

Introduction

Two-Sentence Horror is a creative form of storytelling where users attempt to scare the reader in a short medium. In this format, the narrative is developed over just two sentences, challenging writers to evoke fear, suspense, and tension in a limited amount of words. The first sentence typically sets the stage or introduces a seemingly normal situation, while the second sentence delivers a horrifying and usually unexpected twist, leaving the reader uneasy and in shock. This minimalist form of storytelling emphasizes the potency of short-form narratives, greatly leaning into the ability for people to imagine. This is even more important nowadays as most forms of media have become short-form, ranging from social media posts to YouTube videos rather than reading books.

Two-Sentence Horror is primarily found on a subreddit named `r/TwoSentenceHorror`. They have roughly 1.3 million subscribers with an average of 200 to 300 post a day. As of December 2023, they are the 447th most popular subreddit. Given the Reddit user base of 430 million monthly active users and 52 million active daily users (as of 2019), `r/TwoSentenceHorror` is, all things considered, a very active subreddit.

We found the concept of Two-Sentence Horror interesting because of various reasons, one being that it is an unexplored field. As far as we were aware of, there is one other research paper (titled “HorrifAI: Using AI to Generate Two-Sentence Horror”) that attempts to create a similar language model. Through our exploration, we found no publicly available dataset for Two-Sentence Horror that we deemed usable for our models. Given the sheer scale of the subreddit `r/TwoSentenceHorror`, we figured this would be a good opportunity to scrape real data from user input.

We also intended on pursuing Two-Sentence Horror as we found language models relatively poor at understanding human emotion. We believed it would be interesting to see if a model could understand how to invoke fear in a reader from these horror stories.

In addition, replicating how a Reddit user may speak and comment on a public forum is a relatively mature field. For reference, subreddits such as `r/SubredditSimulator`, `r/SubSimulatorGPT2`, and `r/SubSimGPT2Interactive` display other users’ attempts at creating language models based on real-time data of other subreddits. From what we gathered, the account that would normally post Two-Sentence Horror content, `u/twosentencehorrorGPT`, has been suspended. This leaves a tremendous opportunity, we can create a language model for short-form horror stories and fill in a niche that has yet to be covered.

Dataset

Our dataset, which we named “Two Sentence Horror (Jan 2015 - Apr 2023)”, is something that we created from scratch. The original data contains about 107K user posts, but after data cleansing, we had about 95K posts available to train and fine-tune with.

The dataset was collected from the subreddit `r/TwoSentenceHorror` using the PullPush.io API supplied by Reddit over the course of roughly 1100 GET requests. With a frequency of 200 to 300 new stories a day, extracting all data would become overwhelming, especially when most of the stories submitted are very low-quality. Thus, our criteria for a post was that it had at least 21 or more upvotes and was posted after January 1st, 2015. From observation, stories either

received none or over 21 upvotes. We decided that this metric would be a relatively okay cut-off for determining the quality of a story.

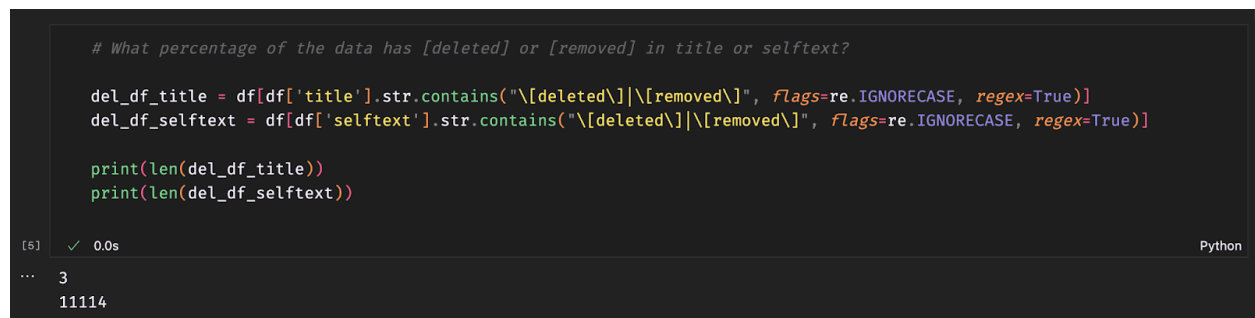
When extracting, we looked for five specific elements of a Reddit post. Primarily, the post title and selftext contained the story itself (with the title being the first sentence and the selftext being the second). We also extracted metrics evaluating a post's performance - its score (number of upvotes), num_comments (number of comments), and gilded_count (number of awards received).

