Novel Genre-Based Story Summary Generation

Stanford CS224N Custom Project

Alexis Echano
Department of Computer Science
Stanford University
aechano@stanford.edu

Jenny Mai
Department of Computer Science
Stanford University
mcmai@stanford.edu

Abstract

Our project focuses on leveraging Generative Pre-trained Transformers (GPT Models) to generate diverse book summaries prompted a genre/list of genres. To measure the effectiveness of augmenting GPT models for the purpose of generating story summaries, we implement two baselines: the LSTM model and a vanilla GPT-2 model without fine-tuning. Then, we implement two more GPT-2 models and augment both of them by (1) using better decoding strategies, (2) GPU optimizations, and (3) fine-tuning on story summary dataset. Using both automatic and qualitative evaluation of each model, we ultimately find that the fine-tuned GPT-2 model with 5 epochs performs best in consideration of semantic value of the book summary. By focusing on short book summaries, we develop a supplemental natural language processing tool that has the potential to efficiently prototype unique and novel story ideas for the creative literature field.

1 Key Information to include

- Mentor: Heidi Zhang
- External Collaborators (if you have any): None
- Sharing project: None

2 Introduction

With the advent of large language models, text generation has been a hot topic within the natural language processing community. However, many natural language processing text generation research focuses on generating coherent and goal-directed texts through training on large bodies of homogeneous text, such as large movie scripts or full stories. As a result, this sacrifices diversity of data and text fluency. In our project, we focus on generating coherent, genre-focused story summaries through the use of a diverse dataset of short book summaries and cutting-edge natural language processing models such as Generative Pretrained Transformers, or GPT models. By expanding to smaller but more heterogeneous datasets, we hope to progress the use of text generative models as a creative tool for diverse ideation of future ideas. In the case of generating short book summaries, we hope to aid authors in formulating unique ideas for their own novels.

3 Related Work

Fatima et al. (2022) explores a wide variety of natural language processing models used or currently being researched in text generation. In our work, we focus on two models from the paper: the Long Short-Term Memory and the Generative Pre-trained Transformer model.

Long Short-Term Memory: The first model we introduce is the Long Short-Term Memory (LSTM) model. LSTMs are an extended version of recurrent neural networks (RNN) that can encode information over longer text and solves the vanishing gradient problem Hochreiter and Schmidhuber (1997). Because LSTMs hold information over many iterations well, we use LSTMs as a baseline to compare GPT models to.

Research done by Santhanam (2020) explores the potential of using context in LSTM models to generate both syntactically and semantically correct sentences. Given a set of context input text, the output text should be related to the context given to the model. We use this idea in our models by providing prompts with specified genre(s) that correspond with specific book summaries during training, such that the model can theoretically draw relations between the context and the book summary. Using LSTMs as a text generation model is now widely explored through many internet webpages and blogs. For guidance on implementing a simple

LSTM using Pytorch, we turned to a public Kaggle notebook named "Text generation via RNN and LSTMs (PyTorch)" by Purva Singh..

Generative Pre-Trained Transformers: Generative Pre-Trained Transformers are highly researched and utilized models within NLP. OpenAI's GPT model has gone through multiple iterations and in this project, we focus on GPT-2. The original paper behind GPT's creation by Radford et al. (2018) aimed to used unlabeled corpus text, which makes up many documents and articles on the Internet, in discriminative fine-tuning to complete specific tasks like commonsense reading. Since GPT-2 was comprehensively trained on 40 GB of Internet text, it has outperformed other language models while not needing to be trained on specific tasks or data (though has the capability to do so in fine-tuning processes). GPT-2 has also made large strides in producing large human-like articles from a variety of human-generated prompts and even being able to answer questions about historical events and creative storytelling (OpenAI).

Focusing more on creative text generation for both models, Alabdulkarim et al. (2021) experimented with goal-directed story generation using LSTMs and GPT-2. The researchers used a corpus of science fiction stories and a reward system to encourage the model to create stories with goal verbs like mentioning "discover" in the story. They also incorporated both quantitative and qualitative analysis to evaluate their final model, which used GPT-2 architecture to generate goal-directed stories and found success in generated text with the goal achieved. However, they discovered pitfalls in repetition and other grammar related errors within the outputs.

