CS 230: Film Success Prediction Using NLP Techniques

Joshua A Gross, William C Roberson, J Bay Foley-Cox

February 2021

Abstract

While Natural Language Processing has been applied to the film industry already - and with some success - Deep Learning techniques have not previously been used to study the predictive capabilities of movie scripts on revenue and reviews, due to the difficulty of long-form language processing. We demonstrate a Natural Language Processing (NLP) model which takes encoded data from film scripts to predict box office revenue and critical reviews. Our model takes segments of long form text documents as input, in addition to some structured data, and returns real number values predicting likely revenues and review scores. We found dramatic predictive capabilities from movie scripts on revenue and reviews, in particular when compared to structured data alone.

1 Related work

The most analogous work to ours comes from a Carnegie Mellon study [1] which used short text excerpts from written critical reviews to predict box office revenue. As mentioned previously, a critical difference between this work and ours is the size of a singular input datum.

2 Dataset and Features

Our dataset may be separated into two major parts: a set of structured categorical and numerical data retrieved from IMDb, and a set of scripts from which we generate word frequency vectors and scene description vectors.

Our data has come from two main sources: the Internet Movie Database (IMDb) and the The Internet Movie Script Database (IMSDb). Our first step in data acquisition was to modify a Python script in order to pull all movie scripts from the IMSDb. We have stored this code in "scrape.py" and the HTML file for each script in "scripts." This file was also responsible for gathering the IMDb IDs associated with each movie script, which we used to relate all the structured data collected from different sources (since different movies can have the same title). These IDs were stored in "script-ids.txt." We used these IDs in "metadata-extraction.ipynb" to generate "movie-metadata-updated.csv," the current source of structured data which we used to train our model. This metadata includes categorical data and numerical data. The categorical data has the film's languages, genres, and the quarter the film was released. The numerical data consists of the film's budget, runtime, and release year. These are used to predict IMDb ratings and Worldwide Gross Box Office Revenues.

The other piece of data is from the movie scripts. In an effort to facilitate our long-form language processing, we performed a frequency analysis on each of our movie scripts and returned a vector of the top 250 most common words in the script. We encoded these words as one-hot vectors using the

total list of words (all words from all scripts) to match indices. We fed these vectors directly into our model, which then used a pre-trained (GloVe) word embedding matrix to reduce them to lower dimensional vectors.

We also attempted to learn based on the scene directions in the scripts (the bolded text describing settings and actions), sampled down to contain a reasonable number of words (150). This data was collected by extracting the scene descriptions from each script (the contents of $< b >$ tags in the script HTML), keeping the first 150 words, and arranging those words into a matrix of ordered one-hot vectors as we did for word frequency.

3 Model and Approach

We approached the problem of predicting movie success from scripts by building a system which could be used to generate a variety of models, each approaching the task in different ways. Each of these respective models could predict both movie revenue and critical reviews on a real number valued 1-10 scale.