The data was extracted using Python's requests library (file reference: 'dataset/code/scrape_data.ipynb'). It was stored into a CSV file and is publicly available on Kaggle listed as the dataset "Two Sentence Horror (Jan 2015 - Apr 2023)". The original data is named "org_reddit_scrape_20_Jan2015_timestamp.csv".

Dataset Exploration

While we extracted 107K user posts, only about 95K of those posts were usable for our model. While exploring the dataset, we noticed common trends in certain user posts that we could deduce were not either not a story or not fit for our model. For one, many Reddit tags are surrounded by "[" and "]" tokens, such as "[removed]" or "[MAY2020]". Posts that were deleted were removed from the dataset completely, whereas posts that contained tags remained with the tags removed (refer to 'dataset/code/process_data.ipynb').

While removing 12K posts may seem like a lot after this first step of data cleansing, it is important to note that between May and July 2023, a change on Reddit's API pricing and availability caused a protest of many Reddit users. Besides causing a site-wide blackout, many users deleted their accounts in protest, resulting in a greater influx of "[removed]" and "[deleted]" posts.



```
# What percentage of the data has [deleted] or [removed] in title or selftext?

del_df_title = df[df['title'].str.contains("\[deleted\]|\[removed\]", flags=re.IGNORECASE, regex=True)]
del_df_selftext = df[df['selftext'].str.contains("\[deleted\]|\[removed\]", flags=re.IGNORECASE, regex=True)]

print(len(del_df_title))
print(len(del_df_selftext))
```

[5] ✓ 0.0s Python

... 3
11114

Figure 1: Amount of data that was deleted or removed on Reddit

After removing all posts that seemed to be invalid as well as tags in the texts of posts, we originally proceeded with attempting to train our models but frequently ran into either memory problems or only predicting padding tokens. Upon further observation, our dataset had a very imbalanced distribution of token lengths.

While the longest selftext was 461 tokens long (all sequences were padded to the longest length for both title and selftext), 99.76% of the data was below 50 tokens, 98.83% was below 40 tokens, and 92.18% was below 30 tokens. The result was a significant waste of memory with sequences being over-padded for seemingly no benefit. We made the design decision to remove all posts from our dataset above 40 tokens as we considered that a good balance between retaining the context of r/TwoSentenceHorror while also saving significantly on memory usage while training.

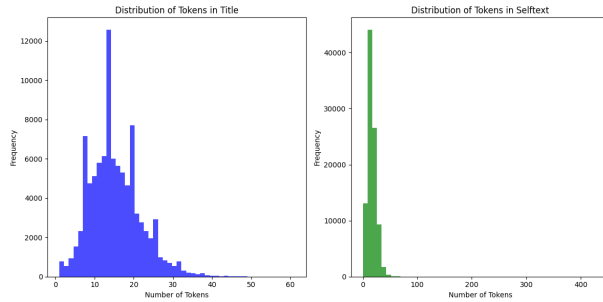


Figure 2: Data (all tokens)

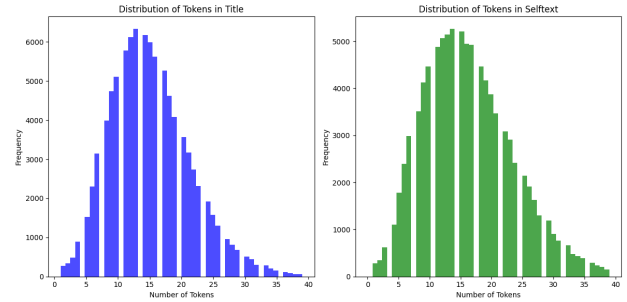


Figure 3: Data (40 or less tokens)

Methods and Results

With a combination of material from class and research, we decided to come up with four methods to implement our model. Our base model was a Keras Sequential model that we altered from Homework 5 of class (Natural Language Processing, CS4120). Then, we explored outside class material to look into more advanced transformers such as Seq2Seq, GPT-Neo, and Bart.

Our **Keras Sequential** model is a replication of Homework 5, where we tokenize and pad the dataset based on an n-gram size of 3, then sequence it using Word2Vec to create word embedding vectors (embedding size of 100). After creating a data loader, we trained the model for one epoch (due to risk of overfitting) on a batch size of 128. The model architecture was an input Dense layer, a Dropout layer, then an output Dense layer.