Lastly, another research team studied genre-controlled full story generation using contrastive learning methods and GPT-2 through training on a movie synopsis dataset. They improved upon the GPT-2 model by manipulating the loss function and incorporating supervised learning to learn different genre features of a story. Their conclusions found that it was possible to generate meaningful stories with genre controls, though at the loss of text fluency and overall coherence (Cho et al., 2022).

4 Approach

4.1 Baseline

4.1.1 LSTM Model

We implement a simple LSTM model with 30M parameters as a baseline. LSTM models implement long "short-term" memory using layers with forget gates, input gates, output gates, and cells to determine what information to retain and to drop. The forget gate looks at the previous cell state to determine what information to retain and what information to discard. Input gates control what new information keep in the cell state. The output gate controls how important that information is for the next cell. The hidden layers and cell states are updated to continually determine how to deal with information Hochreiter and Schmidhuber (1997).

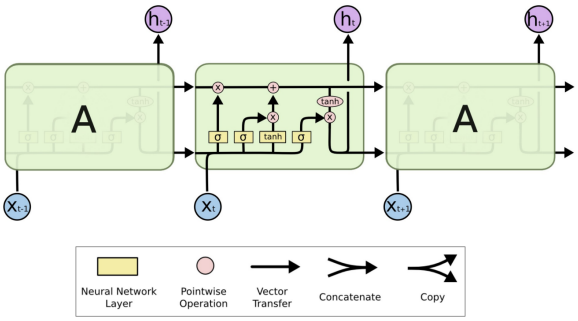

Figure 1. Simple LSTM architecture from Colah (2015)

4.1.2 Vanilla Pre-Trained GPT-2 Model

We use an unmodified version of HuggingFace's smallest GPT-2 model, originally created by OpenAI, with 124M parameters as a baseline. In our research, we did not modify the underlying structure of GPT-2 and instead, used the original GPT architecture put forth by Radford et al. (2018) as a foundation. The unsupervised pre-training for the transformer model attempts to maximize the following likelihood:

$$L(\mathcal{U}) = \sum_{n=i} \log P(u_i | u_{i-k}, ..., u_{i-1}; \Theta)$$

where $\mathcal{U}$ represents an unsupervised corpus of tokens, $\{u_1, u_2, ..., u_n\}$. Unlike a complete transformer that has both an encoder stack and decoder stack, the generative pre-training model uses a multi-layer Transformer decoder that uses multi-headed self-attention over input context tokens and feedforward layers to output a distribution of tokens (Alammar, 2019; Radford et al., 2018).

To further improve the original GPT model, research conducted by Radford et al. (2019) formulated the next iteration called GPT-2. In this model, researchers included layer normalization at each sub-block's input and scaled residual layer weights by $1/N$ where N is the number of residual layers. They additionally increased the context size from 512 to 1024 tokens, increased the batch size to 512 and expanded the vocabulary to 50,257. While this model achieved a good perplexity value of 8.6 and an F1 score of 89 which nears human performance, it is not perfect (Radford et al., 2019).

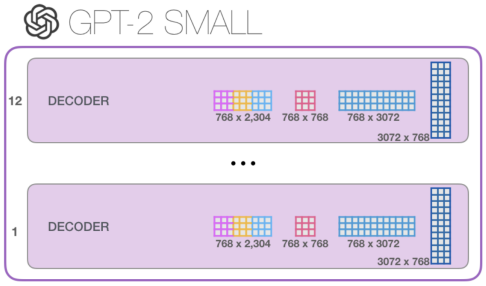

Figure 2. Small GPT-2 architecture from Alammar (2019)

There exist prompts where the GPT-2 model cannot generate relevant and coherent outputs due to a lack of task-

specific training. Therefore, to solve this shortfall, we aim to fine-tune the GPT-2 model using our book summary data and additionally implement GPU memory optimizations for training and add different decoding methods in GPT-2's final decoder block.

