Final_Report_draft_final-2.docx

by Paresh Pradhan

Submission date: 27-Feb-2023 05:05PM (UTC+0000)
Submission ID: 198085350
File name: Final_Report_draft_final.docx (1.73M)
Word count: 31635
Character count: 185189

Generating Stories by Prompting Pre-trained Language Models

Paresh Pradhan
Student ID: 1038004

Thesis Report for
Master of Science in Machine Learning & Artificial Intelligence

Liverpool John Moores University & upGrad

March 2023

ACKNOWLEDGEMENTS

I would like to thank my supervisor Dr. Anil Kumar Vuppala for all his help and advice with this thesis. I would also like to thank my parents; without whose support this would have not been possible. I also appreciate all the support I received from the other students in by batch. Lastly, I would like to thank the LJMU and Upgrad for the studentship that allowed me to conduct this thesis.

ABSTRACT

Automatic Story Generation task has been studied for many angles. But with the advent of Large Language Models and their ability to use Prompt-Learning to generate coherent text in a Zero-Shot & Few-Shot manner has taken the Automatic Story Generation task in entirely new directions. Although the Large Language Models have the capability to generate coherent text, tasks like Controllable Story Generation, where the generation needs to follow a connected chain of events, are still a challenge. This is the problem that this thesis tries to solve. This thesis discusses the prior work done for Automatic Story Generation and Prompt-Learning. The thesis explores Prompt-Learning, In-Context Learning and Instruction Tuning capabilities of Large Language Models and then uses these techniques to generate stories. The generated stories are evaluated across different criteria including size and type of the Language Models and using different evaluation criteria – both qualitatively and quantitatively. The thesis discusses the successes, failures and what could be improved in the future.

TABLE OF CONTENTS

LIST OF TABLES i
LIST OF FIGURES ii
LIST OF ABBREVIATIONS iii
CHAPTER 1: INTRODUCTION 12
1.1 Background of the Study 12
1.2 Research Questions 13
1.3 Aim & Objectives 13
1.4 Significance of the Study 14
1.5 Scope of the Study 14
1.6 Structure of the Study 15
CHAPTER 2: LITERATURE REVIEW 16
2.1 Introduction 16
2.2 Automatic Story Generation 16
2.2.1 Introduction 16
2.2.2 Structural Models 17
2.2.3 Planning Based Models 17
2.2.4 ML Models 18
2.2.4.1 Story Abstraction 18
2.2.4.2 Script Learning and Generation 18
2.2.4.3 Story Completion 20
2.2.4.4 Story Generation 21
2.2.5 Story Evaluation 24
2.2.6 Challenges in Automatic Story Generation 27
2.3 Prompt Based Learning 29
2.3.1 Evolution of Prompting in NLP 29
2.3.2 Introduction to Prompting 30
2.3.2.1 Prompt Addition 30
2.3.2.2 Answer Search 31
2.3.2.3 Answer Mapping 31
2.3.2.4 Design Considerations for Prompting 31
2.3.3 Pre-trained Language Models 32

2.3.3.1 Training Objectives 32
2.3.3.2 Noising Functions 32
2.3.3.3 Directionality of Representations 33
2.3.3.4 Typical Pre-training Methods 33
2.3.4 Prompt Engineering 35
2.3.4.1 Prompt Shape 35
2.3.4.2 Manual Template Engineering 35
2.3.4.3 Automated Template Engineering 35
2.3.5 Answer Engineering 36
2.3.5.1 Answer Shape 36
2.3.5.2 Answer Shape Design Methods 37
2.3.6 Training Strategies for Prompting Methods 38
2.3.6.1 Training Settings 38
2.3.6.2 Parameter Update Methods 39
2.3.7 Prompt-relevant Topics 40
2.3.7.1 Few-shot Learning 41
2.3.7.2 QA-based Task Formulation 41
2.3.7.3 Controlled Generation 41
2.3.7.4 Data Augmentation 42
2.3.8 Challenges in Prompting 43
2.3.8.1 Prompt Design 43
2.3.8.2 Answer Engineering 43
2.3.8.3 Selection of Tuning Strategy 44
2.3.8.4 Multiple Prompt Learning 44
2.3.8.5 Selection of Pre-trained Models 44
2.3.8.6 Transferability of Prompts 45
2.3.8.7 Combination of Different Paradigms 45
2.3.8.8 Calibration of Prompting Methods 45
2.4 Summary 47
CHAPTER 3: RESEARCH METHODOLOGY 48
3.1 Introduction 48
3.2 Algorithms & Techniques 48

3.2.1 Pre-Trained Language Models 48
3.2.1.1 GPT-2 48
3.2.1.2 GPT-3 49
3.2.1.3 GPT-3.5 50
3.2.1.4 OPT 50
3.2.1.5 OPT-IML 51
3.2.1.6 BLOOM 51
3.2.1.7 BLOOMZ 52
3.2.1.8 LaMDA & PaLM 52
3.2.2 n-Shot Learning 53
3.3 Methodology 54
3.3.1 End-to-End Pipeline 54
3.3.2 Data Description 55
3.3.3 Data Preparation 55
3.3.3.1 ROCStories 55
3.3.3.2 WritingPrompts 56
3.3.4 Implementation 57
3.3.4.1 Generate Prompt from Outline 57
3.3.4.2 Generate Story from Prompt 58
3.3.5 Evaluation 59
3.4 Tools 60
3.4.1 Software 60
3.4.2 Hardware 61
3.5 Summary 61
CHAPTER 4: IMPLEMENTATION 62
4.1 Introduction 62
4.2 Data Preparation 62
4.3 Story Generation 65
4.3.1 Steps for Story Generation 65
4.3.2 Zero-Shot Story Generation 68
4.3.3 One-Shot & Few-Shot Story Generation 68
4.4 Summary 69

CHAPTER 5: RESULTS & EVALUATION 70
5.1 Introduction 70
5.2 Quantitative Evaluation 70
5.3 Qualitative Evaluation 73
5.3.1 Generated Story Example – 1 73
5.3.2 Generated Story Example – 2 76
5.4 Summary 80
CHAPTER 6: CONCLUSION & RECOMMENDATIONS 81
6.1 Discussion and Conclusion 81
6.1.1 Best Model for Story Generation 81
6.1.2 Quantitative Metrics for Story Generation 81
6.2 Contribution 81
6.3 Future Work 82
6.3.1 Benchmark against Other Models 82
6.3.2 Benchmark against Existing Approaches 82
6.3.3 Use Better Evaluation Metrics 82
REFERENCES 84
APPENDIX A: RESEARCH PROPOSAL 105

LIST OF TABLES

Table 5.2.1: Evaluation Scores

LIST OF FIGURES

Figure 3.2.1.1: GPT-3 Model Sizes

Figure 3.3.1.1: E2E Pipeline

Figure 3.3.3.1: ROCStories Outline Generation

Figure 3.3.3.2: WritingPrompts Outline Generation

Figure 3.3.4.1: Zero-Shot Prompt

Figure 3.3.4.2: Few-Shot Prompt

Figure 3.3.4.3; Prompting for Story Generation

Figure 4.2.1: Data Preparation Steps

Figure 4.2.2: Sample Outline

Figure 4.2.3: Sample Prompt

Figure 4.3.1.1: Story Generation Steps

Figure 5.2.1: Evaluation Scores Plot

Figure 5.3.1.1 Outline – 1

Figure 5.3.1.2 Generated Story: GPT2 – 1

Figure 5.3.1.3 Generated Story: Bloomz-1B7 – 1

Figure 5.3.1.4 Generated Story: Ada – 1

Figure 5.3.1.5 Generated Story: Babbage – 1

Figure 5.3.1.6 Generated Story: Curie – 1

Figure 5.3.1.7 Generated Story: Davinci – 1

Figure 5.3.2.1 Outline – 2

Figure 5.3.2.2 Generated Story: GPT2 – 2

Figure 5.3.2.3 Generated Story: Bloomz-1B7 – 2

Figure 5.3.2.4 Generated Story: Ada – 2

Figure 5.3.2.5 Generated Story: Babbage – 2

Figure 5.3.2.6 Generated Story: Curie – 2

Figure 5.3.2.7 Generated Story: Davinci – 2

LIST OF ABBREVIATIONS

AI: Artificial Intelligence

API: Application Programming Interface

ASG: Automatic Story Generation

BERT: Bidirectional Encoder Representations from Transformers

BLEU: BiLingual Evaluation Understudy

BLOOM: BigScience Large Open-science Open-access Multilingual Language Model

CBT: Children's Book Test

CC: Common Crawl

CIDEr: Consensus-based Image Description Evaluation

CLM: Causal Language Modeling

CTR: Corrupted Text Reconstruction

EACS: Embedding Average Cosine Similarity

FLAN: Finetuned Language Models

FTR: Full Text Reconstruction

GAN: Generative Adversarial Network

GENCI: Grand Équipement National de Calcul Intensif

GMS: Greedy Matching Score

GPT: Generative Pre-trained Transformer

GPU: Graphics Processing Unit

HANNA: Human-ANnotated NArratives

HTML: HyperText Markup Language

IDRIS: Institute for Development and Resources in Intensive Scientific Computing

IML: Instruction Meta-Learning

L2R: Left-to-Right

LAMA: LAnguage Model Analysis

LaMDA: Language Model for Dialogue Applications

LLM: Large Language Model

LM: Language Model

LSTM: Long Short-Term Memory

MCNC: Multiple Choice Narrative Cloze

METEOR: Metric for Evaluation of Translation with Explicit ORdering

ML: Machine Learning

MLM: Masked Language Model

NER: Named Entity Recognition

NLG: Natural Language Generation

NLP: Natural Language Processing

OOV: Out-of-Vocabulary

OPT: Open Pre-trained Transformer

PaLM: Pathways Language Model

PLM: Pre-trained Language Model

PMI: Pointwise Mutual Information

PPL: Perplexity

RAKE: Rapid Automatic Keyword Extraction

RLHF: Reinforcement Learning from Human Feedback

RNN: Recurrent Neural Network

RoBERTa: Robustly Optimized BERT Pretraining Approach

ROUGE: Recall-Oriented Understudy for Gisting Evaluation

SCT: Story Cloze Test

STCS: Skip Thoughts Cosine Similarity

SMT: Statistical Machine Translation

USPTO: United States Patent and Trademark Office

VECS: Vector Extrema Cosine Similarity

xP3: Crosslingual Public Pool of Prompts

CHAPTER 1: INTRODUCTION

1.1 Background of the Study

In recent years, with the emergence of large Pre-trained Language Models (PLMs), the quality of machine-generated text has improved significantly (Rashkin et al., 2018; Radford et al., 2019; Zhang et al., 2019b; Brown et al., 2020; Guan et al., 2020; Bakhtin et al., 2021). Today, models can generate text that is indistinguishable from human-written text (Clark et al., 2021).

Controlling the generation is still a challenge however, even though large-scale PLMs have demonstrated excellent capabilities in producing coherent and intelligible text (Keskar et al., 2019; Radford et al., 2019; Zellers et al., 2019). A more thorough examination of generated text reveals problems like topic drift and self-contradiction (Fan et al., 2019; Bisk et al., 2020; Dou et al., 2020; Gao et al., 2020a; Tan et al., 2020; Dziri et al., 2021). These flaws stand out in open-ended text generation tasks that require a high level of coherence, such as story generation.

Stories generated using language models have shown to lack discourse coherence (Bosselut et al., 2018; Ji and Huang, 2021), global planning (Hua and Wang, 2020; Tan et al., 2020) and common-sense knowledge (Ji et al., 2020; Xu et al., 2020). While the individual sentences in a generated text seem logical and fluent, when put together, the overall story often does not make much sense (See et al., 2019; Goldfarb-Tarrant et al., 2020). In long-form text generation, sentences tend to repeat which leads to reduction in story quality (Yao et al., 2019).

To provide structure to the generation process, recent works have tried to use explicit content planning. The content plan comes in different forms. (Fan et al., 2018) used prompts. (Xu et al., 2018; Yao et al., 2019) used keywords and key-phrases. (Fan et al., 2019) used semantic frames. (Sun et al., 2020) used summaries. To make use of these content plans, PLMs generally require fine-tuning on content-plan related data. A challenge with fine-tuning PLMs is that, in addition to needing training data, the model tends to learn frequently occurring events from the content plan and derives common sense information from them (Fan et al., 2019). This leads to lack of variety in generated stories.

Another novel approach to address linguistic issues without fine-tuning is provided by the recently proposed prompt-based learning (Liu et al., 2021a). In this framework, task-specific

prompts can be used to address text-based problems. Researchers have shown that using prompts, PLMs can solve existing or new generation tasks without need for fine-tuning (Brown et al., 2020; Li and Liang, 2021).

Although prompt-based learning looks promising, there are still some challenges. Prompts are highly task-specific and are hard to transfer or reuse for new tasks (Gao et al., 2020c). Even for the same task the prompts may not work well for all instances in a large population (Le Scao and Rush, 2021).

To deal with challenges in fine-tuning, this work tries to use prompt-based learning for story generation and in-context examples to provide control signal for the generation.

1.2 Research Questions

This thesis tries to answer the following questions:

1. The approaches for story generation with fine-grained control require fine-tuning of PLMs. Can these approaches be used with Prompt-based learning to generate stories in a Few-Shot manner without fine-tuning?
2. The previous methods largely use GPT2 as base model. Can using the latest generation GPT3 (or alternatives) improve the text generation capabilities?
3. Prompt-based learning has been used to generate text in a zero-shot & few-shot manner. Can this be extended to story generation task?

1.3 Aim & Objectives

This work tries to use Prompt-Learning and explore the Zero-Shot & Few-shot capabilities of GPT3 (& alternatives) for controllable story generation task.

Objectives:

- To conduct a comprehensive review of available literature with regards to Automatic story generation and Prompt-Learning.

- To explore the viability and then develop a method to generate stories using Prompt-Learning.
- To evaluate the generated stories using automated story generation evaluation metrics.
- To compare the developed method against state-of-the-art models/methods.

1.4 Significance of the Study

Story Generation is a field under active research. While short-form story generation has been studied extensively, long-form story generation is relatively under-explored. Although fine-tuning based approaches have been used in previous works, there is a lack of research in generating stories without fine-tuning.

This work tries to fill these gaps by adding to the existing literature, providing benchmarks, and contributing code. This work also explores recent developments in Prompt-based learning and Few-Shot generation.

In terms of application, this work helps story writers write better stories in conjunction with AI. This can help writers get new ideas or get over the writer's block.

1.5 Scope of the Study

The scope of this thesis work is defined as follows:

- The thesis work is to be completed within 17 weeks after submission of research proposal.
- The experimentation will be conducted using open-source software and models and APIs available with free-trial.
- The experimentation will be conducted using available GPU on personal workstation.
- Human evaluation of the generated story is not a part of this thesis work. The evaluation will only focus on automated metrics.

