Context-Aware Story Generation

Thakar, Viral Bankimbhai
vthakar@torontomu.ca
501213983

Gole, Montgomery
mgole@torontomu.ca
501156495

April 2023

Abstract

This paper presents a context-aware story generation system that utilizes both global and local context to create cohesive and natural stories. We use a curated dataset sourced from George R. R. Martin's *Fire and Blood* novel, as well as his series of novels called *A Song of Ice and Fire*, and fine-tune a Generative Pretrained Transformer (GPT) model for story generation. Our approach involves generating stories based on structured information consisting of characters, events, places and key words. We evaluate the quality of the generated text using both quantitative and qualitative measures. Our results show that incorporating both global and local context significantly improves the coherence and naturalness of generated stories. The contributions of this work include a curated dataset for story generation, a fine-tuning methodology for generating stories based on structured information, and an evaluation of the quality of the generated text. We found that roughly 50% of text generated by our model is more than 75% different from the original source while 50% of text generated by our model is 75% similar to the original source. We have used cosine-similarity based on MPNet-generated sentence embeddings to capture this interesting find.

1 Introduction

1.1 Motivation

The field of Natural Language Processing — NLP has made significant strides in recent years, with deep learning models like Generative Pretrained Transformer — GPT achieving impressive results in various language-related tasks, including language modeling and text generation [7]. However, generating coherent and engaging story text remains a challenging task for NLP models. While many techniques have been used to automate the generation of narrative texts, there is still a need to improve the quality of the generated stories, and to explore new approaches that can lead to more creative and innovative AI. Moreover, there is a growing interest in using AI-generated stories in various applications, including entertainment, education, and marketing. Therefore, there is a need for further research in this area to advance the state of the art and to develop new methods for context-aware story generation.

1.2 Research Objectives

The main objective of this research is to explore the effectiveness of GPT models for context-aware story generation based on structured information. To achieve this objective, we have set the following research objectives:

1. To curate a dataset of story excerpts from popular novels and to preprocess the dataset by cleaning the text and labeling it with relevant metadata, including characters, events, places and key words.
2. To fine-tune a GPT model using the preprocessed dataset a to improve its ability to generate high-quality, coherent story text.
3. To evaluate the performance of the fine-tuned model by measuring its ability to generate compelling and natural story text using both human and automated metrics.
4. To compare the performance of the fine-tuned model with original novel and to analyze the strengths and weaknesses of different approaches.

By achieving these research objectives, we aim to advance the state of the art in context-aware story generation and to contribute to the development of more creative and innovative AI. This paper is organized as follows: Section 2 provides an overview of related work, Section 3 describes our methodology in detail, Section 4 presents and discusses our results, and finally, Section 5 concludes the paper and outlines future directions for research.

2 Related Work

Various models used for story generation are described in [2], which are ranging from graph-based models and heuristic search approaches to machine learning (ML) models. Most ML-based models currently utilize Recurrent Neural Networks — RNN approaches for seq2seq models. Some ML-based methods do not necessarily create a full story but instead create abstractions of stories, such as [6] and [2], which predicts missing events in a script. Other ML efforts focus on creating statistical models to determine the association of a set of events and a single event. Story completion is another area of research where deep learning methods are used to generate story endings [5]. Researchers in this area recognize that understanding the unfinished story's plot is necessary to generate a proper ending. Story completion has also seen researchers attempt to complete missing sentences in stories, such as [10], who did so with GPT-2 [3].

As mentioned earlier, full-text story generation methods usually use RNN-based methods. However, their lack of memory can cause a lack of coherence in generated stories, which is a clear drawback of RNN-based methods. Transformer-based methods, such as those in [9], found a lack of diversity in GPT-2's potential for story generation when conditioned on prompts. It has also been found that Large Language Models — LLMs typically *"degenerate text by generating text that is bland, incoherent, or gets stuck in repetitive"* [2]. Therefore, it is interesting to question how the performance of these models improves when given more fine-grained, carefully curated key information. Specifically, given the apparent mid-performance of Large Language Models — LLMs like GPT-2 when trained on *prompt, story* pairs, it is worth investigating whether these models can improve when provided with more detailed and specific information.

3 Preliminaries

In this section, we provide a mathematical description of the GPT models and present the training process using pseudocode.

3.1 GPT Model

The GPT models are language models based on the Transformer architecture [**?**]. The core of the GPT models is a stack of N Transformer blocks, each consisting of a multi-head self-attention mechanism and a feedforward neural network. The model takes as input a sequence of tokens, and generates the next token in the sequence based on the previous tokens.