4.2 Improving the GPT-2 Model

To further build upon the base GPT-2 model, we employed three different ways to improve the model which includes (1) using better decoding strategies for text generation, (2) adding optimizations to reduce GPU memory usage during training and (3) finally, fine-tuning the model on our book summary data.

4.2.1 Decoding Strategies

In the final decoder layer of the GPT-2 model, the output vector is multiplied by the embedding matrix to get probabilities for different tokens of the vocabulary. From this, the model can choose the next token based on its calculated probability, like just selecting the word or token in the vocabulary with the highest value (Alammar, 2019). HuggingFace's process for language generation enables customization for this text generation through their pre-built `generate()` method, which capitalizes on GPT-2's autoregressive language generation. This means that generating text is based on probabilities of possible word sequences that are dependent on given context (von Platen, 2020). Below are three strategies we used to generate relevant story summaries.

Beam Search: This decoding strategy keeps a given number of hypotheses at each time step as possible paths of text generation. This method mitigates any earlier tokens from being ignored in the output sequence, as they may have lower probabilities but do end up contributing to the overall generated text sequence (HuggingFace, c). The 2 beam hypothesis is illustrated with a sample output in the figure below, created by von Platen (2020).

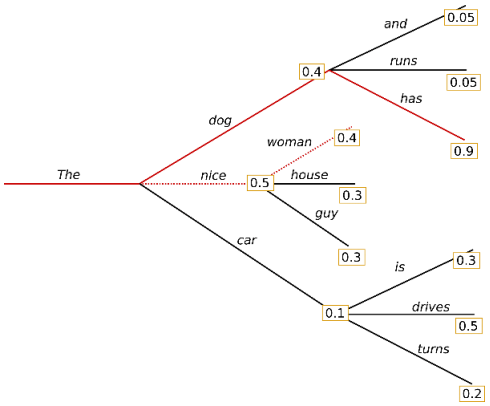

Figure 3. Beam search with 2 beams to generate output

"Generate a book summary with genre science fiction:\n\nThe story is a story about a group of people who are"

Figure 4. Sample output of the vanilla GPT-2 model with 2 beams

N-Gram Penalties: One issue that arises with beam search is that generated text contains repeated word sequences, as seen in Figure 3. This is because the process aims to find the entire word sequence with the highest probability. To prevent this issue, we included n-gram penalties where an existing sequence of n tokens are deliberately not included in the output sequence to prevent repetition (von Platen, 2020).

Multinomial Sampling: Lastly, we implemented multinomial sampling with the previous beam search and n-gram penalty methods through enabling the `do_sample` parameter in the text generation process. This means that the chosen word is instead determined by its conditional probability distribution below (von Platen, 2020).

$$w_t \sim P(w|w_{1:t-1})$$

where w_t represents the next chosen word.

"Generate a book summary with genre science fiction:\n\nA collection of tales from the 1980s, including "The Last"

Figure 5. Sample output of the vanilla GPT-2 model with all three strategies

The figure above shows the result of applying all of the above decoding strategies with GPT-2 and HuggingFace's generation method, which will be developed further in the fine-tuning process.

4.2.2 GPU Optimization

In order to actually fine-tune the large language model while saving memory space on the GPU, we utilized methods outlined by HuggingFace (b). The three methods we used are further explained below and were implemented within the training arguments of the HuggingFace Trainer object.

Gradient Accumulation: Instead of calculating gradients for each batch at once, this process calculates the gradients in smaller steps. By accumulating the gradients after a forward and backward pass through the model, we then run the optimization step to be able to run larger batches using less GPU memory but resulting in a slower training time.

Gradient Checkpointing: Enabling this in the training arguments resulted in further declines in memory usage by saving certain activations throughout the backward and forward passes. In doing so, only a fraction of the activations need to be re-calculated for gradients.