1.6 Structure of the Study

The structure of the study is as follows:

- Chapter 1 – Introduction: This chapter provides introduction and background for this research work.
- Chapter 2 – Literature Review: This chapter mentions the related works in the fields of Automatic story generation and prompt-learning.
- Chapter 3 – Methodology: This chapter gives a detailed walkthrough of the methodology followed during the experimentation stage.
- Chapter 4 – Implementation: This chapter details the different experiments performed for the story generation task.
- Chapter 5 – Results: This chapter discusses the results of the experiments performed in Chapter 4.
- Chapter 6 – Conclusion: This chapter concludes the work done in the thesis and discusses future improvements.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This chapter discusses the previous works done in the fields of Automatic Story Generation and Prompt-Learning.

2.2 Automatic Story Generation

2.2.1 Introduction

Automated story generation involves creating a series of events or actions that can be used to tell a story using a computer program (Li et al., 2013). While systems for generating stories have been developed since the 1960s (Ryan, 2017), they have not achieved the level of creativity and complexity seen in human-generated stories and are therefore considered to be weak artificial intelligence systems (Jain et al., 2017).

To be considered truly creative, a computer system must be able to generate stories that are different from those it has seen before. This requires taking into account a wide range of attributes, such as the setting of the story, the desires and motivations of the characters, and the interactions and conflicts between characters. The large number of potential attributes involved in a story can make it difficult for a computer system to efficiently search for and generate a unique story. The story generation process is further complicated by additional factors such as, the goal of the story, story interestingness and believability of the story.

There has been significant amount of research on computational narratives and different systems for story generation. (Gervás, 2009) examined the way these systems attempted to replicate human creativity. Story generation systems were classified into four categories by (Kybartas and Bidarra, 2017): manual writing by the author, plot creation through automated processes, space generation through automated methods, and story generation that utilizes both automated plot and space creation.

2.2.2 Structural Models

There are two main approaches to generating stories with computer programs: story graphs and schemas. The story graph approach creates a branching graph consisting of all the possible story paths. Then a path is selected to create the story. The story graph's quality determines the quality of the story. (Li et al., 2013) proposed the SCHEHERAZADE system collects human experiences in the form of scripts and uses them to generate stories by using these scripts to create a graph.

The schema-based approach makes use of story slots that are chained together to create stories. These slots are extracted from other stories based on continuation of the event-chain.

While structural models can generate well-structured stories and are easy to implement, they are limited in their ability to generate stories with multiple protagonists, and their generated stories may not be coherent or believable due to a lack of focus on the semantics of the story. They may also suffer from the over-generation problem, producing non-story texts that are accepted as stories.

2.2.3 Planning Based Models

Story grammar theories have been criticized for their limited ability to understand stories (Black and Wilensky, 1979; Black and Bower, 1980), as they concentrate on the structure of a narrative rather than its meaning and are not suitable for tales with conflicting objectives or multiple main characters. (Wilensky, 1983) developed the story points theory, which looks at a story as a causally connected event-chain working towards a specific goal. Plotto: The Master Book of All Plots introduced by (Cook, 2011) contains a collection of plot fragments and instructions to create plots. It is based on the story points theory and has been used as a source for computational narratives. However, the Plotter system proposed by (Eger et al., 2015), which generates plots from Plotto's plot fragments, has limitations in terms of story consistency due to its reliance on the story's current state rather than its past states.

2.2.4 ML Models

In machine learning (ML), a story can be viewed as a sequence of story events, and the probability distribution of these events can be learned from a corpus of stories.

2.2.4.1 Story Abstraction

To improve the efficiency of learning and inference in machine learning (ML), it is useful to create a simplified representation of stories, known as a story abstraction, which focuses on the main events and entities in the story (Martin et al., 2018). There are several approaches to creating a story abstraction, each with its own trade-offs. In (Chambers and Jurafsky, 2008, 2009) and (Jans et al., 2012) stories were presented as a chain of events, and (verb, dependency) pairs were used to associate each event with the story's protagonist. While this representation is able to capture the relationships between verbs and their arguments, it can lead to inconsistent subject-verb-object tuples and is limited to representing the actions of a single protagonist.

In contrast, (Balasubramanian et al., 2012) generated relational triples called Rel-grams schemas, in the form [arg1, relation, arg2]. This representation helps show the interactions between entities, but it may be more sparse than other approaches. (Pichotta and Mooney, 2014) proposed a 4-tuple representation with the form [verb, subject, object, prepositional], which can express interactions between entities and was later improved by the inclusion of prepositions (Pichotta and Mooney, 2016a). (Martin et al., 2018) and (Tambwekar et al., 2018) used a [subject, verb, object, modifier] representation, while (Ammanabrolu et al., 2019) used a 5-tuple representation with the form [subject, verb, preposition, object, modifier].

2.2.4.2 Script Learning and Generation

The task of Script learning and generation uses statistical models to analyze relationships between events in a story and then given a chain of events, predict the next event.

(Chambers and Jurafsky, 2008) identified the chain of events in a story and the interactions between events for a character using coreference relationships between events and Pointwise

Mutual Information (PMI). They then produced a list of probable events by classifying the relationship between two events as mentioned in (Chambers et al., 2007).

(Jans et al., 2012) paired each event with the three events after that in the sequence and then used skip n-grams to calculate the statistics for the entire event chain. This method improved upon the PMI method by decreasing sparsity of data and improving the training procedure. The bigram probabilities ranking technique used in this method evaluated events based on their sequence in the chain, considering the events before and after. This improved upon previous techniques by modeling the event chain in the order of observation.

(Balasubramanian et al., 2012) introduced the Rel-grams system which is a Markov model. But unlike (Jans et al., 2012), that focused on argument co-occurrence, Rel-grams focused on co-occurrence between relationships. As such given the relationships and the one of arguments, it could predict the missing argument.

(Pichotta and Mooney, 2014) proposed using multi-arguments with the event structure to represent the connections between events and entities, and to simulate how different entities interact with one another. This helped the system in creating a continuous chain of events for the entire story rather than fragmented event chains based on individual entities created by verb-dependency pairs. This system demonstrated improved prediction accuracy compared to systems that used verb-dependency pairs.

(Rudinger et al., 2015) approached event prediction as a language modeling task and used a Log-Bilinear model to predict story events. (Pichotta and Mooney, 2016a) used Long Short-Term Memory (LSTM) model to predict nouns or coreference information regarding event arguments and demonstrated better performance than baselines. Furthermore, they made event predictions solely using raw text, without relying on event structures (Pichotta and Mooney, 2016b). Their findings revealed that the dissimilarity between raw text models and structured event models was insignificant. This suggests that event structures may not be essential for predicting events, particularly in an encoder-decoder framework.

(Mostafazadeh et al., 2016a) introduced the Story Cloze Test (SCT) and created the ROCStories Corpora to evaluate capabilities of machine-generated story endings based on story context. (Chaturvedi et al., 2017) presented a model for learning scripts, which relied on three semantic

elements: the order of occurrence of events, the emotional development, and the coherence of the topic. (Mostafazadeh et al., 2016b) trained a model that used the embeddings for story context as features and the two alternate endings as labels to predict the correct story ending. (Wang et al., 2017a) used generative adversarial networks (GANs). Here, the task of the generator was to use the story context to produces a fake sample, and the discriminator's task was to identify the real sample against the fake ones. More recent studies (Li et al., 2018a, 2021; Chen et al., 2019; Ippolito et al., 2020) have shown noticeable improvements on SCT results when training is done using large datasets.

2.2.4.3 Story Completion

The objective of the Story completion task is that, given a story context, the machine needs to complete the story plot (Guan et al., 2019). (Roemmele et al., 2017b) used the Children's Book Test (CBT) dataset as their story corpus and used a RNN model to generate the last sentence, one word at a time. It also compared the generated sentence to the original as an evaluation benchmark. The significance of this study lies in its use of various linguistic metrics to automatically evaluate the generated stories. (Hu et al., 2017) presented a hierarchical LSTM model that uses context and previous subevents to predict future subevents. The model generates the future subevent as a sequence of words and considers the topic of the story during generation. (Li et al., 2018b) trained a Seq2Seq model using adversarial techniques to generate different story endings automatically. They believed that existing Seq2Seq models were only appropriate for generation tasks which had gold-standard target labels. However, no such gold-standard label exists in story ending generation tasks, where the story ending can any arbitrary ending that is considered reasonable. A discriminator is used to label the generated story ending as human-written or machine-generated and the generator is trained to fool the discriminator.

(Guan et al., 2019) trained a neural model where story endings are generated based on coherence of the plot and background knowledge. Story coherence is made up of story events, story attributes, and relationships between different story events. ConceptNet is used as a source to incorporate background knowledge, and this is controlled using a multi-source attention.

(Wang et al., 2020) built a model based on the GPT-2 language model (Radford et al., 2019) that could predict the missing sentence in a story given the preceding and following sentences. (Wang and Wan, 2019) did something similar, except their model could also predict missing sentence at the end of the story.

2.2.4.4 Story Generation

(Jain et al., 2017) developed a story generator by combining two systems, first a Statistical Machine Translation (SMT) technique was used to work on individual phrases within a sentence and then a deep Recurrent Neural Network (RNN) which treated each sentence as a single unit and generated complete stories from them. However, during evaluation, the scores were not very high and the semantic similarity between generated outputs and the original input was also not high.

(Choi et al., 2016) generated stories by using a RNN model to predict next sentence, give previous ones. The approach combined two RNN models – the RNN Encoder-Decoder model was used to convert a sentence into its embedding vector representation and vice versa, and the RNN Story Generator used the encoded vectors to generate new sentences. Although the generated sentence got the grammar and overall content correct, it sometimes used words that did not fit the narrative. (Harrison et al., 2017) followed a similar two-step process as (Choi et al., 2016) where they used Markov Chain Monte Carlo sampling with the RNN model to control the next sentence prediction.

(Tambwekar et al., 2018) used Reinforcement Learning to control the story generation of their RNN model. The RNN model was provided with a start and end state for the story with goal of traversing from the starting state to ending state. It was rewarded based on how far the current event was from the final event.

(Ammanabrolu et al., 2019) used the same the policy gradient deep reinforcement learning approach from (Tambwekar et al., 2018) to generate events. On top of that, to improve the quality and interestingness of generated story text, they trained four event-to-text models – Retrieve & Edit model, Template-based model, Seq2Seq model using Monte Carlo beam

decoding and Seq2Seq model with a finite state machine decoder. They found they ensembling these models together gave better results than using them separately.

Although Recurrent Neural Networks (RNNs) have proven to be effective in many sequence-to-sequence problems, they have fallen short in the generation of stories. This is because they frequently fail to produce coherent stories beyond a few sentences. RNN models have a very short memory and as the generation progresses, the model starts losing connection with the older tokens (Khandelwal et al., 2018). As story generation depends heavily on coherence and connecting a sequence of events, RNNs perform poorly as after a short span, the newly generated events have no connection to previous events.

(Fan et al., 2018) broke down the process of story generation into two phases to enhance the coherence of the generated stories. They first used a CNN-based language model to generate the story premise. Then they used a Seq2Seq model to convert the premise into text. To improve the correlation between premise and the story a fusion mechanism was added to the Seq2Seq model. It also had a self-attention module to consider long-range context.

(Yao et al., 2019) introduced a hierarchical story generator that added plot planning into story generation. They use RAKE (Rose et al., 2010) as a keyword extractor to convert each story in the corpus into a story skeleton. Then they use a Seq2Seq model to generate stories from these skeletons using two methods:

- Static Schema: generates the complete storyline and then converts it into text
- Dynamic Schema: generates the next word and next sentence in the story at every step

The results show that planning the storyline beforehand produces better stories with higher scores across metrics such as coherence, interestingness, fidelity, and overall user preference. Results also show that Static Schema produces more consistent and coherent stories compared to the Dynamic Schema.

(Xu et al., 2018) propose a similar approach as (Yao et al., 2019) where they also use a Seq2Seq model to generate stories from a skeleton of keyphrases. The difference is, they use reinforcement learning approach to identify the key phrases and learn the semantic relationship between sentences in a story. They found that story coherence can be impacted negatively by the length of the generation and how familiar the model is with the input.

(Chen et al., 2021) also take a different approach by generating an outline prior to creating the story. They utilize an existing text summarization tool to extract story plot summaries from the training corpus and pre-train a planning model with natural language summaries. Then they generate stories by passing a story title and outline to a Seq2Seq model. Even though the results are better than previous works, the generated stories still lack coherence.

(Zhai et al., 2019) generated stories from a small corpus by using a hybrid model. First, it uses the story corpus to extract a temporal script graph. Then it samples a story plan through the temporal script graph and uses an agenda generator to plan the story using the sampled plan. Then a neural surface generation unit is used to generate story text according to the story plan.

(Araz, 2020) generated stories from story prompts by using a transformer-based neural network. Although this method generated unique and feasible stories, it did not follow the prompts to the desired extent. The generated text also had grammatical errors and repetitions.

Pre-trained Language Models (PLMs) like GPT2 (Radford et al., 2019) have shown that they can be used to produce text similar to human writing. This led to researchers using these models for story generation tasks. Similar to previous works, (See et al., 2019) introduced two models that were trained to generate stories based on input prompts. GPT2-117, the smallest variant of GPT2, and a pre-trained version of the Fusion model introduced by (Ott et al., 2019). The results showed that the GPT2-117 produced better stories than the Fusion model but it still had a lot of repetition. (Holtzman et al., 2019) also pointed out the ineffectiveness of PLMs for the story generation task, observing that these models produce bland, inconsistent, or repetitive text.

(Guan et al., 2020) claim that insufficient data is the cause of pre-trained models' subpar performance. Unlike other generation tasks, such as summarization, story generation lacks a clear standard to compare the model's output against and this hampers the learning process. To address this, the authors proposed utilizing external knowledge bases to provide additional commonsense knowledge for improved story generation. They also captured the relationship between sentences in a story in terms of causality and time by using multi-task learning. As a result, their model generated more coherent and logical stories compared to standard models.

Adding to the results of (Guan et al., 2020), (Xu et al., 2020) introduced a framework for generating controllable stories which dynamically added commonsense knowledge on top of

language models. During each generation phase, the model predicts a series of keywords based on the story's context, then uses these keywords to retrieve related concepts from a commonsense knowledge database. Then GPT2 model used the context of the story and the most relevant concepts retrieved to generate the next sentence in the story.

(Li et al., 2021) introduced a method for generating open-ended causal relationships using a Transformer model. The method used two separate models for cause and effect which were trained using a corpus of causal relations. The model produced high-quality and varied outputs of causes and effects. A lexical constraint was also added to the model to produce increase the diversity of generated words and phrases.

