussed in the next section.

Clustering

Overview. We attempted to use unsupervised learning for two purposes: 1. To identify certain character types across our corpus of character dialogues and 2. To identify themes or genres in movies. We then hoped to use these clusterings as features in the model that we developed, but as described below, our unsupervised learning task proved to be more daunting than we anticipated.

LSA. First, we tried using *Latent Semantic Analysis* on our corpus of character dialogues (i.e. each document was the dialogue of one of the top 5 characters of one of the films in our corpus) after doing a limited amount of textual preprocessing (tokenization and stop word removal). Unfortunately, some of the most important words in the topics LSA extracted included numbers and character names, as those can be relatively unique, but not informative as to the genre of a film or the personality of a given character. After removing the names of all of the characters in our corpus and removing all numbers, we also decided to filter out all POS that did not include verbs, nouns, adverbs or adjectives (see our appendix for the full list of POS tags we included using NLTK). Even after aggressive filtering, our results were still not very promising. Using stemming and lemmatization did not improve results. We then decided to run LSA on bigrams, trigrams and quad-grams while including stop words with the idea that we could look for phrases used by different types of characters. We used two approaches to this. The first was a brute force method of adding underscores between words. Below we show a bigrams example:

"The man walked" $\Rightarrow$ ["START_the", "the_man", "man_walked", "walked_END"]

We also used the ngram_range parameter of sklearn's `CountVectorizer` to search to extract features that ranged from unigrams to quad-grams. Unfortunately, our results were not any more informative than when we looked solely at unigrams. In future iterations of this project, we hope to do more hyperparameter tuning with sklearn's `CountVectorizer`, such as the number of features to extract and the n_gram range.

DBScan. Although it is possible that one cannot use NLP to extract the genre or character types from character dialogues, we know that topic modeling is not the only way to find this kind of information. We considered using distance-based clustering methods such as k-means and agglomerative clustering, but those methods come with the same challenge as LSA, we have to define the number of clusterings that are allegedly inside of our data, which is troubling as we do not know the distinct number of character types in our dialogue corpus. This is why we decided to focus DBScan, a density based clustering technique, that does not require one to predetermine the number of clusters in their dataset. One does, however need to specify the minimum number of points within needed to be density reachable by a given point within a pre-specified radius for it to be considered a core point of a cluster.

We then extracted the following features for each of the top 5 characters in our movies for use in DBscan: the average number of opinion words per sentence, average number of filler words (i.e verbal pauses such as "um", "uhhhhhh" etc.), average hedge words per sentence (i.e. "maybe", "definitely", etc.), average stop-words per sentence, average curse-words per sentence, Fleisch Kincaid readability score, the compound polarity of each sentence and average unique words per sentence. Finding the best hyperparameters took some trial and error. Initially, we tried using a plethora of hyperparameter combinations and looking at the results of the ones with the highest silhouette metric, but those results were not helpful: the data was broken into two large clusters with a large variety of characters, and a large number of noise points. We then decided to investigate hyperparameter combinations that yielded a large number of clusters. Our most promising hyperparameter combination came when we used a radius parameter of .075 and minimum number of points as 3.

The first cluster contained Mitchell from *The Damned United*, Van Houten from *The Fault in our Stars* and Captain Idaho from *The Postman*, all of whom were harsh gruff characters. The second cluster included Jigsaw from *Saw*, Dr. Waldman from *Frankenstein*, Razor from *The Matrix Reloaded*, a TV anchor from *Signs* and the mirror on the wall from *Shrek*. Although the mirror on the wall from *Shrek* appears to be an anomaly in this cluster, Jigsaw and Razor are very similar characters. Although the TV anchor from *Signs* is not seen as a villain, they give exposition, just as Jigsaw and Razor do. Although Dr. Waldman was not a villain in the frankenstein movies, we found it interesting that he, as well as Jigsaw and the TV anchor, were from horror/thriller movies. The third cluster included Charlie Frost from *2012*, Gibbs from *Pirates of the Caribbean, Dead Man's Chest* and Susie from *Grand Theft Parsons*. Charlie Frost and Gibbs both frequently provide exposition. The fourth cluster included Wendy from *The Ice Queen*, Reece from *The Iron Lady*, Malcom's voice from *Malcom X* and Captain Pike from *Star Trek*. Captain Pike and Wendy appear to be very similar characters. They both come across as "smart alecks" and curse frequently. Malcom also swears often, but his tone is slightly different as he narrates the movie. Reece can

be seen as a smart aleck as well, but he does not curse as often as the other members of the cluster.