Mixed Precision Training: Finally, mixed precision or FP16 training describes a way to reduce the precision of the variables to make their computations faster and take up less GPU memory. Instead of holding the values in 32-bits, they are instead saved in half the precision as 16-bit values. However, gradients are still computed at full 32-bit precision to not lose any robustness in the optimization step.

4.3 Fine-tuning with Book Summary Data

Our main text generation process relies on the fine-tuned version of GPT-2 along with the aforementioned decoding and optimization strategies. We adapted foundational code from

official documentation from HuggingFace (a) and from an article regarding fine-tuning GPT-2 for song lyrics generation by St-Amant. Through using HuggingFace's infrastructure and our train and validation datasets, we fine-tuned the model using 3 epochs and 5 epochs and validated the model after every epoch. We then tested our improved GPT-2 on the test set for quantitative scores and a separate set of genre prompts for human evaluation, which will be further discussed in the experiments section.

5 Experiments

5.1 Data

The dataset we are using in this project is Carnegie Mellon University's book summary and genre dataset (Jasminyas, 2018). It contains 16,559 books from Wikipedia with data about each book's IDs, title, author, publication date, genres, and book summary. It contains 43.5 MB of data in a tab-separated text file.

After doing some data exploration, we found that the most common genres in the dataset are Novel, Science Fiction, Speculative Fiction, and Children's Literature.

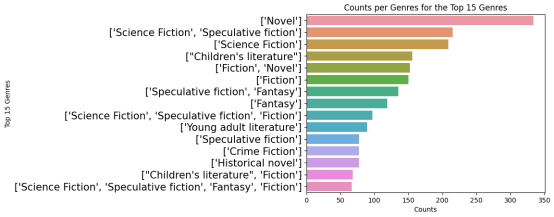

Figure 6. Top 15 most common genres

To integrate relationships between the book summaries and the genres to train our LSTM and GPT-2 models, we prefixed a prompt in front of each summary text – "Generate a book summary with genres [list of genres]:\n". In order to optimize training, we conducted our training, validation and testing on a subset using 5,000 elements of the dataset above. We split the data into 80% training, 10% validation and 10% testing. This is what the data looked like after prefixing the prompt in front of each summary text:

	Text
0	'Generate a book summary with genres Science Fi...
1	'Generate a book summary with genres Fantasy:\n...
2	'Generate a book summary with genres Crime Fict...
3	'Generate a book summary with genres Fiction, N...
4	'Generate a book summary with genres War novel,...

Table 1: Prompt-appended Summaries

After prepending the prompt to the book-summary, the last step to finish preprocessing was to tokenize and create a vocabulary for the data. For the LSTM model, we used the Pytorch tokenizer and vocab builder to easily convert tokens into their token ID's and vice versa. Similarly, to preprocess for the GPT-2 models, we used the HuggingFace GPT-2 specific tokenizers, which conduct a similar process as above.

5.2 Evaluation method

Because we are evaluating the quality of text, we use both machine-centric and human-centric measures.

5.2.1 Perplexity

The first quantitative metric we use to compare our models is the perplexity metric. Perplexity is a measurement of how confident or unconfident the models are in generating text based on the distribution of texts they were trained on. One way to calculate perplexity is through the normalized inverse probability of the test set:

$$PP(W) = \sqrt[n]{\frac{1}{P(w_1, \cdots, w_n)}}$$

Another way to calculate this is as the exponential of the cross-entropy loss:

$$PP(W) = \exp\{-\frac{1}{n} ln P(w_1, \cdots, w_n)\}$$

The lower the perplexity score is on the test set given the model, the better the model is at predicting a target input text sequence.

5.2.2 BERTScores

The second metric we use is the BERTScore precision, recall, and F1-score. To generate these metrics, BERTScore compares the cosine similarity between words of the generated text to the target text. The BERTScore precision, recall, and F1-score are defined as follows:

$$R_{\text{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_\mathbf{j}$$

$$P_{\text{BERT}} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_i \in \hat{x}} \max_{x_j \in x} \mathbf{x}_i^\top \hat{\mathbf{x}}_\mathbf{j}$$

$$F_{\text{BERT}} = 2 \frac{P_{\text{BERT}} \cdot R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}}$$

The BERTScore is correlated with human judgement on both sequence evaluations and system evaluations (Zhang* et al., 2020). We only compare BERTScores for our transformer models as BERTScores leverage pre-trained contextual embeddings from HuggingFace transformer models and our LSTM model is not transformers-based.