2.2.5 Story Evaluation

Assessing the quality of automatically generated stories is a difficult challenge in the realm of story generation because of the subjective nature, variety of evaluation standards, and complex nature of story components. Most existing systems depend on human assessment to evaluate the stories produced, but this approach is rigid, slow, subjective, and lacks a standard of comparison. Furthermore, human evaluators may use their own imagination and understanding to fill in missing elements and give higher ratings to inconsistent stories that may not actually deserve it. (Peinado and Gervás, 2006; Mcintyre and Lapata, 2010; PÉrez and Sharples, 2010; Li et al., 2013; Roemmele and Gordon, 2015)

The following sections examine various methods for evaluating the output of automated story generation systems.

(Chambers and Jurafsky, 2008) proposed the narrative cloze test to evaluate text generation. This test involves presenting a series of story events with one event omitted and the system is asked to predict the missing events from a given set of all possible events.

(Mostafazadeh et al., 2016a) modified the narrative cloze test and introduced the Story Cloze Test (SCT). The test provides the model with four sentences of a story and then two choices for the final fifth sentence. The model's performance is judged based on whether it can predict the fifth sentence correctly.

(Mark and Clark, 2016) introduced the Multiple Choice Narrative Cloze (MCNC) test, which provides a system with five events in a random order from which it needs to select the missing event. This allows the system to utilize more extensive information about the context and the options available, resulting in improved comparisons between different story generation systems.

Once the benchmark or standard is established for comparing the various elements of a generated story, various metrics can be applied to assess the performance of the story generator.

- Statistical models: These models predict the events of a story based on various statistical factors. Evaluation metrics for such models include:
  - N-gram overlap: These types of metrics calculate the n-gram similarity between the predicted and ground truth events. It includes evaluation metrics such as: ROUGE, BLEU, CIDEr and METEOR as seen in references (Martin et al., 2018; Xu et al., 2018; Guan et al., 2019; Yao et al., 2019; Ammanabrolu et al., 2020).
  - Perplexity: This metric gauges the model's capability to predict based on prior context, with lower perplexity values indicating higher accuracy in predictions. Works using perplexity include (Fan et al., 2018; Martin et al., 2018; Ammanabrolu et al., 2020). (Ammanabrolu et al., 2019) defined Perplexity as in Equation 2.2.5.1:

$$Perplexity = 2^{-\sum_x p(x) \log_2 p(x)}$$

$$\text{where, } p(x) = \frac{count(x)}{\sum_{y \in Y} count(y)}$$

Equation 2.2.5.1: Perplexity

    Here x represents a token in the text and Y represents the complete vocabulary.

  - Pointwise Mutual Information (PMI): PMI is useful when an event needs to be selected from multiple options. It is based on the frequency of words appearing together and selects the event with the highest average PMI score among its related entities in the story sequence. Introduced by (Chambers and Jurafsky, 2008), the method has been adopted by others, such as (Jans et al., 2012; Rudinger et al., 2015).

- Embeddings models: Embeddings models use word and sentence embeddings to predict the events in a story. Some such metrics are:
    - Cosine similarity metrics such as - Skip Thoughts Cosine Similarity and Embedding Average Cosine Similarity
    - Greedy Matching Score
    - Average Maximum Similarity: A word-level embedding similarity model proposed by (Roemmele et al., 2017b). It calculates the average of highest similarity scores for each word in the ending and selects the ending with highest average score.
    - Deep Structured Semantic model: Used by (Mostafazadeh et al., 2016a) for the SCT.
    - Conditional Generative Adversarial Networks model: This uses a discriminator to select the right story ending (Li et al., 2018b).

There are several ways to evaluate story generation systems, including comparing the generated stories to a reference, analyzing the linguistic properties of the stories, using customized statistical evaluation measures, assessing the interestingness of the stories based on cognitive theories or social media metrics, and measuring the suspense level in the stories. (Roemmele et al., 2017a) used lexical cohesion, style matching, and entity co-reference as linguistic evaluation measures for generated stories. (Purdy et al., 2018) used grammatical correctness, sequence of events, narrative structure, and local contextuality. (Kartal et al., 2014) calculated the believability of a generated story by multiplying the believability of each action in the story, while (León and Gervás, 2010) used 13 variables such as causality, tension, hypotheses, and interest to guide the generation of stories. (Yao et al., 2019) used inter-story and intra-story repetition to measure diversity in generated stories. (Wang et al., 2017b) used upvotes from social media posts as an approximate metric for quality of story. (Sagarkar et al., 2018) evaluated interestingness of story continuations by crowdsourcing the task. (Behrooz et al., 2019) evaluated story interestingness based on the ability of the system to generate unexpected events. (O'Neill and Riedl, 2014) measured suspense in stories by calculating the cost of escape plans at different time-slices and finding the area under the resulting suspense curve.

There are several drawbacks to evaluating generated stories by comparing them to a reference. One issue is that trying to solve story generation tasks as a classification problem may result in systems that are good at classification but do not understand the semantics of a story and may not be creative enough in their generation. Additionally, the creative nature of generating a story means that it may not be interesting if it is overly predictive. Furthermore, there is no single "correct" answer in story generation, which can be at odds with evaluation methods that require systems to choose a specific answer or be penalized for not doing so.

Some recent works have tried to resolve these problems. To deal with the issue of lack of standardized benchmark datasets, (Guan et al., 2021) proposed OpenMEVA, which provides a comprehensive test suite containing metrics to evaluate correlation with human judgments, generalization to different model outputs and datasets, ability to judge story coherence, and robustness to perturbations. (Chhun et al., 2022), present a collection of six human criteria that are grounded in the social sciences research. Additionally, they introduce an annotated dataset called HANNA, which helps in evaluating the correlations between 72 automatic metrics and the human criteria, in a quantitative manner.

2.2.6 Challenges in Automatic Story Generation

Despite the significant progress that has been made in the field of automatic story generation, the field has not seen as much progress as one might expect, and it still faces a few challenges and limitations (Alhussain and Azmi, 2021).

- Dispersion: The lack of a common set of domain knowledge and evaluation criteria makes it difficult to compare the performance of different story generation systems. This can lead to the success of a system being attributed to its domain knowledge rather than its generation capabilities, and to the evaluation of a system being based on personal human opinion rather than objective metrics. Standardizing these elements would allow for a more accurate assessment of the strengths and weaknesses of different models and facilitate the building upon and improvement of previous work.
- Domain knowledge: Story generation systems have traditionally relied on manually created domain models which resulted in closed-domain stories, but this has limited their

ability to be extended to other domains. Recently, advances in data science and machine learning have led to open-domain story generators that use various sources of knowledge, including story corpora, crowdsourced data, commonsense knowledge, and semantic relations corpora. While these sources can be useful for enhancing the knowledge of both rule-based and machine learning based story generation systems, they also present challenges, such as the lack of standardization and the limited use of semantic relations corpora.

- Pre-trained language models (PLMs): PLMs tend to struggle with repetition, lack of long-range coherence, and logical conflicts. They also have difficulty with commonsense inference. Recent developments such as GPT3 and ChatGPT seem to produce far better results and provide a better avenue for research.
- Story interestingness: To create an interesting story, it is essential to consider both its consistency and structure. While some models focus on one of these aspects, there are only a few systems that aim to balance both. Hierarchical models, which generate a storyline first and then generate the story based on that storyline, have the potential to create stories that are interesting, structured, and consistent. These models can also incorporate elements from cognitive science and literature to increase the interestingness of the generated story. However, most existing story generators do not adequately address all these factors.
- Objective evaluation: The existing story generation systems use common metrics used for text generation tasks, such as BLEU, ROUGE, and perplexity. But these are not suitable for open-domain generation as they expect ground truth references for comparison. This does not work very well with the creative nature of story generation. They also do not correlate well with human judgments (Liu et al., 2016).

2.3 Prompt Based Learning

2.3.1 Evolution of Prompting in NLP

Fully supervised learning has long been the fundamental paradigm in Natural Language Processing (NLP), and machine learning in general (Kotsiantis, 2007). Under this paradigm a model is trained specifically for a target task using a set of source-target samples. As small datasets were insufficient for training NLP models, early models relied largely on feature engineering (Lafferty et al., 2001; Guyon et al., 2002; Och and Ney, 2004; Zhang and Nivre, 2011). As neural networks started being applied to NLP problems, feature selection and engineering became part of the model training process instead (Collobert et al., 2011; Bengio et al., 2013). This led to the next paradigm of architecture engineering. In this paradigm, inductive bias was introduced using careful network architecture design (Hochreiter and Schmidhuber, 1997; Bahdanau et al., 2014; Chung et al., 2014; Kalchbrenner et al., 2014; Vaswani et al., 2017).

Towards the end of 2010's, the standard for model training in NLP moved to pre-train and fine-tune paradigm (Openai et al., 2018; Peters et al., 2018; Dong et al., 2019a; Yang et al., 2019; Lewis et al., 2020; Zhang et al., 2020). This paradigm introduced Language Models (LM), models with (almost) fixed architecture, trained to predict next word from a probability distribution. As the unlabeled and unstructured text data that is required to train language models is widely available, they can be trained on very-large corpus of text and learn general-purpose features of the language they are modelling. This led to objective engineering, where pre-trained LMs were fine-tuned for downstream tasks using task-specific objective functions. For example, a text summarization model can be trained by creating a custom loss function predicting most important sentences in a document as shown in (Zhang et al., 2020).

As of 2021, a new approach to natural language processing called "pre-train, prompt, and predict" has emerged. Instead of fine-tuning pre-trained language models for specific tasks, this approach reformulates downstream tasks into those encountered during the original training of the language model, using a textual prompt. For example, a prompt like " I missed the bus today. I felt so" can be used to ask a language model to continue the prompt with an "emotion" word. Similarly, a prompt like "English: You went to school today. German:" can be passed to a

language model to translate the phrase into French. This approach allows a single, unsupervised language model to solve a wide range of tasks by selecting the appropriate prompts. However, finding the most effective prompts for a given task requires careful prompt engineering. (Petroni et al., 2019; Radford et al., 2019; Brown et al., 2020; Gao et al., 2020c; Raffel et al., 2020; Schick and Schütze, 2020a; Sun et al., 2021)

2.3.2 Introduction to Prompting

One of the main challenges of supervised learning is the need for large amounts of labeled data to train a model to perform a specific task. NLP approaches that use prompts aim to address this issue by training a language model (LM) to model the probability of text itself, rather than the probability of a specific output given input. This allows the LM to predict the expected result without the need for a large amount of labeled data for the specific task.

2.3.2.1 Prompt Addition

To generate a prompt, a prompt-function is applied to the input text, resulting in a modified version of the text. This function usually has two-steps: (1) it applies a template, which is a string with placeholders for the input text and a temporary generated answer that will later be mapped to desired output; and (2) it fills the placeholder for the input text with the actual input. For example, in sentiment analysis, the template might be "Overall, [X] was a [Z] movie," and the input text "I love this movie." would be transformed into the prompt "I love this movie. Overall, it was a [Z] movie." These prompts can be either "cloze" prompts, with a placeholder in the middle of the text, or "prefix" prompts, with the input text coming before the placeholder. In some cases, the placeholder may be represented by a numerical id or a continuous vector, rather than a natural language token. The number of placeholders for prompts and answers can be adjusted as needed for different tasks. (Kumar et al., 2016; McCann et al., 2018; Radford et al., 2019; Schick and Schütze, 2020a)

2.3.2.2 Answer Search

The goal is to find the prediction that maximizes the score for the language model. To do this, a set of possible values are defined for the intermediate generated answer. This set can either be the entire language for generative tasks, or a smaller subset of words for classification tasks. Then the placeholder in the prompt is filled with a potential answer from the set, resulting in a filled prompt. If the filled prompt includes the true answer, it is called an "answered" prompt. Lastly, the list of probable answers is searched by estimating the likelihood of the filled prompts using the pre-trained language models. This search can either be an argmax search, which locates the highest-scoring output, or sampling, which generates outputs at random using the probability distribution of the Language Model. (Liu et al., 2021b)

2.3.2.3 Answer Mapping

Here, the goal is to map answer to the output. In some tasks such as machine translation, the answer itself can be the output, but in some other cases, multiple answers may correspond to the same output. (Liu et al., 2021b)

2.3.2.4 Design Considerations for Prompting

This section outlines several design considerations used for developing a prompting method for NLP tasks (Liu et al., 2021b):

- Pre-trained Model Choice: Several pre-trained language models (LMs) are available and can serve as the backbone.
- Prompt Engineering: The choice of prompt has a significant impact on the accuracy and nature of the task that the model performs.
- Answer Engineering: Based on the type of task, it may be necessary to create a set of possible values for the intermediate generated answer and the mapping function between answers & outputs differently.

- Expanding the Paradigm: The sections presented earlier represent a basic framework for prompting methods, but there are many ways to expand upon and modify this paradigm to improve results or adapt it to different tasks.
- Prompt-based Training Strategies: Different strategies can be used to train the prompt, the LM, or both.

2.3.3 Pre-trained Language Models

There are already several comprehensive surveys available on the impact of pre-trained language models (PLMs) in the "pre-train and fine-tune" approach (Qiu et al., 2020; Raffel et al., 2020; Doddapaneni et al., 2021). This section discusses various pre-trained language models from the perspective of prompting methods.

2.3.3.1 Training Objectives

Pre-trained language models are autoregressive models that predict the next token based on the previous tokens (Radford et al., 2019). The use of denoising objectives presents another method that involves introducing a form of noise to the input sentence, followed by an attempt to reconstruct the original sentence from the altered version. There are two common denoising objectives (Lewis et al., 2020):

- Corrupted Text Reconstruction (CTR): Only repairs the damaged portions of the input sentence back to their original state,
- Full Text Reconstruction (FTR): Rebuilds the entire text, whether it was damaged or not.

2.3.3.2 Noising Functions

The kind of noise used in training objectives that rely on reconstruction can have an impact on performance. For example, in an NER task, noise could be applied specifically to entities within a sentence to improve model performance at predicting entities. There are several types of noising functions that can be used (Liu et al., 2021b):

- Masking: This comprises using a unique masking token, such as [MASK], to replace a token or multi-token span. Random or carefully engineered masking can introduce prior information. Used in (Devlin et al., 2019).
- Replacement: In contrast to masking, another token or piece of information is used in place of the token or multi-token span. Used in (Raffel et al., 2020).
- Deletion: Tokens or multi-token spans are removed from the text without the addition of any masking or replacement tokens. This operation is often used in conjunction with full text reconstruction loss. Used in (Lewis et al., 2020).
- Permutation: This involves dividing the text into different span types such as tokens, multi-token spans, or sentences, and rearranging them into a new text. Used in (Liu et al., 2020)

2.3.3.3 Directionality of Representations

The directionality of representation calculation is an important factor to consider when understanding pre-trained LMs. In general, there are two common approaches:

- Left-to-Right: Each word's representation in this case depends on the word itself as well as every word that has come before it in the sentence.
- Bidirectional: Each word's representation in this case depends on every other word in the sentence, including the words to its left and right.