Formally, let $\mathbf{x} = (x_1, x_2, \ldots, x_T)$ be a sequence of T tokens, where x_i is the i-th token in the sequence. The GPT model predicts the probability distribution $p(x_{T+1}|\mathbf{x})$ over the next token x_{T+1} given the input sequence $\mathbf{x}$. The probability distribution is computed as follows:

$$p(x_{T+1}|\mathbf{x}) = \text{softmax}(\mathbf{W}_o \mathbf{h}_T), \tag{1}$$

where $\mathbf{h}_T$ is the final hidden state of the GPT model after processing the input sequence $\mathbf{x}$, and $\mathbf{W}_o$ is a weight matrix. The hidden state $\mathbf{h}_T$ is computed as follows:

$$\mathbf{h}t = \text{TransformerBlock}(\mathbf{h}t - 1), \quad t = 1, 2, \ldots, T, \tag{2}$$

where $\mathbf{h}_0$ is the initial hidden state, and TransformerBlock is a function that computes the output of a single Transformer block.

3.2 Training Process

The training process for the GPT models can be divided into two stages: pretraining and finetuning.

3.2.1 Pretraining

In the pretraining stage, the GPT model is trained on a large corpus of text using a language modeling objective. The goal is to learn a general language model that can generate coherent and natural-sounding text. The training process is summarized in Algorithm 1.

Algorithm 1 Pretraining of GPT model

```
procedure PRETRAIN(D)
    Initialize GPT model with random weights
    for each epoch do
        for each batch x in D do
            Compute forward pass through GPT model to obtain predicted distribution p(x)
            Compute loss L between predicted distribution and true distribution
            Compute gradients of loss with respect to model parameters
            Update model parameters using optimizer (e.g. Adam)
        end for
    end for
end procedure
```

The training data $\mathcal{D}$ consists of a large corpus of text, which is typically preprocessed by tokenization and truncation to fit the model's input size. The model is trained using maximum likelihood estimation (MLE), where the loss function is the negative log-likelihood of the training data.

4 Methodology

Context-aware story generation is an emerging field in natural language processing that focuses on generating coherent and engaging stories by incorporating contextual information. Contextual information can be of two types: global and local. Global context refers to the overarching themes, settings, and characters of a story, while local context refers to the specific details, actions, and events that occur within the story. By leveraging both global and local context, context-aware story generation aims to create cohesive and natural stories that are both engaging and satisfying for the reader. Through the use of sophisticated machine learning techniques, context-aware story generation is poised to revolutionize the field of creative writing and entertainment by providing a new way to generate compelling and immersive narratives. In this project we proposed a Context-Aware Iterative Approach for story generation - which is illustrated in the 1

