Genre-Themed Movie Plot Generator

Hsiao-Yu "Joy" Chiang
UC Berkeley MIDS
Berkeley, CA
hsiaoyuchiang@berkeley.edu

Ann Yoo Abbott
UC Berkeley MIDS
Berkeley, CA
hyooabbott@berkeley.edu

Shubham Gupta
UC Berkeley MIDS
Berkeley, CA
shubham321@berkeley.edu

Abstract

In this project, our team takes a look at different techniques for text generation primarily focused on generating movie plots of a certain genre. We explore different techniques such as Long Short-Term Memory networks (LSTMs) and Generative Pretrained Transformer-2 (GPT-2) with a focus on Horror and Romance movie genres. For this project, we use Recall-Oriented Understudy for Gisting Evaluation (ROUGE) performance metrics to quantify our results.

1 Introduction

The ability to tell, craft, and understand stories has long been held as a hallmark of human intelligence (Winston, 2011). Storytelling is what connects us to our humanity. It is what links us to our past, and provides a glimpse into our future. Since humans first walked the earth, they have told stories, without a proper spoken or written language. Through cave drawings and bonfire nights, humans have told stories as a means of shaping our existence. Events occur naturally however as humans, we have unique perspectives that shape how a story is reiterated. Storytellers learned early on that listeners tend to enjoy stories that contain a beginning, a middle, and an end. We seem to be drawn to stories where the characters strike strong resemblances to our likelihood. We also yearn for an element that compels us into continuing the story. When that compelling element is met, we enjoy when the story's drama factor incrementally increases to a thrilling climax, followed by a satisfying conclusion. At times, we want to use our imaginations, and sometimes we would prefer to have a story told to us as we passively listen. Many of us enjoy being moved by a story, either emotionally or viscerally, like in a good romance film.

Recent advances in machine learning based approaches for natural language generation have led to exploration of many diverse but related text generation tasks. Our team wanted to bring in a powerful storytelling idea with different text generation methods to build a genre themed movie plot generator. Our aim is to create an original and brief storytelling summary for a movie through the generator. Plot lines were generated to inspire writers by triggering a creative chain of thought and to help set the scene to build characters and contemplate how they fit together in an interesting story. They can also be tuned to a certain genre such as horror/thriller or romance/romantic.

Neural language models are built using recurrent neural networks (RNNs), and the two popular variations of RNNs are Long Short Term Memory networks (LSTMs) and Gated Recurrent Unit (GRU) networks. GRU networks are simpler and easier to modify. However, they are relatively new, so we decided to build LSTMs from scratch for our storytelling movie plot generator. On the other hand, we have seen amazing progress in large scale pretrained language modes such as OpenAI GPT and BERT. OpenAI GPT-2, a direct successor to GPT language model and a large transformer-based language model with 1.5 billion parameters, could learn our summarization from the raw text using no task specific training data. In comparison to our own LSTMs model, we wanted to use OpenAI GPT-2 as well to create different sets of summarization and see the distinction in two different text generations.

After generating some movie plots, an evaluating summarization system was needed for us to tell if the plot generator completed what it was intended to do. ROUGE, Recall-Oriented Understudy for Gristing Evaluation, compared the automatically created plot against our original movie plot by how often certain terms and phrases in the original summary appeared in the machines generated summaries.

2 Background

Generating human quality text is a challenging problem due to meaning ambiguity and difficulty in modeling long term semantic connections. RNNs have shown promising results in this problem domain. The most common approach to its training is to maximize the log predictive likelihood of each true token in the training sequence given the previously observed tokens. Scheduled Sampling, proposed by Bengio et al. (2015), was proposed as an improvement to the maximum likelihood approach by stochastically introducing inference steps during training steps. More recently, Generative Adversarial Nets (GANs) that use a discriminative model to guide the training of the generative model have become popular in the vision domain, and also reinterpreted as a reinforcement learning problem to adapt it to text generation by Yu et al. (2016).