5.2.3 Human-Evaluation Metrics

For our qualitative evaluation method, we chose to rate text generated by each of our models given the same prompt. This is because human-centric analysis would capture important metrics such as semantic value, or the meaning of the sequences generated. As a result, the rubric we rated by are the metrics (1) grammatical correctness/syntax, (2) coherence, and (3) adherence to book summary genre(s). For our rubric, we operationally define grammatical correctness/syntax as

correct punctuation, capitalization, and human-like text generation. Similarly, we define coherence as maintaining meaning or semantics over sequences of words, with little to no irrelevant or random words and phrases. Adherence to the genre is how correlated the words and phrases generated are to the input genre(s). To measure these values, we surveyed 14 volunteers to rate 3 text-generated book summaries with varying genres from each of the 4 models. Scores were limited to a range from 1 to 5, 1 representing low levels of the metric in the generated text to 5, high levels of the metric.

5.3 Experimental details

5.3.1 LSTM Model

To generate the best parameters for text-generation on book summaries, we did hyperparameter tuning on the LSTM model. We applied the AdamW optimizer with beta value 0.9 and epsilon value 10^{-8} as well as a learning rate scheduler such that the learning rate would be divided by 2 per epoch with no improvement. The best text-generation parameters for the LSTM model included:

- Epochs: 18
- Sequence Size: 50
- Batch Size: 124
- Embedding Size: 128
- Hidden Size (LSTM): 512
- Dropout rate: 0.5
- Learning Rate: 0.000125

To evaluate how well our model is doing each epoch, we use the cross-entropy loss function. We trained the LSTM by calculating the validation loss and comparing this to the training loss every epoch until the validation loss became stagnant. Training the LSTM on the dataset took around 2.5 hours.

5.3.2 GPT-2 Models

As part of HuggingFace's default `generate()` method, we included the decoding strategies discussed earlier which included 2 beams for beam search, penalties for 2-gram repetition, early stopping, and inclusion of multinomial sampling.

Next, our fine-tuning process with 3 and 5 epochs utilized default training arguments and GPU optimization techniques. We applied the AdamW optimizer with beta value 0.9 and epsilon value 10^{-8}. These training arguments also include enabling gradient checkpoint and mixed precision training, along with gradient accumulation happening every 4 steps.

Using the HuggingFace Trainer object, we trained the GPT-2 model for the desired number of epochs (3 or 5) at which the model halted training and saved its fine-tuned model configurations for later use. We were able to fine-tune our GPT-2 model using only 8 GB of GPU memory, but having the trade-off of slightly longer training times. Training the GPT-2 model on the book summary dataset took about 40 minutes using 3 epochs and 1.25 hours using 5 epochs.

5.4 Results

First, we compare the test loss and perplexity scores of each model.

Test Loss and Perplexity Scores		
Model	Test Loss	Test Perplexity
LSTM	4.953	141.550
GPT-2 (Vanilla)	2.299	9.961
GPT-2 (3 Epochs)	**1.803**	**6.073**
GPT-2 (5 Epochs)	1.816	6.149

Table 2: Test Loss and Perplexity Scores

We find that the baseline LSTM model does not do as well as any of the GPT-2 models as the test loss and perplexity scores are much higher than any of the GPT models. Overall, the GPT-2 models perform similarly to each other, but the fine-tuned GPT-2 models achieve better loss and perplexity values than the pretrained GPT-2 model.

Next, we compare the BERTScore for each of the GPT-2 models.