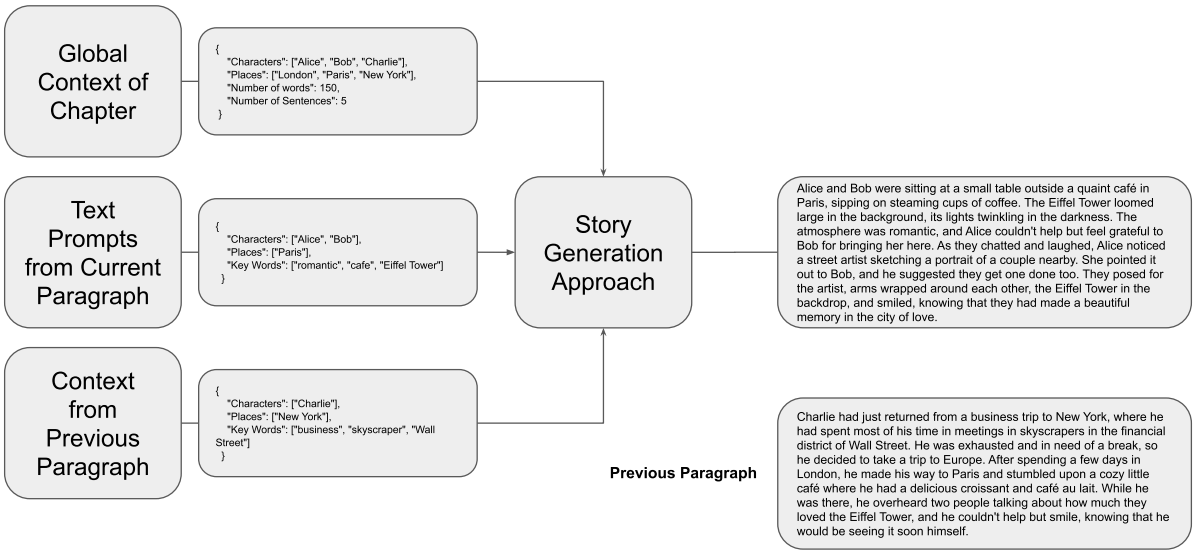


Figure 1: Generic System Diagram of Context Aware Story Generation

To achieve our goal of exploring the effectiveness of fine-tuning Generative Pretrained Transformer (GPT) models for story generation based on a custom dataset, we followed a methodology that involved several steps. First, we collected a dataset of story excerpts from George R. R. Martin's *Fire & Blood* novel, as well as his series of novels called *A Song of Ice and Fire*. This dataset was curated to ensure that it contained a diverse range of characters, settings, and events, while also maintaining a consistent narrative structure. Next, we preprocessed the dataset by cleaning the text and splitting it into smaller chunks — Paragraphs. We also labeled each chunk with relevant metadata, including characters, events, places and key words, to

enable structured story generation. We then fine-tuned a GPT-2 model using the preprocessed dataset and a combination of unsupervised and supervised learning techniques.

Finally, we evaluated the performance of the fine-tuned model by measuring its ability to generate compelling and natural story text. We conducted a human evaluation study in which participants read and rated a selection of generated stories based on their coherence, readability, and overall engagement. Overall, our methodology enabled us to explore the effectiveness of fine-tuning GPT models for story generation based on structured information, and to evaluate the quality of the generated text using both human and automated metrics.

4.1 Data Collection

This paper's data collection began with the *Fire and Blood* [8] novel. However, due to the limited nature of a book, and the large amount of inference needed to understand the attributes of characters and their relationships, this paper turned to online resources for data collection. Other than the *Fire and Blood* text, the main source of this paper's data collection came from the *Game of Thrones Fandom Wiki* [1]. From these wikis, the following data were collected:

- House of Dragons Episode Synopses
- House of Dragons' differences in its television adaptation from the *Fire & Blood* text
- House of Dragons Character Biographies, and Personalities
- House of Dragons Dragon Biographies

4.2 Dataset Curation

The data extracted from our full corpus fall into two main categories: the global context of each chapter, and key information from each paragraph. These data were extracted from a pretrained GPT-3.5's *text-davinci-003* model [3] and formatted into JSON using the following prompt:

Prompt Engineering

Prompt Prefix

"Extract characters, places, events, verbs and adjectives from the give text below. Return valid json following format "characters":[], "places":[], "events":[], "verbs":[], "adjectives":[]. Do not escape the double quotes in the output:"

Paragraph

Aegon I Targaryen was a warrior of renown, the greatest Conqueror in the history of Westeros, yet many believe his most significant accomplishments came during times of peace. The Iron Throne was forged with fire and steel and terror, it is said, but once the throne had cooled, it became the seat of justice for all Westeros.


```
{
    "paragraph_id": 0,
    "paragraph": "",
    "valid": 1.0,
```

```
    "characters": [
        "Aegon I Targaryen"
    ],
    "places": [
        "Westeros",
        "Iron Throne"
    ],
    "events": [],
    "verbs": [
        "forged",
        "cooled"
    ],
    "adjectives": [
        "renown",
        "greatest",
        "significant",
        "fire",
        "steel",
        "terror"
    ],
}
```


Listing 1: Extracted Metadata

Specifically, from each chapter, and its paragraphs, the key information extracted was: characters, places, events, verbs, and adjectives. The verbs and adjectives were extracted in order to attempt to capture the theme, and the writing style of the *Fire & Blood* text. The extracted data can is visualized in 4.2.

Typical text cleansing techniques like the removal of stop words, and lemmatization/stemming are not necessary for this paper's task. This is because LLMs are trained to understand the semantic structure of sentences, and find the relationship between words, punctuation, and inflected forms of words. However, human intervention was necessary when cleaning the paragraphs extracted from the *Fire and Blood* text to ensure that each paragraph followed a similar structure. This simple structure requires that each paragraph is on a separate line from the other paragraph; begins with a capital letter, and ends with a period.

4.3 Models

In this research, we utilized two Generative Pretrained Transformer (GPT) models - GPT-3[3] and GPT-3.5[4]. The GPT models are based on the transformer architecture, which allows for parallel processing of the input data and is capable of generating high-quality natural language text.

The training process involved fine-tuning the pre-trained GPT models on our custom dataset. We used a curated dataset consisting of narrative texts sourced from novels, including George R R Martin's *Fire and Blood* and his series of novels called *A Song of Ice and Fire*. Our approach takes a iterative and context-aware approach to story generation, where the context of the overall chapter and the previous paragraph is given as input for each paragraph being generated. This enables our generation model to generate more story from previously mentioned events, places, and characters, resulting in a more cohesive and natural story.