This model performed, unsurprisingly, the worst of all the models we created. While the generated sentence attempts to be horrifying, it lacks any understanding of the input provided, something we can achieve with an LSTM. Surprisingly, the generated sentences had some coherency to them despite next word prediction being similar to a frequency model.

- Ex 1: **There was a ghost.** Wait, - was rather me.
- Ex 2: **I was horrified when I got my tests back.** I'd put a man to said a lights toward me, their murders is them.
- Ex 3: **My parents told me not to go upstairs.** I've been terrible, it was brought a odd way over my lie, , she know they can stop, I wonder how deep my ankle.
- Ex 4: **I got out of bed this morning.** For two better for to scream as something now before the dark, they came from next.

We noticed that after training for more than one epoch, the model was no longer improving (loss and accuracy were not improving). We figured that upon convergence, we should stop training as that could lead to overfitting. As an experiment, we tried training this model for 10 epochs and, as expected, the model output was very static regardless of input (overfit).

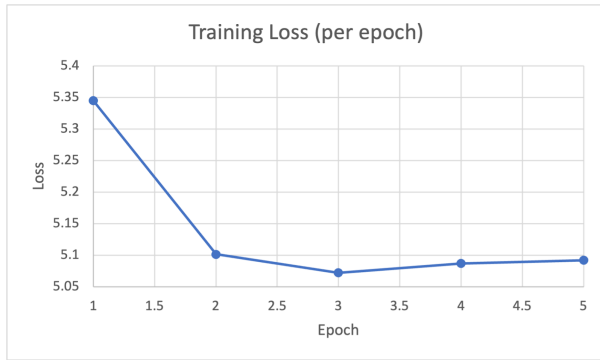


Figure 4: Keras Seq Train Loss

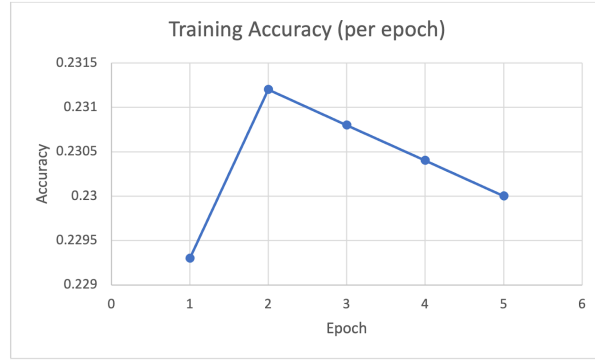


Figure 5: Keras Seq Train Acc

Seq2Seq is the first venture we made into an encoder-decoder pattern. This gave us the ability to specifically provide one input and receive one output (sequence-to-sequence). This pattern also was the first that fit with the structure of our data. We have two “inputs”, the first and second sentence. With an encoder-decoder pattern, we can pass in the first sentence to the encoder and the second sentence to the decoder when training, which helps the model understand the connections between the pair of data. In addition, this model implements an attention layer which, when our model scales to thousands of encoder and decoder units, helps the model decide which units to actually focus on.

Our implementation was done in Tensorflow. We first surrounded the input with <BOS> and <EOS> tokens (beginning and end of sentence), then created the encoder/decoder pattern (embedding size of 256, LSTM size of 256). We experimented with two different runs for this model as training time was fast.

Initially, we wanted to see what would happen if this model was trained for 40 epochs. When logging the metrics to Tensorboard, everything looked perfect at first:

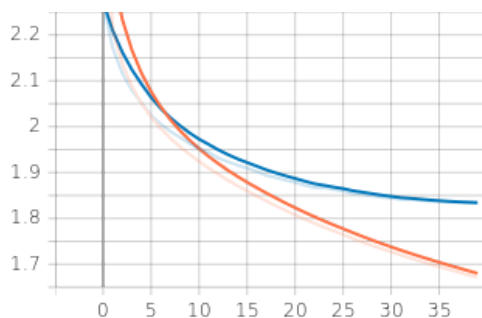


Figure 6: Seq2Seq Loss, 40 epochs

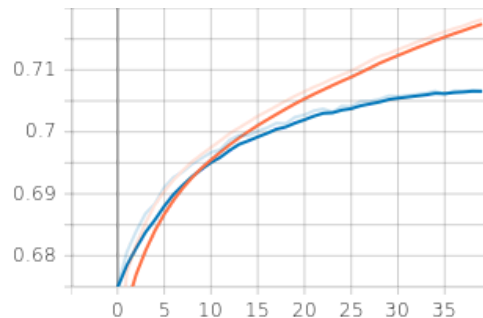


Figure 7: Seq2Seq Acc, 40 epochs

However, on inference, the results were very static. Regardless of the first sentence, the second sentence would be very similar, referencing legs or some sort of body part:

- Ex 1: **It's crawling!** So why do the rest of his house
- Ex 2: **I was horrified when I got my test back.** I was confused when I heard the sound of the sound of my legs and legs
- Ex 3: **My parents told me never to go upstairs.** I was confused when I heard the sound of the sound of my legs and legs

When re-trained for just 5 epochs instead, the results were not much better:

- Ex 1: **I got out of bed this morning.** by the limb struggling that malnourished i dropped you understand that the scopes didn't stop but 2 as instead
- Ex 2: **I was horrified when I get my test results back.** i begged the answer blessedly coming to eat my flesh i an' "i have realise that where not not never who "
- Ex 3: **My parents told me not to go upstairs.** and something was traffic champagne tellers and never confused crawling off the chair on hair of screams
- Ex 4: **There was a ghost.** the needs hershey's slowly blood at miles from allowed full of and screaming and won't keep picking me and approaching the chance smiling away her window

It seems like the model understood some of what the first sentence was saying, but did not properly understand how to parse together coherent English sentences. While we generated our own word embeddings, it would be interesting to reattempt this again with pre-trained word embeddings such as Google's Word2Vec.

BART was the next step in our model journey as we began entering the world of transformers. Unlike it's relative BERT, BART still utilizes a Seq2Seq-like model architecture (encoder-decoder) underneath rather than being an autoregressive model. In a sense, this partially fulfills our search for how our Seq2Seq model may have performed if it had an understanding of English beforehand.

With BART, we began fine-tuning pre-existing models rather than creating them from scratch. The benefit of this is that they come already pre-trained understanding how English works. Using Hugging Face's 🤖 Transformers library (using PyTorch), we can fine-tune the existing weights towards a more horror-themed output. For our model, we fine-tuned BART for 3 epochs at a learning rate of $5e-5$ to avoid catastrophic forgetting.

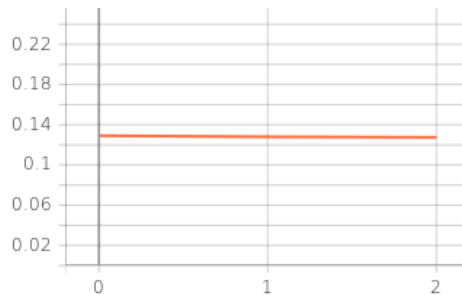
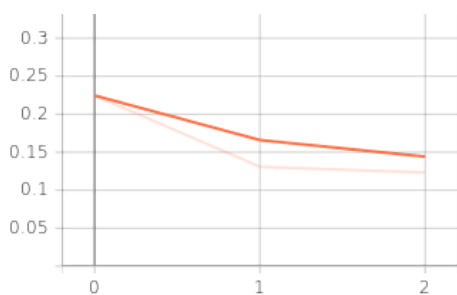


Figure 8: BART, Train Loss, 5 epochs Figure 9: BART, Val Loss, 5 epochs

Given the model's understanding of English and it being fine-tuned to purely two-sentence horror, the model was able to produce very solid horror stories:

- Ex 1: **My parents told me not to go upstairs.** i don't know what's worse, the fact that i'm the only one down here, or that i can hear them screaming.
- Ex 2: **There was a knock on the door.** it was the only way i could get out of the basement.
- Ex 3: **There was a loud noise coming from the basement.** it was only when i turned on the lights that i realized the noise wasn't coming from the basement.
- Ex 4: **There was a ghost.** it was the only thing keeping me alive.

Note, for our BART model, we used Gradio from Hugging Face to host it online, feel free to give it a try! <https://huggingface.co/spaces/voacado/two-sentence-horror-bart>

Gpt-Neo is a replication of GPT-3, created by EleutherAI. The Hugging Face transformers library has a module named GPTNeoForCausalLM that uses any of the GPT variants for the model. Neo variants can range from 125 million parameters to 2.7 billion. The variant we chose was the 125 million for time efficiency and hardware limitations. GPT-Neo was trained on Pile and is best at generating a prompt, even lewd and abrasive, which is perfect for Two-Sentence horror.