BERTScore			
Model	Precision	Recall	F1-Score
GPT-2 (Vanilla)	0.822	0.796	0.808
GPT-2 (3 Epochs)	**0.854**	**0.819**	**0.836**
GPT-2 (5 Epochs)	0.849	0.818	0.833

Table 3: GPT-2 BERTScores

The vanilla GPT-2 model trails behind both fine-tuned GPT-2 models. However, surprisingly, the GPT-2 model fine-tuned on 3 epochs has higher BERTScore precision, recall, and F1 scores compared to the GPT-2 model fine-tuned on 5 epochs. However, these differences are minute compared to the difference with the vanilla GPT-2 model.

Finally, let us compare the human evaluations for each model regarding their text generation.

Human Evaluation Metrics			
Model	Syntax	Semantics	Genre Adhered
LSTM	1.769 ± 0.788	1.564 ± 0.729	2.077 ± 0.881
GPT-2 (Vanilla)	4.590 ± 0.679	4.179 ± 0.937	2.564 ± 1.485
GPT-2 (3 Epochs)	4.641 ± 0.519	4.205 ± 0.778	3.923 ± 1.227
GPT-2 (5 Epochs)	$\mathbf{4.718 \pm 0.536}$	$\mathbf{4.538 \pm 0.577}$	$\mathbf{4.461 \pm 0.895}$

Table 4: Human Evaluation Metrics

We find that the GPT-2 model fine-tuned on 5 epochs does the best when compared with all other models.This result is thought-provoking as it conflicts with the machine-centric metrics. Otherwise, we find that, once again, the LSTM model does the worst in all categories, but interestingly, it almost scores as well as vanilla GPT-2 in adhering to the genre given. In addition, there seems to be more variability in rating the "adherence to genre" for the all GPT-2 models. We explore this further in our analysis.

6 Analysis

Regarding the results of the baseline LSTM model: from our data, LSTM scores much higher in perplexity than the GPT-2 models. This is also inline with the human evaluation metrics of "Coherence" and "Grammatically correct/Syntax" as the story summary outputs from the GPT-2 models resulted in higher human evaluation ratings. However, the human evaluations for "Adherence to Genre" for LSTMs were decently higher than scores for "Coherence" and "Grammatically correct/Syntax". This result may indicate that the words generated are seemingly correlated with the input genre, even if syntax or semantics are not well captured in the final output. This result is similar to the findings in Santhanam (2020) such that the embedded context is adhered to but much of the semantic value is lost. In comparison with the GPT models, we find that the LSTM is a less robust model with regard to retaining various features of textual information.

The GPT-2 models fared slightly better than LSTMs in both the machine-generated metrics and human evaluation. With losses between 1.8 to 2.3 and perplexity values under 10, the vanilla GPT-2 model along with the fine-tuned versions performed better than the LSTM. As for BERTScores, the GPT-2 fine-tuned for only 3 epochs performed well, although only marginally better than the model fine-tuned for 5 epochs. Precision across LSTMs and the GPT-2 models were all above 0.8 which shows high model performance in this text generation task.

Regarding human evaluation of the GPT-2 models, we saw a greater difference between the models than the machine-computed metrics outputted. Though helpful in model evaluation in comparison to other deep learning techniques, metrics like loss and BERTScore do not provide the whole story of model performance. Human evaluation in creative text generation tasks such as this one is the most important measures, as humans are the end readers of the generated text summaries.

As mentioned earlier, LSTMs scored worse than GPT-2 models in "Coherence", "Grammatically correct/Syntax", and "Adherence to Genre". The vanilla GPT-2 model scored similarly to the fine-tuned models in the metric for "Grammatically correct/Syntax" however, failed to adhere to prompted genres as well. This is the opposite finding from our LSTM model as in this case, not many generated words related to the genres but were well-written, which is validated by the fact that pre-trained GPT models uses human written text that does not have particular genre constraints on them. Observing example (A.2), we notice that the generated text from the vanilla, pretrained model does indeed have human-like fluency and summarize cohesive ideas relating to literature and children however, it is unrelated to genre prompt and the task of generating a book summary.