The loss function used during training was perplexity, which is a standard measure used for language modeling. The objective of minimizing perplexity is to minimize the exponentiation of the negative average log probability of each predicted token or the exponentiation of the cross-entropy between the predicted distribution of tokens and the real distribution. The models were trained using the Adam optimizer with a

learning rate of 1e-4 and a batch size of 256. The training was conducted on a single GPU for a total of 10 epochs.

5 Results and Discussions

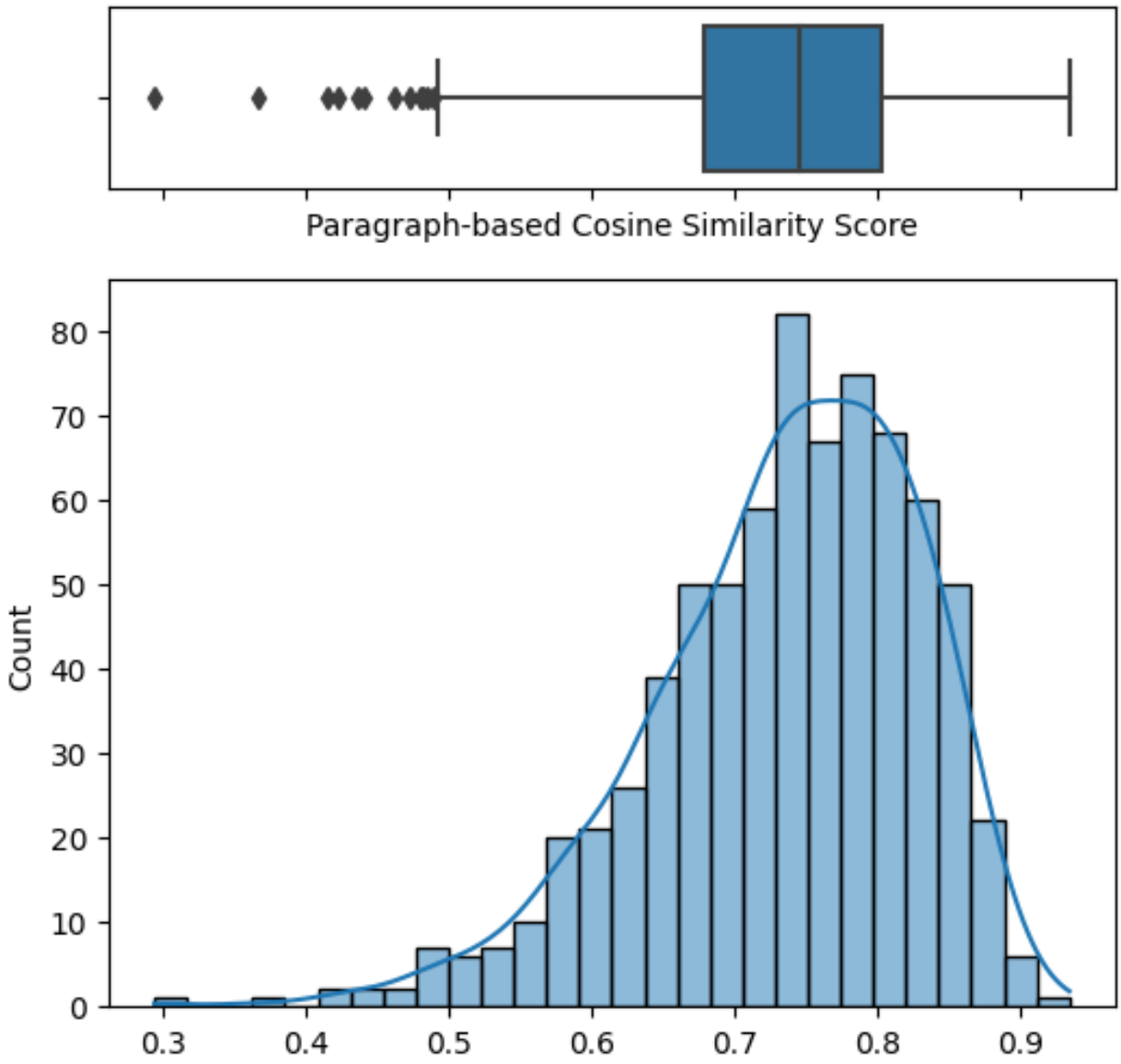


Figure 2: Distribution of Cosine Similarity Between Original Book and Generated Novel

To evaluate the performance of our story generation model, we used the distribution of cosine similarity between the MPnet generated, 768-dimension sentence embeddings for the generated text and the original source. Cosine similarity is a measure of the similarity between two vectors of an inner product space and is commonly used in natural language processing tasks.