The fifth cluster included Sam from *Garden State*, Starke from *Ghost Rider* and Racer X from *Speed Racer*. Although Sam is different from Starke and Racer X, there are some similarities between Starke and Racer X. Both of them ride motorcycles and can be very violent, though racer is closer to a hero than a villain, which Starke is. The sixth cluster includes Humanz brother from *Gamer*, Brody from *Indiana Jones and the Last Crusade*, Miller from *Repo-Man* and Morgan from *Tombstone*. It also included a set of stage directions from *From here to Eternity*, which we did not manage to delete. Miller and Morgan appear to be somewhat similar as they both appear to briefly speak about the meaning of life, but other than that, we were not able to pick up meaningful similarities between the other characters in the cluster. The seventh, 8th and 9th clusters included too many characters to fit into one category, and the algorithm produced 292 noise points, meaning that there were 292 characters determined to be outliers (or non-density reachable by the other clusters.

Predicting Success

Overview. After getting promising results with our clustering approach, we proceeded to use the same features to predict whether a movie was going to be successful or not. Recalling from the *Data* section, we scraped the scripts but also their budget and box office data. With these data points, we calculated the ROI (%) (Return of Investment) for each movie. Due to the opaque and creative accounting methods used by the film industry (*Hollywood Accounting*), we decided to discretize this target into *Successful* (ROI > 0) and *Not Successful* (ROI < 0). After doing so, the final dataset was ready for this binary classification problem.

EDA. The first thing we did was looking at the class labels balance: about 82% of our movies were successful. Thus, our models would need to at least beat this score. Next, we computed the *point biserial correlation coefficient* between each of our features and our target. Unfortunately, none of them had a correlation higher than 0.1 with it. This included our raw features per character as well as our aggregated features. Although discouraging, we still hoped for some sort of pattern to be formed by a combination of these features.

Modeling. After doing this, we first took the highest correlated features and run a grid search with various models, including *Logistic Regression*, *Decision Tree*, *Random Forest*, *KNN*, *Gaussian Naive Bayes* and *AdaBoost*, and with various hyperparameters, looking to achieve the best mean cross-validated accuracy on our training set. However, our models could not beat the no-information rate (≈ 0.82), and when looking at the actual predicted targets, these were all 1s, i.e, our models were not learning anything,

just predicting the most abundant class labels. We aimed to lower the amount of false positives, but this resulted in many many more false negatives and an overall much lower accuracy. For both accuracy and precision, we tried different combinations of features, undersampling and oversampling, and many other things like changing the *success* threshold, discretizing the target in more than two categories, and even switching to Domestic (US) ROI from Worldwide ROI. These attempts were all for nothing, so we are pretty sure that there is no underlying function between linguistic features and profitability.

Conclusions

A film is considered to be successful if its ROI is greater than 0. This indicates that the movie broke even and managed to generate revenue. From a financial standpoint, if a script generates revenue, it can be considered to be a successful one. Our models were not able to identify the underlying function that maps the language features a script to its financial success, as the model predicts all the scripts to belong to the majority class, which is worse than a random guess.

No matter how aggressively we filtered the character dialogues, we were not able to find topics that were representative of genres or character types with LSA. We think that part of this is because we revealed that a few non-character dialogues or camera directions still remained in our corpus. If we took the time to manually remove them, our results might improve. Since were able to find clusters of surprisingly similar characters with relatively basic features using DBscan, we believe that it would be worthwhile to extract more sophisticated linguistic features from the character corpus in order to find better clusterings. For example, Walker et al. extracted how often characters use tag questions, how frequently characters restate prepositions, how frequently characters use first person pronouns and the percentage of times characters merge multiple propositions (instead of leaving them in separate sentences).

References

[1] Eliashberg, J., Hui, S. K., & Zhang, Z. J. , *Deep Learning Techniques for Music Generation - A Survey*, 2007.

[2] Hunter, I. I. I., David, S., Smith, S., & Singh, S, *Predicting box office from the screenplay: A text analytical approach*, Journal of Screenwriting

[3] Walker, Marilyn A., Grace I. Lin, and Jennifer Sawyer, *An Annotated Corpus of Film Dialogue for Learning and Characterizing Character Style*, 2012.

[4] GWU NLP Class Material