3 Data Sources and Preprocessing

We ran all of our experiments on two datasets: Wikipedia Movie Plots from Kaggle website (https://www.kaggle.com/jrobischon/wikipedia-movie-plots) and Carnegie Mellon University (CMU) Movie Summary Corpus from CMU's Machine Learning Department webpage (http://www.cs.cmu.edu/~ark/personas/). The Kaggle dataset contains descriptions of 34,886 movies from around the world including title, origin, director, plot, genre, and etc. We preprocessed the dataset to extract only the title, plot and genre replacing unknown genres with a blank space. Similarly, CMU dataset contains 42,306 movie plot summaries extracted from Wikipedia and aligned metadata extracted from Freebase, including: movie box office revenue, genre, release date, runtime, and language, character names and aligned information about the actors who portray them, including gender and estimated age at the time of the movie's release. The dataset was preprocessed by combining movie metadata and plot summaries to extract title, plot and genre. Finally, we removed unnecessary characters from the dataset. Duplicates from Kaggle and CMU were handled, and our total dataset carried 53,435 movie plots.

4 Generation via LSTMs

4.1 Predicting Level Selection

LSTM networks are state of the art algorithms for sequential data as they can memorize their previously seen elements through an internal state. There are two ways to generate text. One algorithm generates texts based on character and another generates based on text. We expect character-based LSTM networks to be able to mimic grammatically correct sequences with the caveat of a longer training time, while word-based LSTM networks are faster to train and can generate more coherent texts with the only downside being that the generated texts are far from being linguistically coherent. After exploring both methods, we decided to use character-based LSTM networks. The character-based model is more flexible and can learn rarely used words and punctuation. Furthermore, it has a much smaller vocabulary, which saves a lot of space when storing the vocabulary. On the other hand, word-based models can not generate out-of-vocabulary words since they are more complex and resource demanding.

4.2 Implementation

Generating text using the character-based LSTM networks is straightforward. We use TensorFlow's implementation of the Keras API specification (tf.Keras) and reference the text generation tutorial from TensorFlow core (https://www.tensorflow.org/tutorials/text/text_generation) to create our baseline model. We started by vectorizing the text into two lookup tables: one mapping characters to numbers, and another for numbers to characters. Here we restrict our charters to be all lowercase, and the punctuation marks allowed are "," and ".". Our process involves input being a sequence of characters, mapping it to a sequence of indexes to have the model predict the next index for us to convert it back to a character. Next, we divide the text into training examples and the targets. For each input sequence, the corresponding targets contain the same length of text, except shifted one character to the right. For example, with sequence length of 4 and a text such as "movie", the input sequence would be "movi", and the target sequence "ovie". Each index of the vectors are processed as a one-time step and as for the next timestep, executes the process again but in this occurrence, the LSTM networks consider the previous step context in addition to the current input character. Next, we created training batches and built the model.

4.3 Baseline LSTMs model

The baseline model consists of an input layer that is a trainable lookup table that will map the numbers of each character to a vector with embedding dimensions, a LSTM layer, and a dense layer as the output layer. For each character, the model looks up the embedding, runs the LSTM one time step with the embedding as input, and applies the dense layer to generate logits predicting the log-likelihood of the next character. It is possible to customize the hyperparameters in the model, which will be discussed later in the Results section.

5 Generation via GPT-2

Another approach we took to Movie Plot Generation revolved around using a Open AI technology called GPT-2. GPT-2 is a large transformer-based language model trained on websites linked from Reddit. GPT-2 aims to predict the next word given all the previous words in text. The model accepts input in the form of byte-pair encodings which allows faster training than word tokens but all the while keeps the case and formatting of the text. The model also provides outputs in the same format.

When interfacing with GPT-2, we utilized minimaxir's GPT-2-simple python library to allow for fine-tuning the base GPT-2 model on movie plots of different genres. Then using Google Colab and the dataset, which we had cleaned and prepared for our project, we were able to train a new model, generate predictions for test sentences, and evaluate our generated predictions against the reference plot.

5.1 Model Training

We trained two separate models in our project, one model focused on generating movie plots for horror movies and the one model focused on generating movie plots for romance movies. We started off by downloading a base medium size 355M GPT-2 model, which takes up 1.5GB of space up on disk. Then using the data set we had cleaned earlier, we began to fine-tune the base GPT-2 model on a corpus of horror movie plots and romance movie plots for about two thousand steps after which we observed that the average loss had more or less converged. We then exported these trained models for later use.