We took inspiration from common uses of GPT-Neo to implement this for our dataset. First, every entry was edited to begin with bos and end with eos. The two sentences were separated by a bos tag and any that were not at maximum length of fifty were padded with a pad token. For weighting, we played around with a couple methods but came upon preferring posts with over 40 upvotes. This still gave around 60k quality posts and resulted in a lot better sentences.

We used the trainer and training arguments library to train the model with our dataset because these two libraries specialize in transformers. Trainer takes in the model, trainer arguments, and a tailored dataset from the class that we made for it. We played with some argument numbers but settled on 4 epochs and a batch size of 4, which made it around 15 minutes per epoch for a total of an hour training. We ran into memory issues and time constraints, along with imperfect results requiring constant tuning but settled on those numbers.

Then, the GPTNeoForCausalLM model that we made has a generate function that generates some outputs. The results are varying with some not producing a second sentence and some not being coherent. Ultimately, it produced some good quality sentences that we used for our presentation.

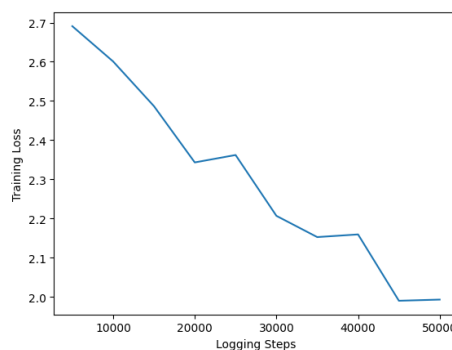


Figure 10: Training Loss over Logging Steps

Next Steps and Conclusion

If we had more time, we would love to expand on evaluating how “scary” a generated story can be. Currently, we access this using Cosine Similarity with TF-IDF and Jaccard Similarity to fetch the most similar post and its weighted score (a mixture of upvotes, comments, and gilded count). We looked into metrics like BERTScore and BARTScore which could also compare the semantic meaning of a sentence to the dataset, but found that even after vectorizing the entire original dataset, took about 21 seconds to calculate which was too slow.

Another next step we would love to look at is seeing how we can get Seq2Seq to generate more English-like sentences, perhaps by using pre-trained word embeddings.

Overall, we are very happy with the quality of the work produced and excited that some of our models were able to perform! For us, this will go onto our resume.

Works Cited

- <https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/reports/custom/15841116.pdf>
 - HorrifAI: Using AI to Generate Two-Sentence Horror
 - Paper from Stanford with similar goal
- <https://backlinko.com/reddit-users>
 - Count of Reddit users (as of 2019)
- <https://www.kaggle.com/datasets/voanthony/two-sentence-horror-jan-2015-apr-2023>
 - Our dataset
- <https://huggingface.co/spaces/voacado/two-sentence-horror-bart>
 - Online Gradio interactive interface for BART model

For fine-tuning:

- <https://stackoverflow.com/questions/60732018/how-to-make-bert-model-converge>
 - Catastrophic forgetting
- <https://github.com/Felflare/rpunct>
 - Restoring punctuation
- <https://stackoverflow.com/questions/46924452/what-to-do-when-seq2seq-network-repeats-words-over-and-over-in-output>
 - Text Degeneration
- <https://arxiv.org/abs/1904.09751>
 - Text Degeneration

Artist's Statement

The Two-Sentence Horror language models we have created can be used to generate short, impactful horror stories that will captivate and unsettle the reader. Imagine if each generated story was displayed on a screen or projected on a wall in a dark room, fading in and out with an ambience fit to create some form of psychological horror.

Unlike regular storytelling, Two-Sentence Horror is a neat form of story-telling that requires intention in every one of the limited word space to effectively convey the intended emotions. Context is an important hurdle for artificial intelligence because many n-gram based models can only look a limited distance beyond the current, making it hard to create a coherent message. Two-Sentence Horror stories often use a twist in the middle to create their surprise and suspense. A model on this would have to learn all of sentiment analysis and text generation, while also understanding broad contexts and connecting elements from the entire story, making it a stepping stone towards greater implementations of natural language processing. By mastering these topics, the model becomes not just a text generator, but a storyteller, a creative mind. Such a model not only shows the capabilities of AI in creativity but also pushes the boundaries of what machines can achieve in understanding and generating compelling narratives.