As indicated in 2, We found that approximately 50% of the text generated by our model had a cosine

similarity of more than 75% with the original source, indicating that our model was able to generate text that closely resembled the input. On the other hand, the remaining 50% of the text generated had a cosine similarity of less than 75%, indicating that our model was able to generate unique and diverse text.

This finding is important because it suggests that our model was able to strike a balance between generating text that was similar to the input and text that was unique and original. This is a desirable trait for a story generation model, as it ensures that the generated text is both coherent and engaging.

Furthermore, the distribution of cosine similarity indicates that our model was able to generate a wide range of text that varied in its similarity to the original source. This is important because it suggests that our model was not simply regurgitating the input but was instead generating new and interesting content.

Overall, the distribution of cosine similarity demonstrates that our model was able to generate high-quality text that was both similar to the input and unique. This is an important step towards developing a robust and effective story generation model. Further samples of results and plots are provided under the appendix section for reference.

6 Limitations & Next steps

A glaring limitation of our approach revolves around the usage of GPT models. The fact that they were trained on data from the web past 2018, means that the *Fire & Blood* text could have been present in their training data which may make it more difficult, or easier for the models to create stories from our custom dataset. This leads to the proposition of a next step which includes replicating this methodology with a novel that came after 2021, as the GPT models tested were trained on data up until 2021.

7 Conclusion

In conclusion, this research presents a novel approach to context-aware story generation using GPT-3 and GPT-3.5 models. By incorporating global and local context information, our model is capable of generating cohesive and natural stories. The evaluation of our model using cosine similarity shows that it is capable of generating unique but similar texts.

There are several avenues for future work in this area. One potential direction is to explore the use of additional context sources, such as images or audio, to improve the quality of the generated stories. Another direction is to investigate the use of more sophisticated evaluation metrics to assess the coherence and structure of the generated stories. Additionally, exploring the use of more advanced language models or fine-tuning the existing ones may improve the quality of the generated stories further. Finally, the potential applications of context-aware story generation are vast, and it would be interesting to investigate how this technology can be used to enhance creative writing, entertainment, and even education.

References

[1] Game of thrones wiki.

[2] Arwa I Alhussain and Aqil M Azmi. Automatic story generation: a survey of approaches. *ACM Computing Surveys (CSUR)*, 54(5):1–38, 2021.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[4] Mingda Chen and Kevin Gimpel. Tvstorygen: A dataset for generating stories with character descriptions. *arXiv preprint arXiv:2109.08833*, 2021.

[5] Jian Guan, Yansen Wang, and Minlie Huang. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6473–6480, 2019.

[6] Bram Jans, Steven Bethard, Ivan Vulic, and Marie-Francine Moens. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 336–344. ACL; East Stroudsburg, PA, 2012.

[7] Holy Lovenia, Bryan Wilie, Romain Barraud, Samuel Cahyawijaya, Willy Chung, and Pascale Fung. Every picture tells a story: Image-grounded controllable stylistic story generation. *arXiv preprint arXiv:2209.01638*, 2022.

[8] George R. R. Martin. *Fire & blood*. Bantam, New York, 2018.

[9] Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D Manning. Do massively pretrained language models make better storytellers? *arXiv preprint arXiv:1909.10705*, 2019.

[10] Su Wang, Greg Durrett, and Katrin Erk. Narrative interpolation for generating and understanding stories. *arXiv preprint arXiv:2008.07466*, 2020.