5.2 Test Set Generation

Using the fine-tuned models, we began to generate plots for our test set. This involved taking the existing movie plots, finding a breaking point somewhere along the plot to split the plot into a prefix and a reference section, and then utilizing our natural language processing (NLP) models to best generate a continuation of the movie plot as provided by the prefix. Although our initial goal was to try to generate a plot continuation that was somewhat meaningful and coherent, we later decided to quantify our model's effectiveness by comparing the generated plots with the reference portion of the original plot. When deciding on how to split the plot midway, we decided to try three different approaches and compare them. The three approaches are:

- 25% Prefix and 75% Reference
- 50% Prefix and 50% Reference
- 75% Prefix and 25% Reference

So if a movie plot consists of 100 words, we would use 25/50/75 words in our prefix and 75/50/25 words in our reference text in each level of our analysis. We settled on using such an approach since we wanted to test to see if providing our model with a larger prefix would result in higher accuracies in relation to the reference text. This technique also provided a consistent way of evaluating our model on short movie plots and also long movie plots.

5.3 Evaluation via ROUGE

When comparing our predicted text with the reference plot lines, our team decided to use the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) evaluation metric. Specifically, we take a look at ROUGE-1, which measures unigram overlap, ROUGE-2, which measures bigram overlap, and ROUGE-L, which measures the longest common subsequence between the prediction and the reference text. Each of these metrics provide us with F1, Recall, and Precision.

6 Results and discussion

6.1 LSTMs Baseline

To give us a sense of the learning capabilities of LSTMs, we defined two baseline models; one trained on the horror plots and another one trained on the romance plots. Each of the two baseline models has a single hidden LSTM layer with 1024 RNN units. The output layer is a Dense layer. After the baseline models were built, we would like to test the models' output by generating text with some sample prefixes.

> **Prefix**: Bella is killed by her husband, one dark and stormy night.
>
> **Horror Model**: She sees him to head the police to the car and a senduce and the continue the money and and the life and the list.
>
> **Romance Model**: After the two children and is a fight completed to be the confusing of the series and a music party and she has a start and married to set the partical and promiting the police and she is a secret find and she is set the banger.

In the sample prefix above, we see how the prefix itself already has horror movie connotations with words such "killed" and phrases like "dark and stormy night". From the generated text above, the characters are separated into word-like groups and many are actual words while some are not. In addition, some of the words make sense, but many do not form a coherent sentence. We are interested in the model output with a sample prefix starting with a romantic scene.

> **Prefix**: Bella marries the man of her dreams last summer on a boat.
>
> **Horror Model**: He is a former and the secret to the actually the party is survivors the secret as the scott and she are students to her believed to the police of the realizes the mission the car and finds the back to the are the town and the move.
>
> **Romance Model**: He is a love and the confesses she is a family and the film is a partice with his father and the boys and his father to a family relationship and she is seried and the party she was a story.

As seen from the prefix above, there are certain romantic themes present in the prefix with references to marriage, dreams, and summer. From the two example prefixes with different genre themes, the words generated in two models are quite different. The horror baseline model used words such as "police", "survive", and "secret", while the romance baseline models generated words such as "love", "confess", and "family". Although some of the sentences are grammatically correct, most are not coherent. The model has not learned the meaning of words, but considers it character-based. When training started, the model did not know how to spell an English word, or that words were even a unit of text.

To quantify and compare the results from different models, we generated plots with prefix from the test set, and used the ROUGE metric to produce the prediction scores tables. Table 1 and Table 2 show the ROUGE scores on baseline horror and romance models respectively. In these tables, R-1, R-2, and R-L is short for ROUGE-1, ROUGE-2, and ROUGE-L respectively.