It is also possible to combine these strategies or condition the representations in a random order, but these methods are less commonly used. The directionality of representation can be calculated using attention masking, which is used in the Transformer architecture (Vaswani et al., 2017).

2.3.3.4 Typical Pre-training Methods

Some popular pre-training methods are mentioned below.

- Left-to-Right Language Model: Left-to-Right Language Models are auto-regressive language models that predict the next word in the sentence (Jurafsky and Martin, 2000).

Some examples include GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020) and GPT-Neo (Black et al., 2021). These Language Models are popular for use as the backbone of prompting methods because they are often large and difficult to train, or not available for public use, making it impractical to use them in a pre-train and fine-tune paradigm.

- Masked Language Models (MLMs): MLMs are a type of bidirectional objective function. They aim to predict masked pieces of text based on their surrounding context. MLMs are often used in prompting methods for tasks that require natural language understanding such as text classification or extractive question answering. BERT (Devlin et al., 2019) and ERNIE (Sun et al., 2019; Zhang et al., 2019c) are some examples of MLM based pre-trained models. Moreover, several prompting techniques have combined MLMs with fine-tuning.
- Prefix and Encoder-Decoder Language Models: For conditional text generation tasks, such summarization and machine translation, where the goal is to generate a target text from an input text, Prefix and Encoder-Decoder models are two popular architectures. Both work with a two-step process:
    - Use an encoder to encode the source text into embeddings
    - Then use a left-to-right language model to decode the embeddings into target text in an autoregressive manner.

  In the Prefix Language Model architecture, decoder decodes the output sequence conditioned on an input prefix sequence. With the exception of the encoder's fully connected mask, the encoder and decoder in this case share the identical model parameters. Examples include UniLM 1 & 2 (Dong et al., 2019b; Bao et al., 2020) and ERNIE-M (Ouyang et al., 2020).

  The parameters of the encoder and decoder are not shared in the second architecture, which is known as the Encoder-Decoder Model. It uses different encoder and decoder models for the input and output text. Examples: T5 (Raffel et al., 2020), BART (Lewis et al., 2020), and MASS (Song et al., 2019).

  These models can be used for text generation tasks with or without prompting and have also been applied to tasks such as information extraction, question answering, and text

generation. Prompting methods can expand the applications of these text generation models and allow for unified modeling across different tasks.

2.3.4 Prompt Engineering

Prompt engineering can be defined as the creation of a function to optimize performance on a specific downstream task. This can involve finding the best prompt template for a given task through manual or automated methods, taking into consideration the shape of the prompt.

2.3.4.1 Prompt Shape

The choice of prompt can affect the performance of the pre-trained model on a downstream task. Cloze prompts, which involve filling in blanks in a text, work well with tasks solved using masked language models (Petroni et al., 2019; Cui et al., 2021). On the other hand, prefix prompts, which involve continuing a string, are better suited for tasks involving generation or auto-regressive language models (Lester et al., 2021; Li and Liang, 2021). Full text reconstruction models can be used with either type of prompt. For tasks like text pair classification that have multiple inputs, prompt templates need to add space for multiple inputs.

2.3.4.2 Manual Template Engineering

In Manual Template Engineering the templates are created manually based on human intuition. This method has been used in the creation of the LAMA dataset (Petroni et al., 2019), a dataset for cloze prompts. Moreover, numerous activities including common sense reasoning, machine translation, and question answering have made use of manually created prefix prompts (Brown et al., 2020). Pre-defined templates have also been used in few-shot learning settings for tasks such as text classification and conditional text generation (Schick and Schütze, 2020c; a; b).

2.3.4.3 Automated Template Engineering

There are several issues with the manual creation of prompt templates, including the time and experience required to do so, and the potential for even experienced designers to fail in finding

correct prompts (Jiang et al., 2020a; b). Automated methods for prompt template design have been developed to tackle these issues.

(Liu et al., 2021b) classified automated design methods as follows:

- Discrete Prompts: The prompts are actual text strings.
- Continuous Prompts: Also, known as soft prompts, Continuous Prompts introduce the inductive bias from prompts directly into the embedding vector space of the model rather than using human-interpretable natural language. Continuous prompts help fix two issues: they allow the embeddings of template words to be in any language, rather than just natural language, and they allow the template to have its own set of parameters. This allows for the possibility of prompt-parameters being fine-tuned using task-specific data from the downstream task rather than being completely dependent on the pre-trained language model's parameters.

Additionally, prompt functions can also be classified as Static prompts where the same template is applied to all inputs and Dynamic prompts where each input gets its own custom template (Liu et al., 2021b).

2.3.5 Answer Engineering

The process of seeking for an answer space and a way to get to the desired result inside that space such that the end result leads to an effective predictive model is known as Answer Engineering. To perform answer engineering, it is necessary to consider two dimensions: the shape of the answer and the method used to design the answer (Liu et al., 2021b).

2.3.5.1 Answer Shape

In natural language processing (NLP) activities, the shape of the answer describes the granularity of the answer and can take several different shapes. One option is to use tokens, which can be individual words from the pre-trained language model's vocabulary. Another option is to use multi-token spans often used for cloze prompts. Sentences or documents can also be used as answers, which are common with prefix prompts. The choice of answer shape depends on the

task being performed. Word tokens or text-spans are often used as answers for classification tasks (Yin et al., 2019), relationship extraction tasks (Petroni et al., 2019), and named entity recognition tasks (Cui et al., 2021). For language generating exercises, answers based on phrases or sentences are frequently used. (Radford et al., 2019) and multiple-choice question answering tasks (Khashabi et al., 2020).

2.3.5.2 Answer Shape Design Methods

Answer engineering requires not just a correct answer shape, but also considerations towards designing the proper answer space and then mapping the answers in answer shape to the output space. Different types of Answers Search methods are listed below:

- Manual Answer Search: There are two main strategies for designing the answer space and mapping it to the output space in manual design:
    - Unconstrained spaces: Here the answer is directly mapped to the output space with a one-to-one mapping. The answer share can take different forms: token sequences (Radford et al., 2019), raw tokens (Petroni et al., 2019), or fixed-length token spans (Jiang et al., 2020a).
    - Constrained spaces: These are often used for tasks with a smaller set of labels such as named entity recognition, text classification, or multiple-choice question answering tasks. For example, (Yin et al., 2019) and (Cui et al., 2021) manually curate a list of words or labels to be used as the answer space for text classification and entity recognition tasks, respectively. A language model is used to predict the likelihood of an answer from the given options in multiple-choice question answering tasks (Zweig et al., 2012).
- Discrete Answer Search: There is less research on automatic answer search compared to prompt search in natural language processing (NLP), but some methods have been developed for both discrete and continuous answer spaces.
    - Answer paraphrasing involves expanding an initial answer space by generating paraphrased answers using a method like back-translation (Jiang et al., 2020b).
    - Prune-and-search techniques initially generate a pruned space of possible answers and then search within this reduced pool to determine the final set of answers.

(Schick et al., 2021) and (Schick et al., 2021) identify most frequent tokens from an unlabeled dataset as the initial answer space. (Shin et al., 2020) apply a logistic-regression classifier based on the answer tokens' contextualised embedding representation. (Gao et al., 2020c) select top-k highest probability words form the vocabulary and further prune the answer space using the zero-shot accuracy for training samples.

  - Label decomposition, used in relation extraction tasks by (Chen et al., 2021), involves breaking each relation label down to its constituent words and using those as the answer.

- Continuous Answer Search: (Hambardzumyan et al., 2021) are one of few works that have explored the use of continuous(soft) answer tokens. Continuous answer tokens can be optimized using gradient descent algorithm. Each class label is given a virtual token, and the token embedding for each class is optimized together with the prompt token embeddings. Unlike other approaches, this one learns a new embedding for each label from scratch rather than using the language model's pre-trained embeddings.

2.3.6 Training Strategies for Prompting Methods

Previous sections describe methods that provide a way to obtain appropriate prompts and corresponding answers. The next step is to consider training strategies that involve explicitly training models along with corresponding prompting methods.

2.3.6.1 Training Settings

In the Zero-Shot setting, a language model can use prompts perform a downstream task without being trained. However, there are also prompting methods that fine-tune the language model along with the prompts. These methods fall into two categories based on how many samples are used to train the model:

- Full-data learning: Large number of training examples
- Few-shot learning: Small number of examples

Prompting methods are especially useful with few-shot learning, as due to the smaller number of training examples, the models cannot produce the desired behavior. Prompting can be used to guide the model in the right direction in these cases (Liu et al., 2021b).

2.3.6.2 Parameter Update Methods

There are typically two types of parameters in downstream task learning using prompts:

- Parameters from the pre-trained model
- Parameters from the prompts

Deciding which parameters to update in which scenario is an important design decision that can affect the model's performance. There are five tuning strategies based on if the parameters of language model or the parameters of prompts need to be tuned. The five tuning strategies are:

- Promptless Fine-tuning: In NLP, the pre-train and fine-tune technique has long been the norm. To distinguish it from learning approaches that include prompts, this technique is known as promptless fine-tuning. Examples of pre-trained models that have been fine-tuned in this manner include BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). This method may not perform well on small datasets and lead to catastrophic forgetting, where the language model forgets things, it was predicting correctly before (McCloskey and Cohen, 1989). The advantages of promptless fine-tuning include simplicity and the ability to fit to larger training datasets, but it has the disadvantage of potentially overfitting or the training being unstable on smaller datasets (Dodge et al., 2020).
- Tuning-free Prompting: This technique does not alter the pre-trained language model's parameters; instead, it creates answers based solely on the prompt. This can be paired with the in-context learning technique, which involves supplementing input with answered prompts. Some examples include LAMA (Peters et al., 2019) and GPT-3 (Brown et al., 2020). Efficiency, the capacity to prevent catastrophic forgetting, and applicability in zero-shot conditions are all benefits of tuning-free prompting. However, this method requires heavy prompt engineering. In addition, providing a lot of pre-answered prompts during testing in an in-context learning environment might be time-consuming and might prevent the usage of big training datasets.

- Fixed-LM Prompt Tuning: As opposed to changing the pre-trained language model, fixed-LM prompt tuning simply modifies the prompt parameters—additional parameters added for the task—using the supervision from the downstream training samples. Examples of this include Prefix-Tuning and WARP (Han et al., 2022). This method has the advantage of retaining knowledge in the language model and being suitable for few-shot scenarios, but it does not work with zero-shot scenarios. It can also have limited representation power while using larger datasets and requires prompt engineering. Typically, the prompts produced are not understandable or changeable by humans.
- Fixed-prompt LM Tuning: In the scenario where both the pre-trained model's parameters and additional prompt-related parameters are updated, both the LM and prompts are fine-tuned in concert. This can lead to improvements in accuracy for certain tasks, particularly in few-shot scenarios. An example of this is shown by (Logan Iv et al., 2021), who found that combining answer engineering and partial language model fine-tuning can help reduce the prompt engineering needed. They create a simple template, referred to as the null prompt, which involves directly combining the input and mask without using any template words, resulting in a high level of accuracy. Advantages of this approach include the ability to specify the task more completely through prompt or answer engineering, leading better few-shot learning. However, it still requires some amount of prompt or answer engineering, and the fine-tuned LMs may not be effective on other tasks.
- Prompt+LM Tuning: Using downstream training samples, the parameters of the pre-trained models as well as the prompt-relevant parameters are updated with supervised training. Examples for this setting include PADA (Ben-David et al., 2021) and P-Tuning (Liu et al., 2021c). This method may be appropriate for situations with a large amount of data, but it involves training and storing all the model parameters, which can result in overfitting with smaller datasets.

2.3.7 Prompt-relevant Topics

In this section, the connections between prompt learning and other learning methods are discussed.

2.3.7.1 Few-shot Learning

A Few-shot learning paradigm is one that uses a small number of training samples as opposed to supervised learning that requires large number of training samples. There are many approaches to achieve few-shot learning:

- Embedding learning: Introduced in (Bertinetto et al., 2016), this approach converts each sample into its corresponding vector and embeds them in a lower-dimensional space.
- Model agnostic meta-learning: Introduced in (Finn et al., 2017), this approach learns features that are quickly adaptable to new tasks.
- Memory-based learning: This methodology, first presented in (Kaiser et al., 2017), depicts each sample as a weighted average of tokens from memory.

Prompt augmentation is another way to perform few-shot learning. It involves prepending labeled examples to the current sample in order to extract knowledge from a pre-trained language model without fine-tuning the model. (Bertinetto et al., 2016; Finn et al., 2017; Kaiser et al., 2017; Wang et al., 2020; Kumar and Talukdar, 2021)

2.3.7.2 QA-based Task Formulation

Previous work on QA-based prompting involves re-formulating various NLP tasks as question answering problems, with the goal of unifying multiple tasks within this framework. This approach is similar to prompting methods, which use text-based questions to specify tasks to be performed and aim to effectively use the knowledge in pre-trained language models. However, previous work on QA formulation did not focus extensively on utilizing pre-trained language models. Examples of this type of work include (Kumar et al., 2016; McCann et al., 2018) for unifying multiple NLP tasks, (Li et al., 2021) for information extraction, and (Chai et al., 2020) for text classification.

2.3.7.3 Controlled Generation

Controlled generation is a method that involves adding additional guidance beyond the input text to a generation model in order to control the generated text. This guidance can take the form of styling rules (Sennrich et al., 2016; Fan et al., 2018), domain tags (Chu et al., 2017), length specifications (Kikuchi et al., 2016), keywords (Saito et al., 2020), key phrases or sentences (Grangier and Auli, 2017; Liu and Nikolic, 2021) to decide content, relation triples (Zhu et al., 2020), or other types of information. Some prompting methods can be seen as a form of controlled generation, as they use a prompt to specify the task and add additional information to the input prompt text. Both controlled text generation and prompting methods involve adding additional, learnable parameters to the input text. A variation of the "prompt+LM fine-tuning" technique, controlled generation with a seq2seq pre-trained model allows for fine-tuning of the prompt and pre-trained language model parameters. Examples include GSum (Dou et al., 2020).

Controlled text generation and prompt-based text generation differ significantly in a few important ways.