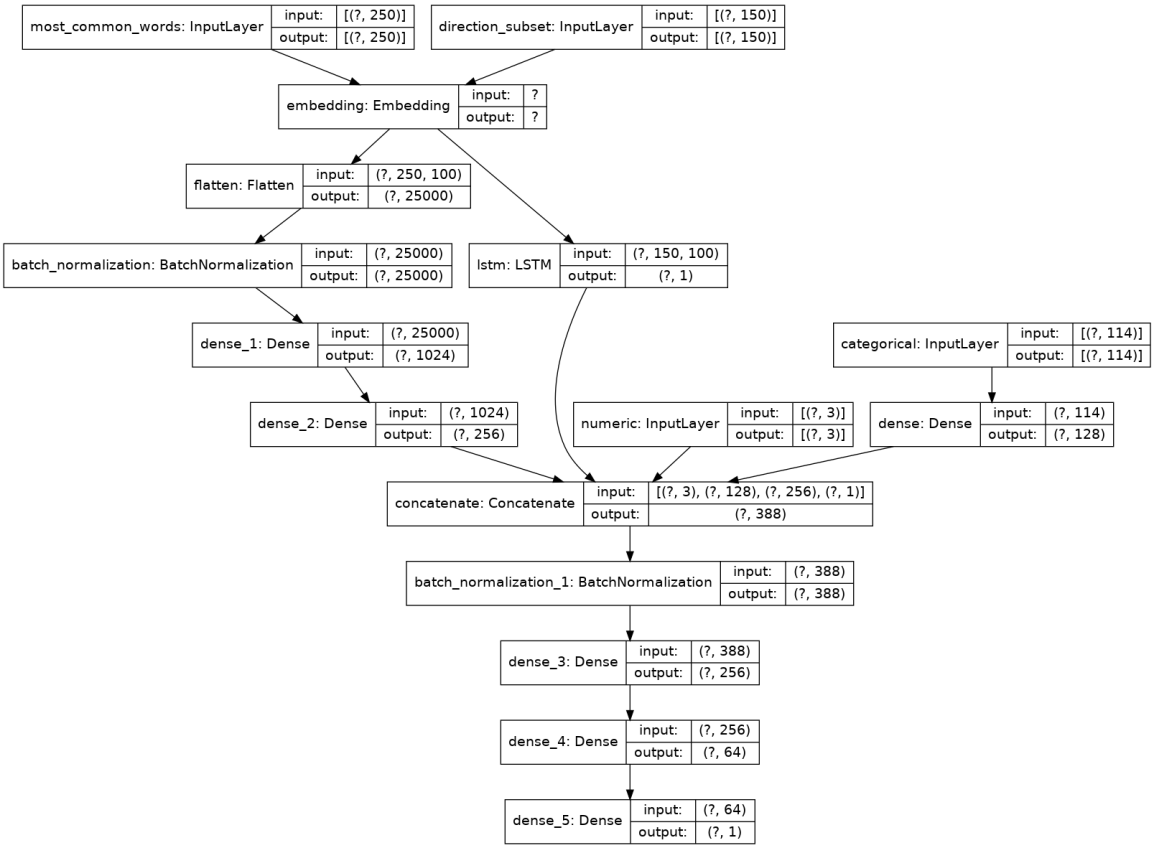
These models were built using Tensorflow and the Keras Functional API. This allowed us to explore making predictions based on different combinations of inputs such as numeric structured data, categorical structured data, word frequency information, and samples of scene description text, all targeting either revenues or reviews.

Our model is oriented in such a way that it can be trained with any combination of these inputs depending on the supplied command line arguments, so we can easily develop and test many model variants to see which inputs are most useful to the model. All inputs included in a particular run of the model are first preprocessed into individual feature vectors, then concatenated and run through a BatchNorm layer followed by a fully connected network to produce a numeric output.
We produce these features as follows:
For the numeric structured data, we return the input data directly.
For the categorical structured data, we use a dense layer without bias to serve as a trainable embedding layer which returns a 128 dimensional embedding of the data.
We apply the 100 dimension GloVe.6B word embeddings for feature extraction on the frequency data (a list of the 250 most frequent words in each script) and then run this output through two dense layers to further reduce dimension. Our goal with this analysis is to perform a quasi-sentiment analysis over these popular words by feeding their embeddings into the model in order of frequency.
For the script direction data we use the same embedding as before and then run the sequence through an LSTM layer producing a single output.

Depending on the command line arguments, the model can also be trained to predict either movie revenues or reviews. We chose to take this approach as opposed to building a combined model to avoid adding an additional hyperparameter to tune (which would be needed to determine the relative weights of rating and revenue predictions on backpropagation). In both cases we use the same architecture except that the output sigmoid has a different range, 1-10 for reviews and 0-23 for log revenues. We discuss our choice to predict the natural logarithm of revenues rather than revenues in Section 5.

We optimize the models using the Adam optimization algorithm with a mean squared error loss function.

Figure 1: Model Topology with all inputs enabled:



4 Significant Challenges

This project contained several significant challenges. The largest conceptual challenge was addressing the difficulty of long-form NLP tasks. As mentioned previously, we addressed this problem by only considering word frequency, which allowed a simpler network that could handle some information from the script without being overwhelmed by its size. We also tried feeding a small portion of the bold scene description text into an LSTM module to see if this could help with predictions, although this approach had limited success.

Many of the other challenges were related to the difficulty of gathering and cleaning data. We had to collect the data from different sources, some of which only had a limited amount of open source information. Thus our data had many missing values, and initially we did not even have enough revenues to train the model. We went through our dataset by hand to correct this, and filled in values that were missing or incorrect. We were surprised by the extremely negative initial results which stemmed from missing data, and spent much of our work cleaning and resourcing our data inputs.