Because the GPT-2 models can produce generated text that is coherent and straightforward to read, it is much easier to determine the adherence of genre for evaluators. However, this metric is harder to measure as it is more subjectve to rate than the other two measures. As a result, it would make sense for evaluation ratings to be more variable and have a higher standard deviation. For the LSTM model and the GPT-2 model fine-tuned with 5 epochs, these standard deviations are much lower as they are on the ends of the range that could be scored.

In contrast, examples (A.3), (A.4) and (A.5) retain human-like fluency and coherence while also providing an actual story summary that adheres to the genre prompts. This improvement over the LSTM and vanilla GPT-2 model is attributed to the fine-tuning process but additionally, the decoding methods used in the text generation process. Unlike previous work that did not dedicate research to other decoder techniques, we included a discussion and exploration of beam search and other methods to properly produce text that succeeds in the task of genre prompted summarization. It achieves the goal of this project while simultaneously maintaining high levels of grammatical correctness and coherence, evidenced by the higher scores in human evaluation in all three metrics for the fine-tuned, optimized GPT-2 models.

As for possible errors in text generation, especially among the high performing GPT-2 model trained for 5 epochs, example (A.5) shows a weak adherence to the children's literature genre. While on average, this model scored 4.461 out of 5 for genre adherence across the three prompts with different genre constraints, it performed better on the science fiction and speculative fiction prompt as shown in (A.4) compared to (A.5). This could be due to our dataset used for fine-tuning having more book summaries in that particular genre grouping, as shown by Figure 6. This unbalanced dataset exposed the model to more examples of science fiction and speculative fiction book summaries in the fine-tuning process, hence the improved output text for this prompt.

Additionally, since we only focused on genre prompts, some details like character or plot events were overlooked by our model. For example, in (A.4), the first sentence generated mentions a human colony, although this subject is unrelated to the rest of the summary about a ship and alien groups. While holistically, the text generated by the GPT-2 models score high on human evaluation metrics, taking a closer look at the connections between subjects, verbs, and story events show that the model is not aware of particular linguistic relationships and is unable to stick to details like the original premise of a human colony. We can make the preliminary conclusion that our GPT-2 models are able to learn genre features of a story and master the fine-tuned task at hand, but cannot conduct any other higher-order story organization.

7 Conclusion

In completing this project, we aimed to develop a model, such that when inputted with a genre-specific story summary generation prompt, the model would be able to generate a passage that adheres to the genres, retains coherence of story points, and maintain human-like levels of fluency and grammatical correctness. To achieve this goal, we evaluated the performance of four different model setups: LSTMs, vanilla pretrained GPT-2, optimized GPT-2 fine-tuned on 3 epochs

and optimized GPT-2 fine-tuned on 5 epochs. The fine-tuned, optimized versions of GPT-2 included a combination of optimal generation strategies for the final decoder block along with memory usage minimization.

Using a dataset of book summaries along with their genres from Carnegie Mellon University to train and evaluate our models, we discovered that the optimized GPT-2 model fine-tuned with 3 epochs performed the best in quantitative measures of perplexity, BERTScores, and loss, though only trivially better than the 5 epoch version. However, the model fine-tuned for 5 epochs scored the highest among human evaluation metrics, which is arguably the most important evaluation in creative text generation tasks such as this one. The combination of these quantitative and qualitative metrics provide a comprehensive look at this goal of creative text generation, which could be implemented in future research to evaluate output text.

Moreover, using basic genre prompts and shorter book summaries to fine-tune a large language model for natural language generation was not studied in-depth before this project, as many other related works used large corpus of movie data to fine-tune GPT-2 to generate story text. All of our models achieved varying levels of success with outputting novel, genre-specific book summaries training on only shorter story summaries, which was the motivating goal behind this research. Building upon the work in this project, some future avenues to explore include attempting to improve our model by modifying the architecture of transformers themselves and further experimentation with other complex language generation models like Facebook's OPT and GPT-3.