- To begin with, controlled text generation frequently focuses on regulating the style or content of the generated text while maintaining the original task (Fan et al., 2018; Dou et al., 2021), and this may not require a pre-trained model. On the other hand, the primary purpose of prompt-based text generation is to make efficient use of the pre-trained model and specify the task itself (Li and Liang, 2021).
- Second, unlike input-dependent prompts (Tsimpoukelli et al., 2021) that are frequently employed in controlled text generation, the majority of recent research on prompt learning for text generation uses prompts at the dataset or task level (Li and Liang, 2021).

2.3.7.4 Data Augmentation

Data augmentation involves modifying existing data to increase the amount of data available for training (Fadaee et al., 2017; Ratner et al., 2017). (Le Scao and Rush, 2021) found that adding prompts to a dataset can lead to similar accuracy improvements as adding hundreds of additional data points across different classification tasks. This shows that performing data augmentation and applying prompts on a downstream task are quite similar.

2.3.8 Challenges in Prompting

Despite the promising results of prompt-based learning across various tasks and scenarios, there are still several challenges that need to be addressed. Some of these challenges are discussed below.

2.3.8.1 Prompt Design

Most research on prompt-based learning has focused on text classification or generation tasks, with less attention given to information extraction and text analysis tasks. This is likely because the design of prompts is more challenging for these types of tasks. In the future, it may be necessary to reformulate these tasks as classification or text generation problems or to use answer engineering to express structured outputs in a suitable textual format. Another challenge is how to express structured information in prompts, such as table, relational structures, tree, or graph. (Aghajanyan et al., 2021) employed structured prompts based on HTML to generate web text, but more complex structures have not been extensively investigated. The performance of a model depends on both the template and the answer, therefore there is also the issue of how to concurrently search or learn for the right mix of both. Some research has focused on selecting answers before templates (Gao et al., 2020c; Shin et al., 2020), while (Hambardzumyan et al., 2021) have demonstrated the capacity to learn both simultaneously.

2.3.8.2 Answer Engineering

Answer Engineering faces two main challenges when dealing with classification tasks.

- When there are many classes, it can be difficult to select an appropriate answer space.
- When using multi-token answers, decoding multiple tokens using language models becomes difficult. (Jiang et al., 2020b) proposed a possible solution for this.

While working with text generation problems, the generated responses may differ syntactically even though they are semantically similar, and most research has focused on using a single answer. There is still research ongoing about ways to guide the learning process in text generation tasks when working with multiple references.

2.3.8.3 Selection of Tuning Strategy

There are various approaches for tuning the prompt parameters, language model parameters, or both. But there is a lack of comprehensive understanding of the advantage and disadvantage these methods and more research is needed.

2.3.8.4 Multiple Prompt Learning

Prompt ensembling involves using multiple prompts, which can increase space and time complexity. There is a lack of research on how to extract information from multiple prompts. (Schick and Schütze, 2020b; a, 2021) utilized an ensemble model for annotating a large dataset, although it is unclear how to choose prompts that are effective for ensembling or how to apply ensemble learning for text generation, in general. Prompt composition and decomposition involve breaking down a complex task input into multiple sub-prompts, but it is not clear how to choose between them. Empirically, prompt decomposition is suitable for token or span prediction tasks, while prompt composition is better for span relation prediction tasks. Prompt augmentation methods are constrained by the length of the input, so there is a need to research how to select and order informative samples. Finally, prompt sharing involves applying prompt learning to multiple domains, tasks, or languages, but there is little research in this area. How to create unique prompts for particular jobs or how to control how they interact with one another aren't entirely understood.

2.3.8.5 Selection of Pre-trained Models

For prompt-based learning, there are several pre-trained language models to choose from, but it is unclear how to pick the optimal one. While there are conceptual introductions on how to choose pre-trained models for different NLP tasks, there is a lack of systematic comparisons of different pre-trained language models being used for prompt learning.

2.3.8.6 Transferability of Prompts

It is important to understand the how much each prompt is specific to a particular model and to improve the transferability of prompts. (Perez et al., 2021) found that:

- In the context of fine-tuned few-shot learning, if there is a sizable validation dataset available for selecting prompts, then the prompts usually yield good results across models that are of comparable sizes.
- In the context of true few-shot learning, where the number of training samples is limited, prompts are not as effective in generalizing to models having comparable sizes.

In both cases, when there is a significant difference in model sizes, the ability of prompts to be transferred is inadequate.

2.3.8.7 Combination of Different Paradigms

At least in part, pre-existing pre-trained models (like BERT) that were created using the pre-train and fine-tune method are responsible for prompt-based learning's effectiveness. A lot of uncertainty still exists over the applicability of pre-training strategies that are successful for the pre-train and fine-tune approach to prompt-based learning, as well as whether new pre-training approaches are required to increase the accuracy and usability of prompt-based learning techniques. More research needs to be done in this space.

2.3.8.8 Calibration of Prompting Methods

The capacity of a model to produce precise probability for predictions is known as calibration. It is important to keep in mind that the probability distribution of a pre-trained language model's generation probability might not be properly calibrated, when attempting to predict an answer. (Zhao et al., 2021) identified three potential traps that may cause pre-trained language models to exhibit bias towards specific responses when answered prompts are provided - Recency bias, Majority label bias and Common token bias.

2.4 Summary

There has been a lot of work done in the field of Automatic Story Generation. But most of the works use the previous generation GPT2 as the pre-trained language model for story generation. There are very few works that utilize the power of the latest generation language models such as GPT3 or its alternatives.

Even though Prompt-Learning is a relatively new paradigm, there has been significant amount of research done in this field as well. Prompt-learning has been used for various NLP applications including text generation. But there is lack of works that use prompt-learning for narrative story generation tasks.

This work tries to fill the above gaps by using prompt-learning for story generation and contributing benchmark scores with the latest generation language models.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Introduction

This work focuses on the story generation and prompt learning capabilities of Pre-trained Language Models. The sections below discuss some of the concepts and algorithms required for the thesis work.

3.2 Algorithms & Techniques

3.2.1 Pre-Trained Language Models

Large Language Models (LLM) or Pre-trained Language Models (PLM) are language models trained on Causal Language Modeling (CLM) objective, where CLM is the task of predicting next word given a sequence of words. These models are used as base model in a Transfer Learning scenario. Recently, with improved capabilities of LLMs along with frameworks such Prompt-Learning these models have started to be used for Few-Shot & Zero-Shot Learning.

Below is a discussion of some of the most recent State-of-the-Art Pre-trained Language Models for text generating task.

3.2.1.1 GPT-2

Introduced in the (Radford et al., 2019), GPT-2 is the second generation of Generative Pretrained Transformer. The model was pretrained on a sizable corpus of English data and is a transformer-based model. The model excels at generating texts from a prompt, which is the task it was pretrained for. This model was developed by researchers at OpenAI using a dataset called WebText (not public). WebText consists of the text content from 45 million outbound links posted to the social media platform Reddit. The data was pre-filtered for sexually explicit or offensive content. The model used in this thesis is a relatively small model with 124 million parameters. There are larger versions of this model available – gpt2-medium (355 million), gpt2-large (774 million) and gpt2-xl (1.5 billion) parameters.

3.2.1.2 GPT-3

Introduced in the (Brown et al., 2020), it is the next generation of GPT-2 model with the largest version of the model having 175 billion parameters. It was trained using ~45 TB of text data collected from internet sources:

- CommonCrawl: Contains petabytes of data collected by crawling the internet webpages for over 8 years (Commoncrawl.org, 2023).
- WebText2: A cleaned version of original WebText used for GPT-2 along with data from new links posted since GPT-2.
- Books1 & Books2: Books corpora from the internet.
- Wikipedia: English language pages from Wikipedia.

The GPT-3 model is trained in the following sizes:

Model Name	n_{params}
GPT-3 Small	125M
GPT-3 Medium	350M
GPT-3 Large	760M
GPT-3 XL	1.3B
GPT-3 2.7B	2.7B
GPT-3 6.7B	6.7B
GPT-3 13B	13.0B
GPT-3 175B or "GPT-3"	175.0B

Figure 3.2.1.1 GPT-3 Model Sizes

GPT-3 is a decoder only model. In other words, during training only the decoder part of the transformer architecture is used to train the model.

OpenAI has not made the GPT-3 models available for download as open-source pre-trained models. Instead, the company provides these as APIs that can be used for a fee. More on this is discussed in Section 3.3.

3.2.1.3 GPT-3.5

Due to the nature of training for Causal Language Models, the models predict next word based on the previous words, regardless of what the user instruction maybe. But for Prompt-Learning, particularly in a Zero-Shot scenario, the results depend a lot on the initial instruction provided by the user. OpenAI realized that models like GPT-3 that predict only based on CLM objective were not aligned with their users. So, they further fine-tuned GPT-3 using Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017), which resulted in the InstructGPT model (Ouyang et al., 2022). OpenAI claims that InstructGPT models are much better at following instructions that previous models. They are also less likely to generate factually incorrect text as facts.

These InstructGPT models were further fine-tuned to create GPT-3.5 models (OpenAI, 2023b). At the time of writing this thesis, OpenAI's best models are from GPT-3.5.

3.2.1.4 OPT

Open Pre-trained Transformer (OPT) Language Models were introduced in (Zhang et al., 2022) by Facebook/Meta. It is trained using the same basic architecture as GPT-3 with CLM as its objective. For evaluation as well, it uses the same prompts and experimental setup. The OPT model is trained on a dataset of size 800GB containing data from following sources:

- BookCorpus (Zhu et al., 2015): Consists of more than 10K unpublished books.
- CC-Stories(Trinh et al., 2018): A subset of the CommonCrawl data processed to match the story-like style of Winograd schemas
- The Pile (Gao et al., 2020b): A 825GB text corpus collated by EleutherAI. Out of this Pile-CC, OpenWebText2, USPTO, Project Gutenberg, OpenSubtitles, Wikipedia, DM Mathematics and HackerNews datasets were selected for OPT.
- Pushshift.io Reddit dataset: A dataset used by (Baumgartner et al., 2020)
- CCNewsV2: Contains English portion of CommonCrawl News dataset. It was used in RoBERTa paper (Liu et al., 2019).

The OPT-175B model was trained for roughly ~33 days of continuously using 992 80GB A100 GPUs. Like GPT-3, OPT is also trained in different sizes ranging from 125M to 175B parameters. But unlike GPT-3, OPT models are available for download as pre-trained models. However, Meta has limited the usage of the model only for research purposes.

3.2.1.5 OPT-IML

Just like OpenAI's InstructGPT, Facebook improved upon OPT models with Instruction Tuning, with a process called Instruction Meta-Learning (IML) (Iyer et al., 2022).OPT-IML models are trained on a large benchmark of 2000 NLP tasks called OPT-IML Bench. This benchmark created for Instruction Meta-Learning (IML) is created from 8 existing benchmarks such as Super-NaturalInstructions (Wang et al., 2022), FLAN (Wei et al., 2021b), PromptSource (Bach et al., 2022), etc. The 30B model was fine-tuned on 64 40GB A100 GPUs.

3.2.1.6 BLOOM

Training Large Language Models not only requires extremely large datasets, but also very high compute resources. There are only a very few organizations that can afford the cost needed to train the language models from scratch. This makes it so that only the richest organizations have access to these models, and it is difficult for other researchers as well engineers use or contribute to these models and related research.

Another problem is that most of the textual information available on the internet is written in English and a few other high-resource languages. This leads to the language models having high performance on English, while much worse performance on low-resource languages. And since training these models is costly, fine-tuning them for low-resource languages is not easy.

Recognizing these problems, the open-source community and researchers came together to form the BigScience initiative. It's an open collaboration bootstrapped by HuggingFace, GENCI and IDRIS. This initiative gathered more than 1000 academic, industrial, and independent researchers from many fields of research including AI, NLP, social sciences, legal, ethics and

public policy. These researches developed the BigScience Large Open-science Open-access Multilingual Language Model (BLOOM) (BigScience et al., 2022).

BLOOM is an autoregressive Large Language Model, trained to continue text from a prompt on vast amounts of text data using industrial-scale computational resources. It can generate text in 46 languages and 13 programming languages, on which it was trained amounting to 1.6 TBs of text data. It has a decoder-only architecture and is modified from Megatron-LM GPT2 (Shoeybi et al., 2019). It was trained using 384 A100 80GB GPUs. The largest version of the model has 176 billion parameters, although it is available in smaller sizes as well.

3.2.1.7 BLOOMZ

BLOOMZ (Muennighoff et al., 2022) models are the BLOOM versions of instruction-tuned models. They can follow human instructions in dozens of languages zero-shot. They were trained by fine-tuning BLOOM language models on a cross-lingual task mixture called the xP3 dataset (Muennighoff et al., 2022). xP3 (Crosslingual Public Pool of Prompts) is a collection of prompts & datasets across 46 of languages & 16 NLP tasks.

3.2.1.8 LaMDA & PaLM

Like other organizations, Google has trained Large language Models, such as:

- Language Model for Dialogue Applications (LaMDA) (Thoppilan et al., 2022)– A model trained using only dialogues and open-ended conversations.
- Pathways Language Model (PaLM) (Chowdhery et al., 2022) – A 540-billion parameter, dense decoder-only Transformer model trained.

But unlike the other language models mentioned above, Google does not provide these models for usage as either open-source or commercial APIs.

The related works for this section have been discussed in Section 2.2.3.

3.2.2 n-Shot Learning

Although the idea of Zero-Shot and Few-Shot learning existed earlier, it was only very recently, with the advent of very large Pre-trained Language Models and Prompt Learning that these techniques have truly started to shine. The motivation for these ideas comes from the requirement of large training datasets for Supervised Learning in NLP. Although with Transfer Learning, the size of training datasets required has gone down, a few hundred labelled records per class are still required to achieve high accuracy for domain specific usecases. Creating these labelled datasets is a time-taking and costly endeavor. This is where n-shot learning tries to help.

The idea behind n-Shot learning is it tries to predict the correct label with very few samples without having to fine-tune the base model to a particular task or dataset. In other words, the model weights are not updated, and the reference samples are all passed during inference time.

n-Shot learning can be broadly classified into the following categories:

- Few-Shot Learning: A few samples (3-10) of source to target pairs are used to guide the prediction towards the required labels.
- One-Shot Learning: Same as Few-Shot Learning but only one reference is provided to the model.
- Zero-Shot Learning: This is the most extreme version of Few-Shot Learning where the model is expected to predict correctly without Zero reference samples.

Some related works for this topic have been discussed in Section 2.2.9.1.

3.3 Methodology

3.3.1 End-to-End Pipeline

The end-to-end pipeline for the methodology is shown in Figure 3.3.1.1.