Metric	**25%**	**50%**	**75%**
R-1 F1	0.193	0.178	0.145
R-1 Precision	0.259	0.209	0.131
R-1 Recall	0.236	0.247	0.253
R-2 F1	0.021	0.018	0.014
R-2 Precision	0.022	0.032	0.012
R-2 Recall	0.025	0.024	0.023
R-L F1	0.114	0.114	0.111
R-L Precision	0.136	0.122	0.098
R-L Recall	0.133	0.149	0.169

Table 1: LSTM Horror Genre Prediction Baseline Scores

Metric	**25%**	**50%**	**75%**
R-1 F1	0.223	0.206	0.164
R-1 Precision	0.286	0.227	0.142
R-1 Recall	0.278	0.288	0.292
R-2 F1	0.030	0.026	0.018
R-2 Precision	0.040	0.030	0.016
R-2 Recall	0.035	0.035	0.032
R-L F1	0.155	0.151	0.141
R-L Precision	0.187	0.162	0.126
R-L Recall	0.170	0.186	0.207

Table 2: LSTM Romance Genre Prediction Baseline Scores

With a single LSTM layer, Adam optimizer function, sparse categorical cross-entropy loss function, and trained for 100 epochs, both of the baseline models have the highest F1 scores with 25% as the prefix from the original plots.

6.2 Fine Tuning LSTM

We would like to see how the models can be improved with fine tuning. We have implemented multiple LSTM models to see if it can generate sentences more coherently, but many do not. For example, we could not find a better optimizer and loss function other than Adam and sparse categorical cross-entropy. Furthermore, adding a softmax activation function in the Dense layer did not improve the generation either. In addition, changing the learning rate did not improve the model performance. However, the baseline model with an additional LSTM layer and dropouts with a probability of 20 did make a difference on the ROUGE scores. The results are produced in below tables.

Metric	25%	50%	75%
R-1 F1	0.219	0.204	0.174
R-1 Precision	0.291	0.243	0.168
R-1 Recall	0.268	0.283	0.293
R-2 F1	0.028	0.025	0.018
R-2 Precision	0.041	0.032	0.018
R-2 Recall	0.032	0.031	0.028
R-L F1	0.140	0.134	0.122
R-L Precision	0.161	0.140	0.107
R-L Recall	0.172	0.185	0.202

Table 3: Two-layers LSTM Horror Genre Prediction Scores

Metric	25%	50%	75%
R-1 F1	0.244	0.221	0.181
R-1 Precision	0.298	0.241	0.161
R-1 Recall	0.298	0.310	0.318
R-2 F1	0.035	0.029	0.021
R-2 Precision	0.046	0.034	0.019
R-2 Recall	0.041	0.040	0.038
R-L F1	0.169	0.162	0.147
R-L Precision	0.192	0.166	0.127
R-L Recall	0.197	0.213	0.233

Table 4: Two-layers LSTM Romance Genre Prediction Scores

Table 3 and Table 4 show the ROUGE scores on tuned horror and romance models respectively. From the results, we can conclude that adding an additional layer and dropouts to the model can help improve the F1 score.

6.3 GPT-2

With our fully trained GPT-2 model, we decided to observe sample plot generations for both the horror and romance models. We provide the model with some sample prefixes and showcase the generated plot continuation below.

> **Prefix**: Bella is killed by her husband, one dark and stormy night.
>
> **Horror Model**: The next day, her husband, a small-time drug dealer, is murdered. The next day, her son, a young boy, is murdered. The next day, her daughter, a young girl, is murdered.
>
> **Romance Model**: She is left with her two sons, who are now in love with each other, and a daughter, who is in love with the boy she left for.

As we saw earlier, this sample prefix of Bella being killed by her husband has clear horror movie connotations. In the horror model plot continuation, we can see that not only is the text syntactically correct and flows well, but also the text continues the horror movie themes by mentioning murders and things such as ”small-time drug dealer”. However in the romance model plot continuation, despite the prefix only slightly hinting at a romantic element with the word ”husband”, we see the romance model generate text with this romance theme with references to love and phrases such as ”the boy she left for”. However, the plot of the romance model itself does not make much logical sense since it hints that Bella is still alive and her sons are in love with each other. We observed yet another sample generation, this time with a prefix with more romance related themes.