References

Amal Alabdulkarim, Winston Li, Lara J. Martin, and Mark O. Riedl. 2021. Goal-directed story generation: Augmenting generative language models with reinforcement learning.

Jay Alammar. 2019. The illustrated gpt-2 (visualizing transformer language models).

JinUk Cho, MinSu Jeong, JinYeong Bak, and Yun-Gyung Cheong. 2022. Genre-controllable story generation via supervised contrastive learning. *ACM*, page 2839–284.

Colah. 2015. Understanding lstm networks.

Noureen Fatima, Ali Shariq Imran, Zenun Kastrati, Sher Muhammad Daudpota, and Abdullah Soomro. 2022. A systematic literature review on text generation using deep neural network models. *IEEE Access*, 10:53490–53503.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

HuggingFace. a. Fine-tune a pretrained model.

HuggingFace. b. Performance and scalability: How to fit a bigger model and train it faster.

HuggingFace. c. Text generation strategies.

Jasminyas. 2018. Cmu book summary dataset. `https://www.kaggle.com/datasets/ymaricar/cmu-book-summary-dataset`.

OpenAI. Better language models and their implications.

Patrick von Platen. 2020. How to generate text: using different decoding methods for language generation with transformers.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Illya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, rewon Child, David Luan, Dario Amodeii**, and Illya Sutskever**. 2019. Language models are unsupervised multitask learners.

Sivasurya Santhanam. 2020. Context based text-generation using lstm networks.

François St-Amant. How to fine-tune gpt-2 for text generation.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A Appendix

A.1 LSTM Sample Output

Generate a book summary with genre children's literature:

due the third central life , nobles introduces into a apparent shopping shop and his dismal brother , describing death back across their way with a girlish desert . his wife , math wakefield , is his experiences . when in the other world , rincewind has that jeremy had a contest , and commands the splendid , the rules . it also becomes apparent in the garden , and ends . as an fortified , molly tongue has here his essence to become advance of the gantry and became able to live to the seaside during the department .

A.2 Vanilla GPT-2 Sample Output

Generate a book summary with genre children's literature:

This is the first of a series of posts on the topic of genre kids' literature. This post first appeared in the March 2016 issue of the American Library Association's journal Science. The first post was titled "The Science of Science Fiction and Fantasy." The second post is titled Science and the Arts of Sci-Fi."

A.3 3 Epoch GPT-2 Sample Output

Generate a book summary with genre novel:

The novel is set in the life of a young woman in London. The story begins with her husband, who is in a car accident, and her mother, whom she does not know at the time. She is pregnant with their second child, but she cannot bear to leave them. After a long journey to London, she meets and falls in love with a man who lives nearby. He is a talented painter and painter's son. At first, he refuses to let her go because of her age. But when he is invited to join her at his home, the two become friends. They become lovers and eventually have a child together. Later, as she is leaving for work, they meet again

A.4 5 Epoch GPT-2 Sample Output 1

Generate a book summary with genre science fiction, speculative fiction:

The novel opens with the arrival of the first human colony on the planet. The ship, which has been in orbit for over two thousand years, is able to detect the presence of a mysterious alien race, the "Unseen Ones". They are a race of intelligent, intelligent beings that live in the solar system. In this novel, they are described as humanoid, bipedal, and have a distinct personality. They have long since disappeared from the surface world of Earth and are thought to have been wiped out by a violent alien invasion. However, their presence on Earth has left them in an uneasy truce with humanity. Humanity has not yet fully recovered from its isolation and has decided

A.5 5 Epoch GPT-2 Sample Output 2

Generate a book summary with genre children's literature:

The novel is set in the fictional town of Waverly, Maine, where the narrator is a young man who lives in a small town. The townspeople, including the town council, believe that there is no better place to live than the woods, which they call "Waverleys". However, the villagers are not so sure. They are convinced that the village is haunted by ghosts, and that they must find a way to solve the mystery of the hauntings. At the same time, there are rumors of a supernatural being called "The Witch", who is responsible for the haunting. This mysterious figure is referred to as the "White Witch" by the locals, who believe