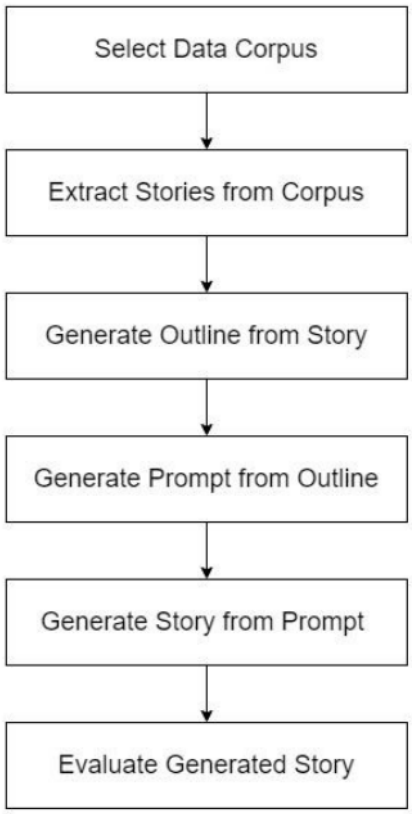

Figure 3.3.1.1 E2E Pipeline

The End-to-End pipeline has the following steps:

- The data corpus is selected from one of ROCStories and WritingPrompts
- The stories are extracted from the corpus
- The Outlines are extracted from each Story

- The Outline is used to Create Prompts
- After generating Prompts, the generated Prompts are used to generate Stories by Prompting the Language Models
- After the stories are generated, they are evaluated using Automated evaluation metrics

3.3.2 Data Description

This work makes use of two standard story generation datasets:

- **ROCStories**: Introduced by (Mostafazadeh et al., 2016a), this dataset contains ~98K 5-sentence long stories along with story titles. This dataset is widely used for short-form story generation tasks.
- **WritingPrompts**: Introduced by (Fan et al., 2018), this dataset contains ~300K human-written stories along with the starting prompt used to write the story. These stories were collected from the Reddit, an online social media forum. These stories are long-form multi-paragraph stories, and hence useful for more complex task of long-form story generation.

3.3.3 Data Preparation

3.3.3.1 ROCStories

The ROCStories dataset contains 5-sentence stories with very simple plotlines. These can be re-purposed to create story outlines as shown in Figure 3.3.3.1. But as there are no target stories for these outlines, multiple samples of outline-story pairs cannot be created. Thus, the outlines from ROCStories can only be used in Zero-Shot manner.

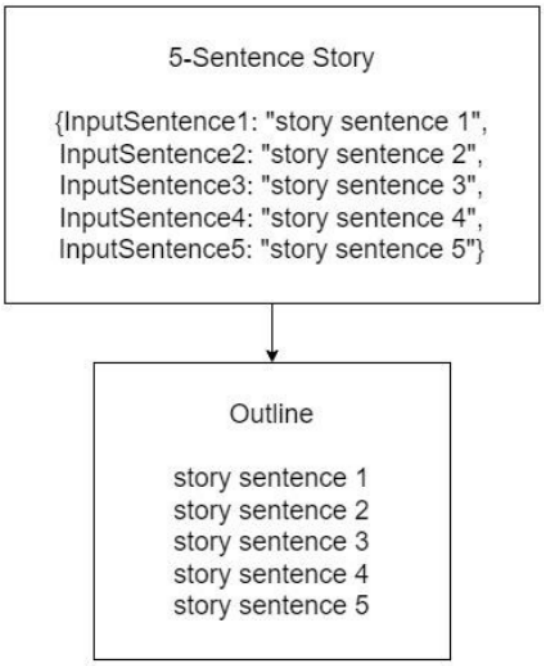

Figure 3.3.3.1 ROCStories Outline Generation

3.3.3.2 WritingPrompts

The method requires sample source-target pairs of outline to story. While story text can be derived from the WritingPrompts dataset, there is no dataset of outlines readily available. Hence, the outlines need to be extracted from the story. These outline-story pairs are then sampled during the Zero-shot/Few-shot inference.

The outline is extracted as a short extractive summary of the story. A new story is then generated from the outline summary as shown in Figure 3.3.3.2.

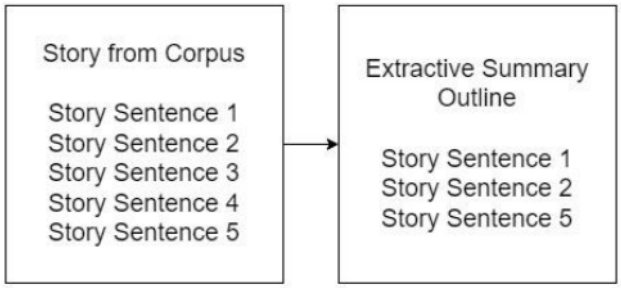

Figure 3.3.3.2 WritingPrompts Outline Generation

3.3.4 Implementation

3.3.4.1 Generate Prompt from Outline

In this step, the story outlines generated in the previous step are converted into Prompts to be used for Prompt-Learning.

For Zero-Shot Prompt, an instruction is added to the beginning of the outline. The instruction can be something like: “Generate a story using the following outline:”.

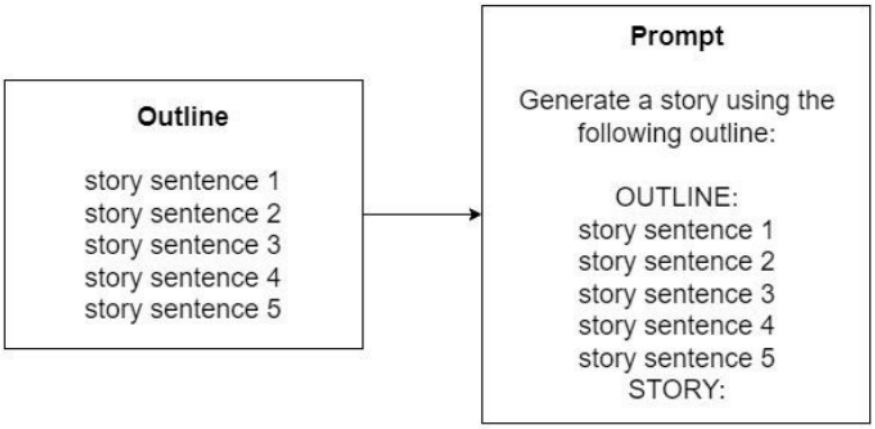

Figure 3.3.4.1 Zero-Shot Prompt

For Few-Shot Prompt, along with the instruction at the beginning, the prompt also contains a few samples of outline-to-story pairs. The final sample in the prompt only has the outline and the model is expected to generate the story for the final outline.

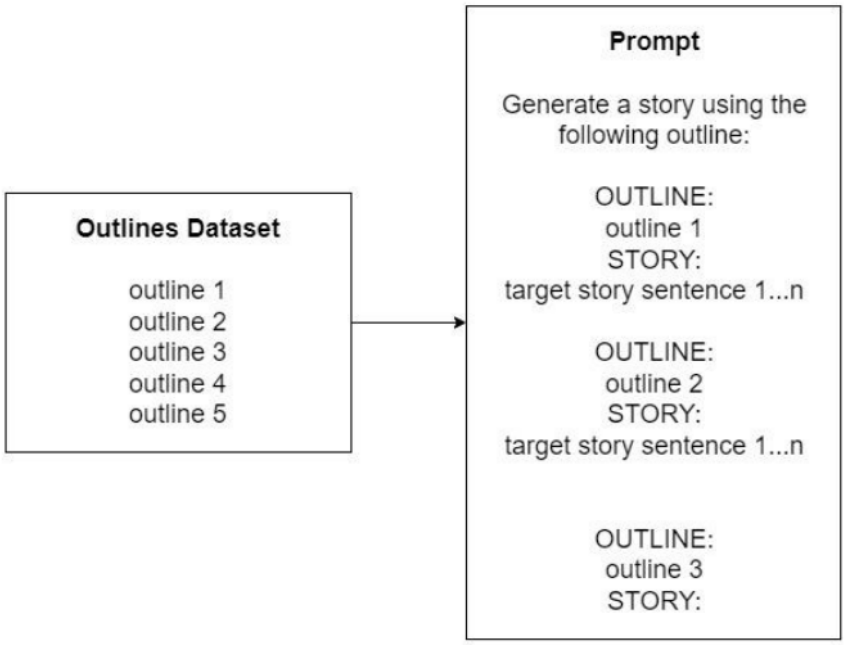

Figure 3.3.4.2 Few-Shot Prompt

3.3.4.2 Generate Story from Prompt

The prompt generated in the previous step is passed to the model as input for inference. The model returns the generated story paragraph corresponding to the query outline as prediction.

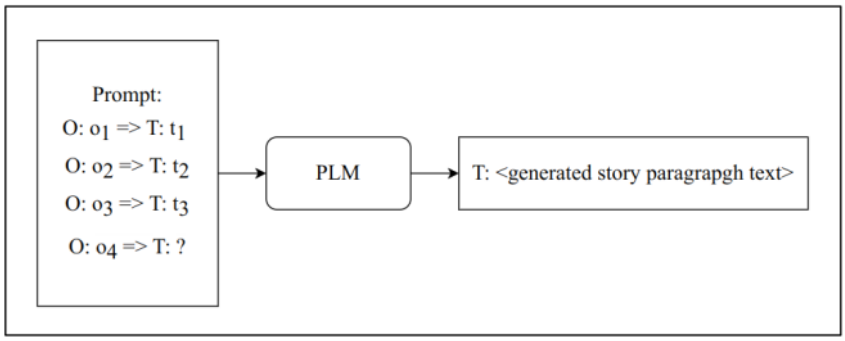

Figure 3.3.4.3 Prompting for Story Generation

3.3.5 Evaluation

The generated stories are evaluated using multiple metrics. This work only focuses on evaluation using Automatic Metrics. Human evaluation of the generated stories is not within the scope of this work.

The metrics used for evaluation are:

- Perplexity: Perplexity (PPL) is one of the most common metrics used for evaluating language models. It is defined as the exponentiated average negative log-likelihood of a sequence, calculated with exponent base 'e'. Perplexity calculates how likely it is for a given model to generate a given input text sequence. It serves as a metric for evaluating the model's ability to learn the distribution of the text data that it was trained on. The range of this metric is [0, inf) with a lower score being better. Perplexity is calculated as follows:

  $$perplexity = e^{**}(sum(losses) / num_tokenized_tokens)$$

  One limitation of Perplexity is that the output value relies heavily on the text that the model was trained on. As such, perplexity scores are not comparable across different models or datasets.
- BLEU: Introduced in (Papineni et al., 2002), Bilingual Evaluation Understudy (BLEU) is a metric generally used to evaluate the quality of machine translated text. Quality is measured using the n-gram overlap between generated text (machine translation) and ground truth (human translation). The central idea behind BLEU is that the closer a machine translation is to a professional human translation, the better it is. BLEU shows a high correlation with human judgement of quality and is a popular metric. Scores are calculated for individual sentences by comparing with ground truth and then these scores are averaged over the entire corpus to get a score for overall quality. BLEU scores range between 0 to 1. A higher score implies that the predicted and reference texts are similar.

  Some limitations of BLEU are:

  - It compares only tokens and not meaning of the sentences. So, semantically different sentences might have a high score if they have many of the same words.
  - Given how the score is calculated, shorter predictions achieve higher scores than longer ones.

- BLEU scores are not comparable across different parameters, datasets, or languages.
- Intelligibility or grammar are not considered towards the score calculation.

- ROUGE: Introduced in (Lin, 2004), Recall-Oriented Understudy for Gisting Evaluation (ROUGE) is a set of metrics generally used to evaluate summarization and translation tasks. The metrics compare a generated (predicted) summary or translation against a set of references (human-labelled).
- BERTScore (Zhang et al., 2019a): BERTScore uses pre-trained contextual embeddings from BERT and calculates the cosine similarity between words in candidate and reference sentences. BERTScore has shown correlation with human judgment on sentence-level and system-level evaluation. It produces Precision, Recall & F1 scores, each of which ranges from 0 to 1.

These metrics will be compared across different factors.

- Based on Pre-trained Language Model used:
  - GPT2
  - GPT3
  - BLOOM
- Based on the Inference type:
  - Zero-shot Generation
  - One-shot Generation
  - Few-shot Generation

3.4 Tools

3.4.1 Software

The software used for this project are listed below:

- Coding Language: Python 3.9
- Salient Python Packages used for this project:
  - Jupyter-Lab (v3.5.2): Web-based interactive development environment (IDE) for notebooks, code, and data.

    - Pandas (v1.5.2): Python package for Data manipulation, especially tabular data.
    - Numpy (v1.24.1): Python package for working with arrays and matrices.
    - Transformers (v4.25.1): Machine-Learning library for PyTorch, TensorFlow and JAX. Transformers provides APIs and tools to easily download and train state-of-the-art pretrained models from Huggingface-Hub.
    - Openai (v0.26.1): A python package provided by OpenAI to access the OpenAI model APIs.
    - Evaluate (v0.4.0): A python library, by Huggingface, for easily evaluating machine learning models and datasets.
- Word Editor: Microsoft Word
- Browser: Microsoft Edge

3.4.2 Hardware

The hardware used for this project are listed below:

- Laptop/Workstation with following configuration
    - RAM: 32 GB
    - GPU: Nvidia RTX3000 - 6 GB

3.5 Summary

This chapter discussed the Algorithms and Techniques used for the thesis. It also discussed in detail, the Methodology of the implantation in Chapter-4 and hardware and software tools used for the implementation.

CHAPTER 4: IMPLEMENTATION

4.1 Introduction

This chapter discusses the details of the code-level implementation. The sections in the chapter are as follows:

- Data Preparation: This section talks about the dataset selected and the discusses the various pre-processing steps applied to the dataset.
- Story Generation: This section details the different experiments performed for generating stories.

4.2 Data Preparation

As discussed in section 4.3.3, there were issues while using the WritingPrompts dataset. Thus, the ROCStories dataset was selected for this thesis. Figure 4.2.1 shows the steps performed for data preparation.