> **Prefix**: Bella marries the man of her dreams last summer on a boat.
>
> **Horror Model**: The couple settle down and have a child. When the couple are in the kitchen, they notice a large piece of meat on the floor and quickly run to it. They open it to find that it has been mutilated.
>
> **Romance Model**: But the real boyfriend, a rich businessman, is

in love with Bella. Bella is heartbroken
and leaves town.

As seen before, the romantic themes are prevalent in the prefix text. The romance model continues these themes with words such as ”boyfriend” and ”heartbroken”. The horror model, on the other hand, quickly transitions from a romantical theme into a horror theme with reference of mutilation. The horror model in this case, also makes more somewhat logical sense than the romance model.

Aside from these sample generations, we tried to quantify our model plot continuations by using the ROUGE metrics mentioned earlier to compare the generated text with the original plot. In Table 5 below, we show the performance results of our horror model using the 25%, 50%, and 75% framework laid out earlier.

Metric	**25%**	**50%**	**75%**
R-1 F1	0.295	0.285	0.259
R-1 Precision	0.310	0.306	0.280
R-1 Recall	0.287	0.278	0.257
R-2 F1	0.044	0.042	0.039
R-2 Precision	0.048	0.047	0.043
R-2 Recall	0.043	0.040	0.039
R-L F1	0.212	0.212	0.203
R-L Precision	0.293	0.273	0.241
R-L Recall	0.178	0.184	0.187

Table 5: GPT-2 Horror Genre Prediction Scores

From this table, we can see that the horror model performed the best when only 25% of the original plot was used as a prefix. Also, it seems that the larger the prefix became, the worse the F-1 scores for all three ROUGE-1, ROUGE-2, and ROUGE-L metrics. We see a similar pattern with the performance of the romance model in Table 6.

Metric	**25%**	**50%**	**75%**
R-1 F1	0.296	0.290	0.261
R-1 Precision	0.317	0.308	0.284
R-1 Recall	0.285	0.282	0.258
R-2 F1	0.045	0.044	0.038
R-2 Precision	0.049	0.047	0.044
R-2 Recall	0.043	0.042	0.037
R-L F1	0.214	0.219	0.209
R-L Precision	0.287	0.274	0.247
R-L Recall	0.181	0.192	0.195

Table 6: GPT-2 Romance Genre Prediction Scores

Both the scores of our romance model and their performance trend is similar to that of the horror model. A possible explanation behind this trend is that having a smaller prefix leaves a much larger reference text for comparison. And since the ROUGE metric focuses on the number of overlapping words, having more words to compare with can lead to more overlaps and hence larger scores in both precision and recall. Whereas a larger prefix would leave only a few words available for overlap. As a result, our model would find the process of predicting more words complicated.

It is also worth noting that although the scores from our ROUGE evaluations are quite low, it does not mean that the generations produced by our model are gibberish. Instead, this evaluation allows us to view the performance of our model through one particular frame by comparing it to the specific original movie plot.

7 Conclusion

Our project pushed us to dive deep into the world of NLP, and connect various themes and topics we learned in class to real problems relating to movie plot generation. Through the use of LSTM networks and transformer architecture, we aim to build models that mimic human creativity in romantic and horror themes. And although our generation at the moment is limited by its logical consistency, we have succeeded in generating story lines that carry forward the essence of their assigned genres.

References

Bamman, David, et al. 2013. *Learning Latent Personas of Film Characters.*

Bengio, Samy, et al. 2015. *Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks.*

Bhooshan, Suvrat, et al. 2016. *Evaluating Generative Models for Text Generation.*

R, Justin 2018. *Wikipedia Movie Plots.*. www.kaggle.com/jrobischon/wikipedia-movie-plots

Tensorflow *Text Generation with an RNN: TensorFlow Core.*. www.tensorflow.org/tutorials/text/textgeneration

Winston, Patrick. 2011. *The Strong Story Hypothesis and the Directed Perception Hypothesis.*

Yu, Lantao, et al. 2016. *Seqgan: sequence generative adversarial nets with policy gradient.* http://people.ischool.berkeley.edu/ hyooabbott/w209/hw1pt2