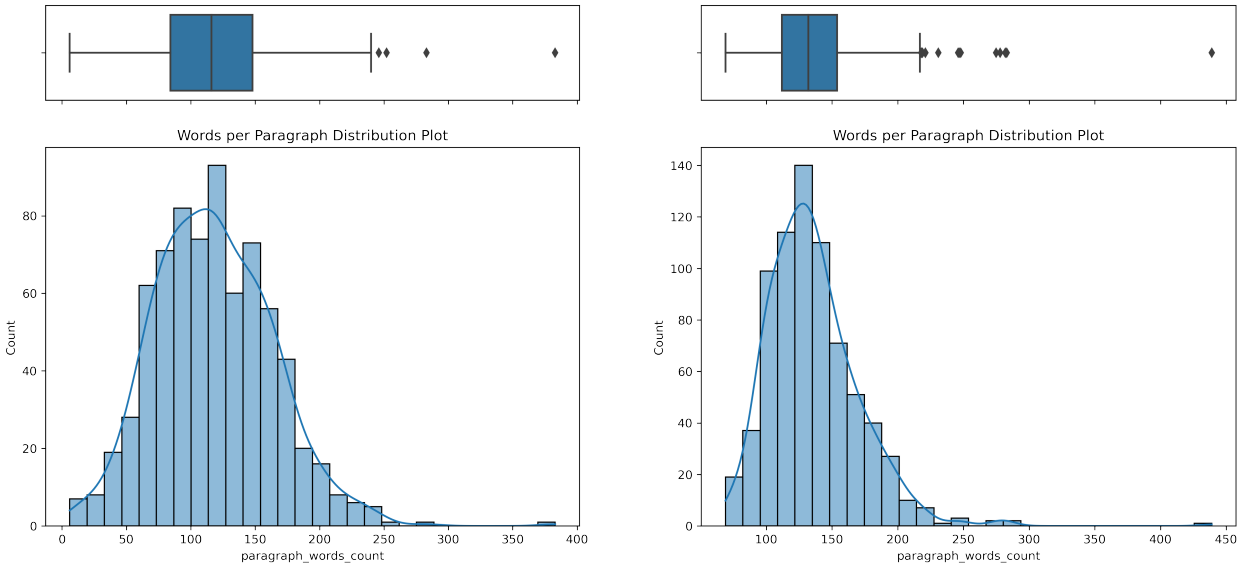
A Sample Results

Table 1: Sample Results

Fire and Blood	**Generated**
Aegon I Targaryen was a warrior of renown, the greatest Conqueror in the history of Westeros, yet many believe his most significant accomplishments came during times of peace. The Iron Throne was forged with fire and steel and terror, it is said, but once the throne had cooled, it became the seat of justice for all Westeros.	Aegon I Targaryen was renowned for forging something greater than himself. He cooled six realms of deadly fire with his sword made from steel, he rode a dragon that had terrorized Westeros for centuries, and on the Iron Throne of Westeros he sat. The significance of his legacy stretched to the far ends of the realm, and his accomplishments are still remembered as one of the greatest in history.
The reconciliation of the Seven Kingdoms to Targaryen rule was the keystone of Aegon I's policies as king. To this end, he made great efforts to include men (and even a few women) from every part of the realm in his court and councils. His former foes were encouraged to send their children (chiefly younger sons and daughters, as most great lords desired to keep their heirs close to home) to court, where the boys served as pages, cupbearers, and squires, the girls as handmaidens and companions to Aegonǎ2019s queens. In Kingǎ2019s Landing, they witnessed the kingǎ2019s justice at first hand, and were urged to think of themselves as leal subjects of one great realm, not as westermen or stormlanders or northmen.	Aegon I was determined to bring former enemies into the fold in order to reconcile the Seven Kingdoms to Targaryen rule. He made great efforts and travelled far across Westeros, urging people at every corner of the realm, from Duskendale and Gulltown to Evenfall Hall and Tarth, to serve him. His famed courage shown through fire and steel had cemented his renown even before he sat on the Iron Throne with Rhaenys by his side. As people witnessed his greatness first-hand, many were encouraged by Good Queen Alysanne's leal pardons or King Jaehaerys' conquests during The Great Council of Harrenhal, so much so that Aegon received gifts from all across - from a Redwyne girl sending Aegonfort's stones for building-up Dragonstone castle, or a bastard born man gifting Visenya's Hill for turning it into a royal residence - as a token of their former loyalties soon turned loyalty to House Targaryen.