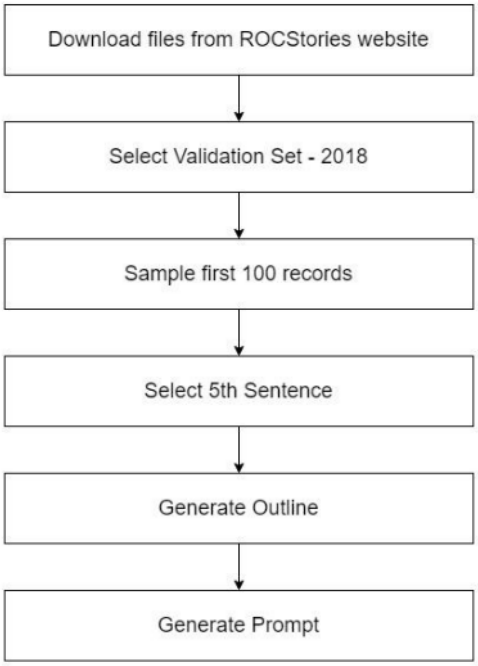

Figure 4.2.1: Data Preparation Steps

The steps performed are described below:

1. Download files from ROCStories website: The csv files for the dataset can be downloaded from the ROCStories website (cs.rochester.edu, 2022) after filling up a form. The files available are as follows:
    a. Spring 2016 dataset – validation& test set
    b. Winter 2017 dataset – validation & test set
    c. Winter 2018 dataset – validation & test set
2. Select Validation Set – 2018: From the above versions, the Winter 2018 dataset was selected as it is the latest and is also the one the authors recommend for use. The validation set was chosen because it contains all 5 sentences of the story, while the test set contains only the first 4. This is read into a pandas dataframe.
3. Sample first 100 records: The story generation process was run only on the first 100 stories in the winter-2018-val set. This was done primarily to limit the cost for GPU utilization and OpenAI API usage.
4. Select 5th Sentence: Since the dataset was created for Story Cloze test, the 5th sentence is available as 1 of 2 choices - RandomFifthSentenceQuiz1 & RandomFifthSentenceQuiz2. The AnswerRightEnding column provides the information for which is the correct 5th sentence. Using this, the correct 5th sentence is added to a new column - InputSentence5.
5. Generate Outline: The 5 sentences are joined together with newline (\n) character to create the story outline and added to a new column – outline. A sample of the outline is shown in Figure 4.2.2.

```
Jim found an old disposable camera in the bottom of his junk drawer.
He began snapping away at everything around him.
The counter clicked down to one final photo.
The gravity of the situation began to dawn on Jim.
Jim took time to decide what he would take a picture of.
```

Figure 4.2.2: Sample Outline

6. Generate Prompt: Instructions and tags are added to the generated outline to create a prompt that will be passed to the language model. A sample of the prompt is shown in Figure 4.2.3.

```
Generate a multi-paragaph story using the following outline:
OUTLINE:
Jim found an old disposable camera in the bottom of his junk drawer.
He began snapping away at everything around him.
The counter clicked down to one final photo.
The gravity of the situation began to dawn on Jim.
Jim took time to decide what he would take a picture of.
STORY:
```

Figure 4.2.3: Sample Prompt

4.3 Story Generation

4.3.1 Steps for Story Generation

Figure 4.3.1.1 shows the steps performed for story generation.

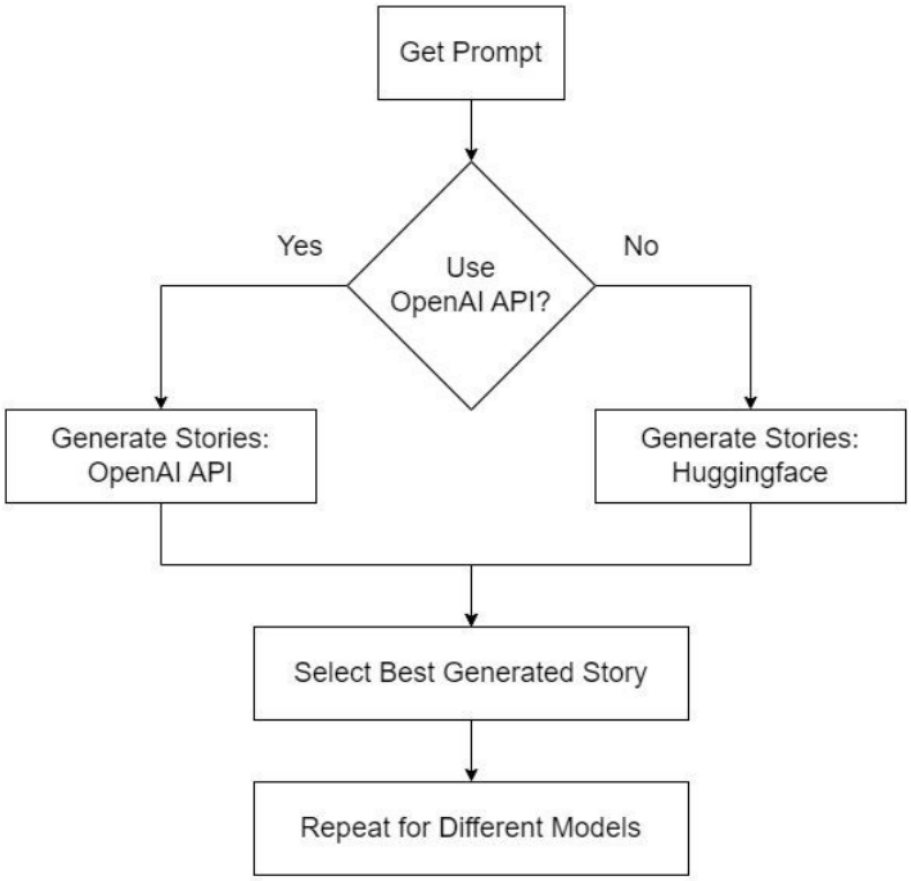

Figure 4.3.1.1: Story Generation Steps

The steps performed are discussed in detail below:

1. Get Prompt: The prompt is read from the output of Data Preparation.
2. Use OpenAI API: There are two methods in which the story generation is done.

a. Using OpenAI's Python Library for GPT3 models.
b. Using Hugginface Python Library for open-source models.

3. Generate Stories:
    a. OpenAI API: OpenAI provides a python library that can be used to access different versions of pre-trained GPT3 models. This is a paid service, but OpenAI provides free credits at the start of the account. The project was completed within the limits of these free credits. The following parameters were passed to the library to generate the text:
        i. model: The model ID to specify the version of GPT3 model to be used. Can be one of –text-ada-001, text-babbage-001, text-curie-001, text-davinci-003. All of these variants were queried to generate different versions of stories based on the same outline.
        ii. prompt: The prompt used to generate completions. The output of the Data Preparation stage is passed here.
        iii. max_tokens: This parameter determines the maximum number of tokens to be generated. This number includes the prompts as well. Most models have an upper limit for this at 2048 tokens. Only text-davinci-003 model has an upper limit of 4096 tokens. For this work, max_tokens is set to 500, which is good enough to generate short multi-paragraph stories.
        iv. temperature: This parameter determines the sampling temperature to use. It's a value between 0-1. Higher values (0.9) are useful for more creative applications such as story generation, while lower values (0) are useful for well-defined answers. For this work, temperature is set at 0.9.
        v. n: This parameter determines how many stories will be generated for each prompt. For this work, n is set at 3 for all model except text-davinci-003, where n is set to 1. This was done to keep cost in check and also because the generation quality of text-davinci-003 is very high and it doesn't need a best_of_n sampling.
        vi. presence_penalty: It's a value between -2.0 to +2.0. The higher the value, the more likely the model is to talk about new topics. For this work,

presence_penalty is set at 0.5. The generated story should talk about some new topics while not straying too far from the provided story outline.

vii. frequency_penalty: It's a value between -2.0 to +2.0. Higher positive values decrease the likelihood of the model generating same lines verbatim. For this work, presence_penalty is set at 0.8. The model is expected to generate a story based on the outline while not repeating the outline.

b. Huggingface: Huggingface also provides a python library that can be used to access different versions of open-source pre-trained GPT3-like models hosted on the Huggingface-hub. The following parameters were passed to the library to generate the text:

i. inputs: The prompt used to generate completions as tokenized ids. The output of the Data Preparation stage is passed to the tokenizer and the output of the tokenizer is passed to inputs parameter.

ii. do_sample: If set to False, model uses greedy sampling to generate text. This leads to less variation in the generation. For this work, do_sample is set to True.

iii. min_length: This parameter determines the minimum generation length. For this work, min_length is set to 50.

iv. max_new_tokens: This parameter determines the maximum generation length ignoring the prompt length. For this work, max_new_tokensis set to 500.

v. top_k: This parameter determines the top k number of tokens with highest prediction probability. The generated token is then sampled from this filtered distribution. For this work, top_k is set to 50.

vi. top_p: This parameter determines the top p percentile of tokens to be kept. The generated token is then sampled from this filtered distribution where the probabilities add up to p. For this work, top_p is set to 0.90.

vii. num_return_sequences: This parameter determines the number of independent responses to be generated. For this work, num_return_sequencesis set to 3.

4. Select Best Generated Story: After the story generation is completed, the longest story is selected as the best story. The longer stories were shown to be qualitatively better than the shorter ones. The model also tends to simply repeat the outline as the generated output which shows as the shortest generated choice.
5. Repeat for Different Models: Steps 1-4 were repeated for different generative models as listed below:
    a. OpenAI GPT3 - text-ada-001
    b. OpenAI GPT3 - text-babbage-001
    c. OpenAI GPT3 - text-curie-001
    d. OpenAI GPT3 - text-davinci-003
    e. Huggingface Hub - bigscience/bloomz-1b7
    f. Huggingface Hub - gpt2

4.3.2 Zero-Shot Story Generation

The steps in Section 4.3.1 were performed in a zero-shot manner. The zero-shot story generation was repeated for all the model mentioned in Step-5 of Section 4.3.1. The results for the story evaluation for all these models is discussed in Chapter-5.

Zero-shot generation was also performed WritingPrompts dataset. For this dataset, story outline was generated from the stories present in the dataset. While this made it possible to use the data for both Zero-shot and n-shot purposes, the quality of the generated outlines was not very good. This caused the stories generated from these outlines to be poor as well. As the quality of generation was very poor compared to the stories generated using ROCStories dataset, these results were not included in the evaluation section in Chapter-5.

4.3.3 One-Shot & Few-Shot Story Generation

Generating stories in a One-Shot and Few-Shot manner led to certain challenges as discussed below.

1. For ROCStories dataset, it was not possible to use it for a One-Shot and Few-Shot generation. The 5-sentence stories were used as outline to generate expanded stories from them. There were no target stories in the dataset to use as generation target for One-Shot and Few-Shot generation.
2. The One-Shot and Few-Shot experiments were also performed with WritingPrompts dataset. Here, since the stories are longer than in ROCStories, it was possible to generate outlines from the stories and use the actual stories as generation target in One-Shot and Few-Shot prompts. This time the problems occurred during the actual generation of the stories. The addition of stories into the prompts made the size of the prompts very big. As such, during prediction, the prompts themselves took up most of the memory in the GPU, leaving little to no memory for the generated tokens. This led to CUDA out-of-memory errors while generating stories. This was especially a problem with Few-Shot prediction as the Few-Shot prompts had multiple stories embedded in them making them too large.

In the end, as the experiments for One-Shot and Few-Shot generation did not lead to any results due to GPU memory issues, One-Shot and Few-Shot experiments were not included in the evaluation section in Chapter-5.

4.4 Summary

This chapter discussed the steps taken to preprocess the data for prompt learning and how the pre-processed data was used to generate stories in a Zero-shot manner.

It also mentions the challenges faced during One-Shot and Few-Shot generation and why their results are not included in the evaluation section in Chapter-5.

CHAPTER 5: RESULTS & EVALUATION

5.1 Introduction

This chapter discusses the results of the experiments performed in Chapter-4. The results are evaluated in both a quantitative and qualitative manner:

- Quantitative Evaluation: In 5.2, the generated stories are scored using multiple automated text generation metrics. The scores are generated for stories generated from different models.
- Qualitative Evaluation: In Section 5.3, samples of generated stories from different models are shown so the generated stories can be compared qualitatively.

5.2 Quantitative Evaluation

Table 5.2.1 shows the scores for different metrics for stories generated by different models.

Table 5.2.1: Evaluation Scores

	gpt2	**bloomz-1b7**	**ada**	**babbage**	**curie**	**davinci**
bleu	0.01	0.22	0.19	0.21	0.16	0.07
perplexity	18.88	91.76	70.76	27.74	25.82	32.66
rouge1	0.15	0.45	0.46	0.54	0.48	0.29
rouge2	0.03	0.28	0.31	0.35	0.27	0.14
rougeL	0.1	0.37	0.38	0.44	0.38	0.23
rougeLsum	0.14	0.43	0.45	0.5	0.43	0.27
bertscore_mean_precision	0.7	0.81	0.8	0.84	0.82	0.75
bertscore_mean_recall	0.77	0.84	0.88	0.9	0.9	0.9
bertscore_mean_f1	0.73	0.82	0.84	0.87	0.86	0.82

Some key points regarding the calculation of these scores are discussed below:

- All scores except perplexity were calculated by comparing the generated story against the outline used to generate the story.
- Perplexity was generated by using distilgpt2 model as the base model.
- The "evaluate" python library was used to calculate the scores.
- The scores were rounded up to 2 significant digits.
- As BERTScore generated scores for each instance, the scores were averaged to get mean values for the precision, recall and f1.

Figure 5.2.1 shows the same information in the form of a plot. As can be seen in the plots, there is a clear pattern as we move from gpt2 towards davinci model. The language models shown on the x-axis are in an increasing order or size and complexity.

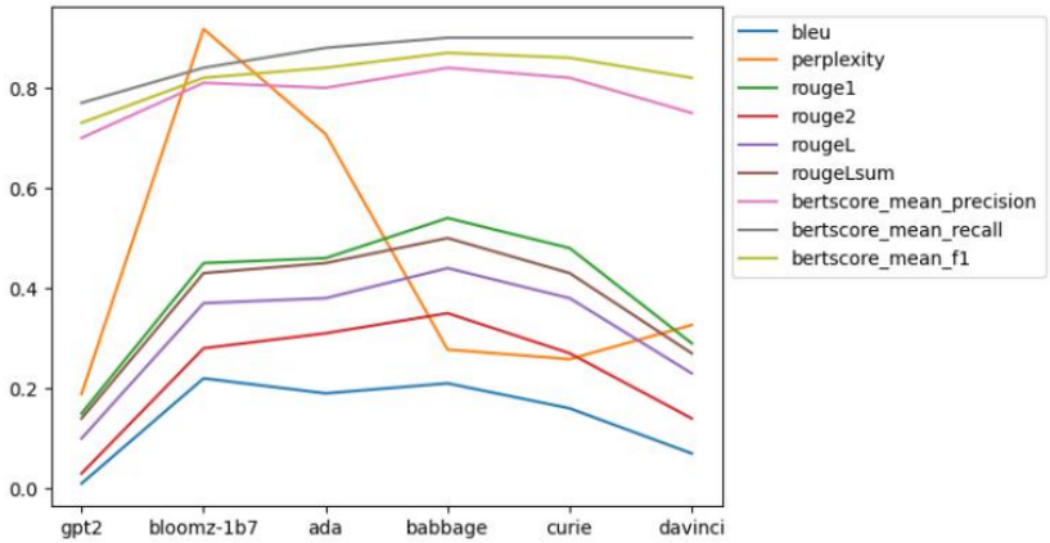

Figure 5.2.1: Evaluation Scores Plot

Apart from Perplexity the rest seem to follow pattern, where the scores increase till babbage and then either stay flat or go down. Perplexity seems to be doing largely the opposite. The plot is discussed in detail below.