It was also difficult to overcome the limited size of our dataset, considering the complexity of the task. We responded to this problem by first sorting through our dataset by hand to clean the values that were obviously incorrect, and setting incorrect or unfindable values to the average for

Table 1: Model Mean Squared Errors

	Log Revenue MSE	Score MSE
Structured and Frequency	1.999	0.499
Structured, Frequency, and Direction	2.085	0.560
Structured	3.299	1.064
Frequency	3.606	0.958

Table 2: Hyperparameter Choices

Parameter	Magnitude
Learning Rate	0.0002
L2 Lambda	0.005
Batch Size	500
Epochs	4000

each category of numerical data. We also implemented the features of the model described in Model and Approach that limit overfitting. However, neither of these techniques were able to fully cope with the limited size of our dataset, so the model suffers from a higher variance than we could have achieved if a much larger dataset had been available.

5 Experiments and Results

Our project had several goals. We not only aimed to accurately predict film success through NLP techniques, but to demonstrate the marginal benefit of such techniques as compared to simpler analyses. We used benchmarks of 0.88 and 3.8 MSE as the respective benchmarks for mean squared error on Review Accuracy and ln(Worldwide Gross Revenue) Accuracy, the latter of which is an accuracy measure for the natural log of Worldwide Gross Revenue. We derived these benchmarks by considering the MSE which would be produced from guessing mean values for each of our relevant statistics. As demonstrated in Table 2, our models became dramatically more accurate through the use of frequency analysis with pretrained (GloVe) word embeddings.

Our experimentation largely fell into two categories: architectural adjustments and hyperparameter tuning. The latter issue merely was a process of trial and error; we have included our hyperparameter choices in Table 2.

Overfitting and implementing regularization were major challenges we faced in constructing our model. Our early models had development set error an order of magnitude higher than training set error, which we initially addressed only through batch normalization. We later attempted implementing L2-regularization which was quite successful, but dropout regularization (with keep-Percentage ranging from 0.05 to 0.3) had little effect on our issues. We also found that decreasing the depth of our model dramatically improved our overfitting due to training set memorization.

We also had initial difficulty learning to fit revenue data. This was partly due to its nonlinear distribution, which led us to try predicting ln(revenue), the natural log of the revenue. This alteration immediately led to success, with our MSE falling far beneath our benchmark as explained above.

6 Conclusions and Future Work

Our results show that adding frequency analysis of unstructured script data to the simpler, structured data model significantly improved the model's predictive ability on both revenues and ratings. As shown in Table 1, the log revenue MSE improved by a factor of 7/4 and the rating MSE improved by a factor of 2 after introducing frequency analysis, while scene direction information slightly hurt model performance. This demonstrates that the script does contain additional information that is either not present in or difficult to extract from the metadata.

However, with more time and resources we would have liked to explore variations on our architecture which might have led to less overfitting. In addition, the variety of sources from which we drew our revenue data could have led to data mismatch issues. This could be addressed by manually cleaning the data more thoroughly, finding better open-source databases, or changing the scope of the project to look at another type of media (such as TV episodes) where data is more prevalent. We are however very satisfied with our resulting program and our contributions to the pool of large scale NLP research.

Although the recurrent neural network component of the project ended up slightly reducing model performance, we believe that significant improvements could be made in this area. Because many of the scripts were low quality and user generated, there were many typos which prevented us from using a pretrained embedding for many words in some script direction text. With so many words missing and limited training data we were not able to achieve useful results. However, with a larger quantity of higher quality data, we would be able to achieve more success and then experiment with different models and different subsets of the scripts to see which work best for predicting revenues and reviews.

Additionally, we only tried two different forms of data from the script: word frequency and scene directions. More research could be done in deciding what information from a script contains the most information about film success. Examples could include character analysis on the script's proper nouns, many-to-one analysis on a random sampling of words, many-to-one analysis on the first few words from each scene, etc.

7 References

[1] Joshi, Mahesh, and Dipanjan Das. "Movie Reviews and Revenues: An Experiment in Text Regression." Https://Www.aclweb.org/Anthology/N10-1038.Pdf.

[2] Mestyán, Márton, et al. "Early Prediction of Movie Box Office Success Based on Wikipedia Activity Big Data." PLOS ONE, Public Library of Science, journals.plos.org/plosone/article?id=10.1371

[3] Zhao, Yao, et al. *SEAL: Segment-Wise Extractive-Abstractive Long-Form Text Summarization.* arxiv.org/pdf/2006.10213.pdf.

[4] "The Internet Movie Script Database (IMSDb)." *The Internet Movie Script Database,* www.imsdb.com/.

[5] "Using pre-trained word embeddings." *Keras,* keras.io/examples/nlp/pretrained-word-embeddings/ .

[6] Agrawal, Samarth. "What the heck is Word Embedding." towardsdatascience.com/what-the-heck-is-word-embedding-b30f67f01c81 .

[7] "IMDbPY" imdbpy.github.io/ .