The Targaryens also brokered many marriages between noble houses from the far ends of the realm, in hopes that such alliances would help tie the conquered lands together and make the seven kingdoms one. Aegon’s queens, Visenya and Rhaenys, took a special delight in arranging these matches. Through their efforts, young Ronnel Arryn, Lord of the Eyrie, took a daughter of Torrhen Stark of Winterfell to wed, whilst Loren Lannister’s eldest son, heir to Casterly Rock, married a Redwyne girl from the Arbor. When three girls, triplets, were born to the Evenstar of Tarth, Queen Rhaenys arranged betrothals for them with House Corbray, House Hightower, and House Harlaw. Queen Visenya brokered a double wedding between House Blackwood and House Bracken, rivals whose history of enmity went back centuries, matching a son of each house with a daughter of the other to seal a peace between them. And when a Rowan girl in Rhaenys’s service found herself with child by a scullion, the queen found a knight to marry her in White Harbor, and another in Lannisport who was willing to take on her bastard as a fosterling.	Aegon and his sisters Visenya and Rhaenys brokering matches across the Seven Kingdoms to ensure their rule was secure. Aegon took a special delight in sealing these matches as it ensured loyalty from each of the seven kingdoms. Notable matches made included the double wedding between House Blackwood and House Bracken, as well as another one, involving Lord Ronnel Arryn of The Eyrie with a Redwyne girl, at The Arbor. Torrhen Stark of Winterfell married an Evenstar of Tarth in a lavish ceremony at White Harbor for all rivaled to see. Queen Visenya then arranged for marriages between noble families far and wide; from Lannisport to Casterly Rock to Tarth, House Corbray, Hightower and Harlaw were all sealed by alliance through betrothals brokered by her own hands. In addition, a Rowan girl was found with child by a scullion shortly after this flurry of marriage was arranged; she had been taken into fosterling by Orys Baratheon’s house but she would remain forever aloof from her betrothed due to the bastard she bore.
Though Aegon had designated King’s Landing as his royal seat and installed the Iron Throne in the Aegonfort’s smoky longhall, he spent no more than a quarter of his time there. Full as many of his days and nights were spent on Dragonstone, the island citadel of his forebears. The castle below the Dragonmont had ten times the room of the Aegonfort, with considerably more comfort, safety, and history. The Conqueror was once heard to say that he even loved the scent of Dragonstone, where the salt air always smelled of smoke and brimstone. Aegon spent roughly half the year at his two seats, dividing his time between them.	During Aegon’s reign, the designated seat of power was moved from Dragonstone to a more comfortable and safe location. King’s Landing had much historical importance since it was close to his conquest in the War for Seven Kingdoms one. Aegon had several castles built surrounding this area, such as Äegonforẗbetween King’s Landing and Dragonmont, which he spent much time installing himself and had to be ten times reinforced with its walls lined with salt. As part of his love for royal customs, he even changed its name from Smoky Castle to Äegonfort.̈In addition, nearly all the resources were spent on constructing a castle at Dragonstone. An enormous smoky tower known as Dragonmont now stands tall there where Aegon Targaryen once ruled over Westeros with Rhaenys and Visenya by his side before reluctantly being doubted by some of his own people.

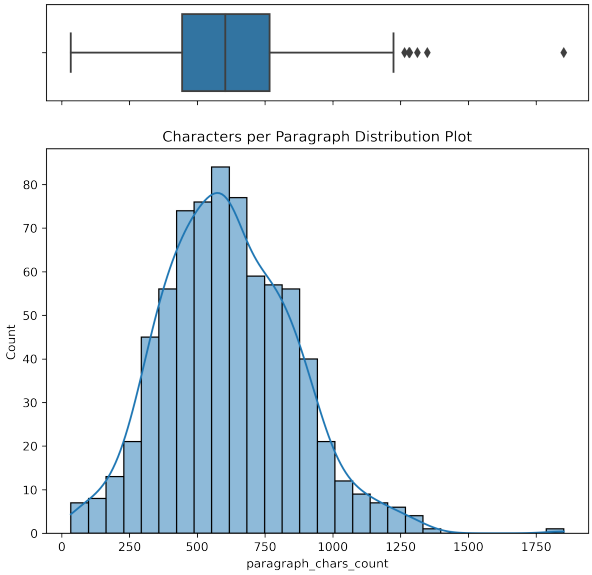
B Exploratory Data Analysis of Generated Novel



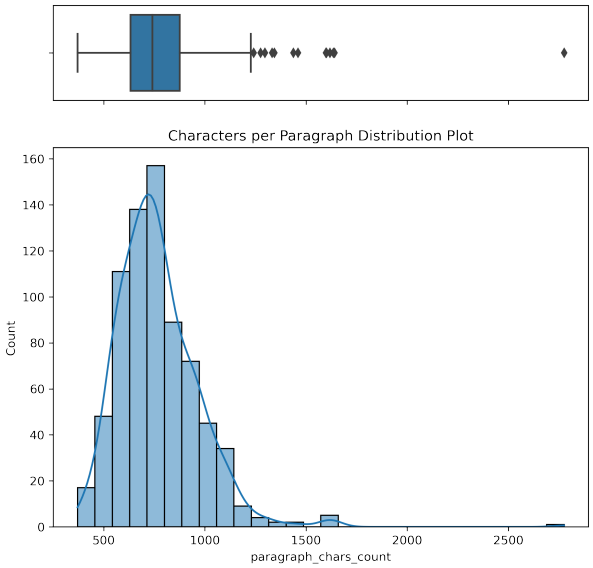
(a) Original Novel

(b) Generated Novel

Figure 3: Distribution of Words per Paragraph

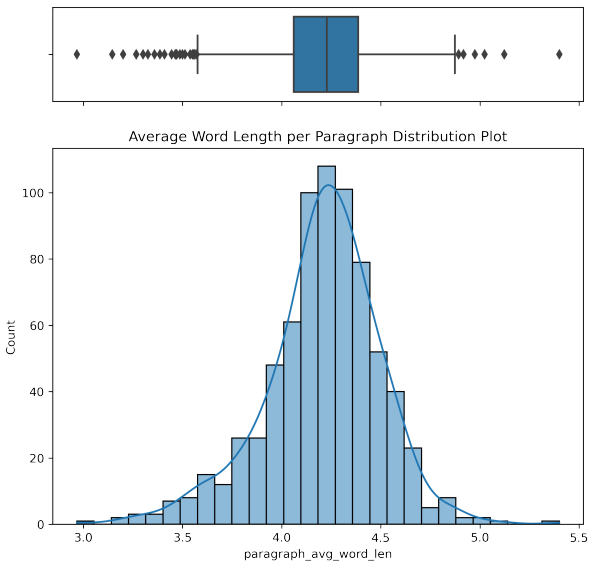


(a) Original Novel

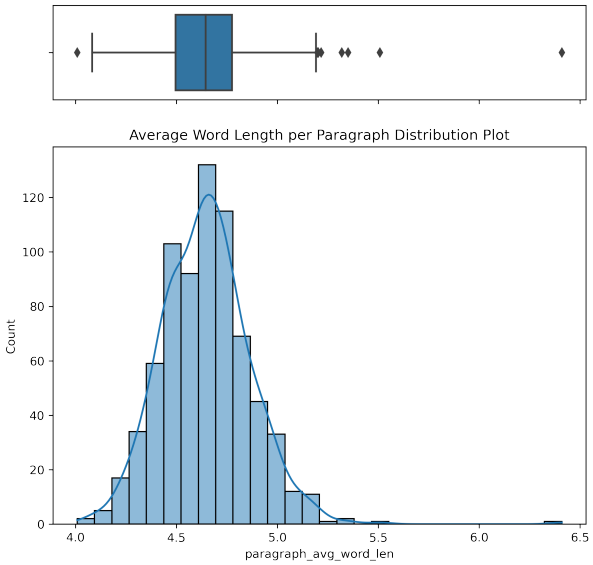


(b) Generated Novel

Figure 4: Distribution of Characters per Paragraph

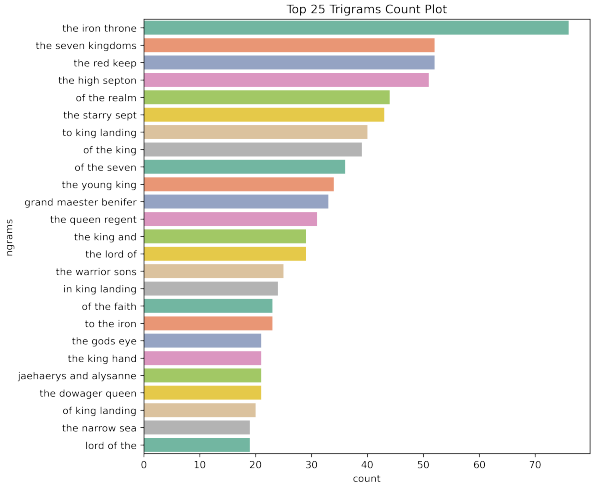


(a) Original Novel

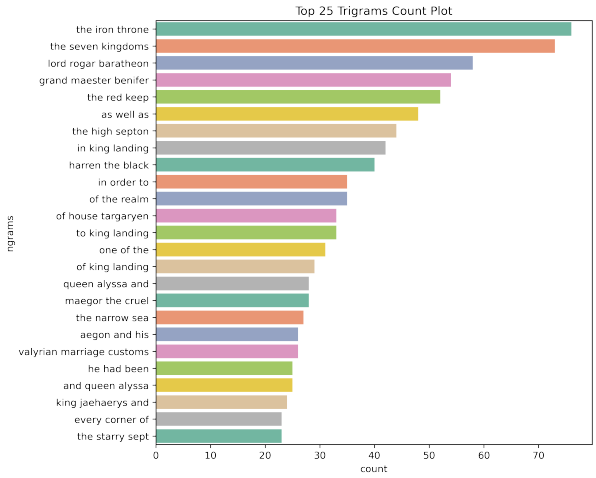


(b) Generated Novel

Figure 5: Distribution of Average Word Length per Paragraph



(a) Original Novel



(b) Generated Novel

Figure 6: Count plot of Trigrams