- GPT2 model is very poor at zero-shot generation task and is much smaller and older than the other models. As such, it's no surprise the scores for GPT2 are the lowest for each metric.
- There is a steep rise from GPT2 to Bloomz-1B7 model for all scores. Although a 1 billion parameter model is relatively small by language model standards, it still shows much better performance than GPT2. As shown in Section 5.3, the story generated from Bloomz-1B7 model closely follows the outline. This leads to having high scores for BLEU, Rouge and BERTScore.
- Ada is the smallest model that OpenAI provides and as can be seen in the plot, it has similar performance to Bloomz-1B7. Babbage is a slightly better performing version of Ada. Till Babbage, the generated story closely resembles the outline and is of similar size. Thus, the scores keep increasing.
- Starting with Curie the generated story starts getting longer and more varied. It still follows the outline but gets increasingly more creative and expansive, with the story from Davinci model having multiple paragraphs and being a lot more descriptive than the previous models. This increase in length and diversity of words used, leads to decrease in the scores even though the story follows the same outline.
- Perplexity was calculated with distilgpt2 model as base. As such smaller models are more likely to have similar probability distribution of words. As the model gets bigger (trained on larger corpus of data), with more varied and creative choice of words, the perplexity decreases.

5.3 Qualitative Evaluation

This section shows 2 samples of outlines and the stories generated with different model using those outlines.

5.3.1 Generated Story Example – 1

```
Ignacio wants to play a sport while he is in college.
Since he was a good swimmer, he decides to try out for swim the team.
Ignacio makes it onto the team easily.
At the first swim meet, Ignacio wins second place!
Ignacio won a silver medal.
```

Figure 5.3.1.1 Outline – 1

```
This story ends at the first round of the French Open.
The story starts with a girl as she enters a bar.
The story ends with Ignacio's head flying in the air while he is at the bar.
On the fourth round, Ignacio gets his bronze medal.
On the fifth round, Ignacio gets his bronze medal.
On the sixth round, Ignacio gets his bronze medal.
The story ends with a story from the team.
The story begins with a group of girls as they go to the beach in the park of Saint
Pompon.
On the sixth round, the girls are at the beach.
On the seventh round, the girls are at the beach.
One of them says "This is great, I'm very happy."
The next moment, Ignacio's body is in a pool.
The next moment, he is floating under the water.
There is a very awkward moment in the story. The story ends with a girl saying "A ve
ry proud girl" when the girl says "I'm very proud of you."
Here are a few interesting things:
The story starts in Saint Pompon.
The story ends with a short story.
And here is a short story:
```

Figure 5.3.1.2 Generated Story: GPT2 – 1

At the first meet, Ignacio swims very well. He runs well. He does not take any risks and wins easily. The team is very surprised by his performance at the first swim mee t. Ignacio does not take any risks and wins easily. The team is very surprised by hi s performance at the first swim meet.</s>

Figure 5.3.1.3 Generated Story: Bloomz-1B7 – 1

Ignacio is in college and he's a swimmer. He wants to play a sport, so he tries out for the team. He makes it onto the team easily. At the first swim meet, Ignacio wins second place! This makes him very proud and he is very excited to finally be playing in a competition!

Figure 5.3.1.4 Generated Story: Ada – 1

Ignacio was so excited to try out for swim the team his first year of college. He qu ickly made it onto the team and excelled, winning second place in his division. The next meet, Ignacio won a silver medal. This continued to go well for him as he won m ore medals and cups until finally he set the world record in a major event. With thi s new record, Ignacio became an expert swimmer and was able to secure a scholarship tocollege.

Figure 5.3.1.5 Generated Story: Babbage – 1

Ignacio loved swimming ever since he was able to swim competently. He had always wan ted to try out for a swim team, so when he saw an opportunity to do so while in coll ege, he took it. He made it onto the team easily and at the first swim meet, he won second place! Ignacio was ecstatic and thrilled with his win. However, his joy was s hort-lived as he learned that he had also won a silver medal. This made him even mor e proud of himself and gave him a boost of confidence for future swim meets.

Figure 5.3.1.6 Generated Story: Curie – 1

```
Ignacio had been considering playing a sport in college. He had played sports in hig
h school, but swimming was always his favorite. With that in mind, he decided to try
out for the swim team at his college.

When Ignacio arrived for the tryouts, he saw several swimmers already competing and
knew it would be tough competition. However, Ignacio was a good swimmer and performe
d well during the tryouts. After being evaluated by the coach and showing off his sk
ills in the pool, Ignacio made it onto the team easily.

At their first swim meet of the season, everyone was excited to see what they could
do as a team. There were some excellent swimmers competing against them, so everyone
on the team knew they had their work cut out for them! During one of his events, Ign
acio amazed everyone when he came in second place! Everyone cheered him on as he won
a silver medal and brought pride to his team.

Ignacio was proud of himself for making it onto an elite college swim team and doing
so well at their first meet of the year - not many people can say that! He now looke
d forward to pushing himself even further with this new opportunity that presented i
tself before him.
```

Figure 5.3.1.7 Generated Story: Davinci – 1

Figure 5.3.1.1 shows Outline-1. It is 5-sentence story-plot taken from the ROCStories dataset.

Figure 5.3.1.2 shows the story generated using GPT2 model based on the outline. Although the story takes some elements from the outline, the story is mostly gibberish. It lacks any kind of Coherence or Discourse. The hyperparameters of GPT2 model can be tuned further to generate decent quality stories, but this was an unconstrained generation. When the generation is performed conditioned on the outline, the results are not satisfactory. To get better results with GPT2 model, it needs to be fine-tuned with outline-story pairs as done by (Fang et al, 2021).

Figure 5.3.1.3 shows the story generated using Bloomz-1B7 model. This follows very closely to the outline. But the result is, for the most part, a rephrasing of the outline and adds very little that's new to the story. It also contains some repetition. Perhaps using a larger version of the Bloomz model would resolve some of these issues.

Figures 5.3.1.4, 5.3.1.5, and 5.3.1.6 show the results from Ada, Babbage, and Curie models respectively. The results get increasingly better from Ada to Curie. The story follows the outline, the results are coherent. The Curie model adds more narrative description to the outline.

Figure 5.3.1.7 shows the story generated using Davinci model. This result is distinctly better than the previous ones. The generated story feels like something written by a human. It's broken into multiple paragraphs, each paragraph talking about a separate idea while being connected to the previous. It introduces new and creative ideas to fill in the gaps between the outline sentences, while sticking to the overall structure of the outline.

5.3.2 Generated Story Example – 2

```
Jesse had just started fifth grade.
He tried to make friends with the other boys, but he couldn't.
He decided to make friends with the girls instead.
They girls welcomed him into their social groups happily.
Jesse was picked on by the other boys in school.
```

Figure 5.3.2.1 Outline – 2

The other boys tried to take advantage of him, but Jesse just ignored them.
The boy who did this started to bully the girl who was doing the bullying.
Then there was the boy who did all of this, with the assistance of Jesse's father.
As he started getting more and more friends, the boy started to bully Jesse, but he never hit his partner.
The girl who had started this bullying had been fighting to get Jesse to stop, and the boy began to fight to get her to stop, and he didn't.
But this did not stop his behavior from escalating.
But eventually, Jesse would start to get the girl to stop, and the boy would attack her.
After this, the girl and Jesse started to fight, but when Jesse started to fight back, he got up and started hitting her.
Jesse, his father, and the other girls in the girls group became friends, because they all knew how much they liked each other and how much they knew each other.
Jesse started to lose friends, and he started to get worse.
They were friends that Jesse always had.
They always knew each other, but Jesse was losing friends because he did things that he shouldn't have. He got up and started hitting people, but when he did, he didn't hit the girl he was standing over.
He was hurting people in front of him, and he was hurting him in front of a girl he didn't care about.
But now that the girl he was standing over was a girl he cared about, the boy he was punching, he could do something to her.
He would punch and punch and punch and punch.
He did not even know the girl he was standing over, nor did he care that she was his girlfriend.
But when the girl he was fighting in front of her got punched in the face, he punched her in the face and the girl he was punching got punched in the face.
The boy that was doing the bullying had become friends with her, and the girl that he was punching had become friends with his parents, and the boy that was punching got kicked in the face by the bullies.
The girl that was fighting with the boys was just a friend.
But now that the girl he was standing over was a girl that he cared about, the boy he was punching, he could do something to her, and the girl he was punching got kicked

Figure 5.3.2.2 Generated Story: GPT2 – 2

Jesse went home and cried. The girls and the other boys started laughing at him.
He decided to make friends with the girls instead. They girls welcomed him into their social groups happily.</s>

Figure 5.3.2.3 Generated Story: Bloomz-1B7 – 2

Jesse had tried his best to make friends with the other boys in school, but he just couldn't.
The other boys picked on him and it was just another thing that made his life harder.
But when he went to their girls' schools, they wereidently doing better.
Jesse decided to make some new friends at a social group for girls and he was holes into them happily.
Jesse loved spending time with his new friends and he loved being one of the only girls in the group.

Figure 5.3.2.4 Generated Story: Ada – 2

Jesse tried to make friends with the other boys in school, but he couldn't. He decided to make friends with the girls instead. They girls welcomed him into their social groups happily. The other boys in school started picking on Jesse and making fun of him. Jesse felt left out and abandoned by the girls.

Figure 5.3.2.5 Generated Story: Babbage – 2

Jesse had always been a shy guy. He was worried that he wouldn't be able to make friends and fit in with the other boys in his class. But, after some trial and error, Jesse realized that making friends with the girls was the best strategy for him. They welcomed him warmly into their groups and didn't treat him like he was an outsider. Jesse learned that it's important to have both male and female friends to feel comfortable in any social setting. As long as Jesse maintains a healthy balance of male and female friends, he'll be sure to have a positive social experience no matter where he goes in life.

Figure 5.3.2.6 Generated Story: Curie – 2

```
Jesse had just started fifth grade. He was the new kid and he wasn't sure how to fit
in. He tried to make friends with the other boys, but they weren't too receptive of
him. No matter how hard Jesse tried, it seemed that the boys always treated him with
cold indifference which left him feeling lonely and isolated.

Desperate for companionship, Jesse decided to try a different approach; he decided t
o make friends with the girls instead. Surprisingly, the girls welcomed him into the
ir social groups happily without any hesitation or judgement. Soon enough, Jesse fou
nd himself surrounded by a group of kind and caring female friends who genuinely wan
ted what was best for him and accepted him for who he was.

Unfortunately, even though things were going well for Jesse on one front, he still f
aced challenges from his male classmates at school. The other boys consistently gave
Jesse a hard time about being in the company of girls rather than them leading to so
me pretty shameful moments for poor Jesse in class - especially when it came down to
sports day or having lunch together at school where all of his peers could witness h
is exclusion from their group first hand. Despite this however,Jesse remained determ
ined not let anything stand in between himself and his newfound friendships with the
se brave young ladies before him - no matter what happened next.
```

Figure 5.3.2.7 Generated Story: Davinci – 2

In the previous story, in Section 5.3.1, there was only one character and single chain of thought – swimming and winning a medal. The story outline in Figure 5.3.2.1 is more complex. Not only does it have the main character (Jesse) with his actions and motivation, but the story also includes the Jesse's interactions with the two groups of boys and girls in his class and their actions. This ends up confusing the models during generation.

The GPT2 result in Figure 5.3.2.2 is again gibberish. Also instead of following the outline it tries to continue the story from where outline finishes.

Bloomz-1B7 and Ada don't do much better in Figures 5.3.2.3 and 5.3.2.4.

In Figure 5.3.2.5, Babbage simply re-phrase the outline. But even then, the last line break reasoning and contradicts the outline.

In Figure 5.3.2.6, Curie does better and expands upon the outline. But it misses out on one key plot-point from the outline.

Davinci, in Figure 5.3.2.7, is the only model that produces satisfactory results.

5.4 Summary

After looking at the results from different models, it can be claimed that Davinci model clearly generates the best stories while still following the outline. It can also be claimed that the Quantitative scoring metrics are not sufficient to compare the quality of the stories.

CHAPTER 6: CONCLUSION & RECOMMENDATIONS

6.1 Discussion and Conclusion

Some conclusions drawn during this thesis are discussed below.

6.1.1 Best Model for Story Generation

As shown in the examples in Section 5.3, larger language models tend to produce better results than their smaller counterparts. For a model like Davinci, it generates human level stories even in a zero-shot setting. While size of the of the model is important, another important factor to notice is the instruction tuning that makes the Davinci model much better at following instruction provided as prompts.

6.1.2 Quantitative Metrics for Story Generation

Another key point that can be noticed by comparing Sections 5.2 and 5.3 is that traditional Quantitative metrics used for Text Generation tasks are not very good metrics to identify high-quality stories. Even though the stories generated by Davinci model are qualitatively far better than the others, the metrics cannot identify that properly. These metrics were created for other text-generation tasks such as Machine Translational and Summarization. But Open-ended story generation requires different metrics to determine story quality.

6.2 Contribution

This thesis works adds the following contributions towards existing literature:

- Benchmarks scores for Story generation task using both open-source models and commercial APIs.
- Notebooks/code for Data Preprocessing, Generation and Evaluation.

6.3 Future Work

6.3.1 Benchmark against Other Models

In this thesis, although the OpenAI models stand out, that could be because the open-source models they were compared against were smaller. GPT-2 models are old and outdated. While BloomZ models are more recent and based on the best performing Instruction tuning techniques, the 1.7 billion parameter model used in this thesis is not the most powerful model available. Due to GPU memory constraints, the experiments had to be performed on smaller model instead. In future work, the larger models can also be experimented on and added to the benchmarking.

On similar lines, from an Open-Source models perspective, only the GPT-2 and BloomZ models were used in the thesis. There are other models such Meta's OPT and OPT-IML that can be added to the benchmarking. Even the BloomZ model can be compared against previous Bloom models to verify the improvement in results.

As this thesis was being written, OpenAI introduced the massively popular ChatGPT (OpenAI, 2023a) framework. Although ChatGPT is based on the same InstructGPT technique that is used in OpenAI's other models, it has some further tweaks to make conversations and generation even smoother. This can also to be added to the benchmarking.

6.3.2 Benchmark against Existing Approaches

This thesis compares results across different models. There are other works that have tried similar approaches for Story Generation (Sun et al., 2020, Fang et al., 2021). In the future, the results from this thesis can be compared against similar existing works.

6.3.3 Use Better Evaluation Metrics

This thesis evaluated the generated stories against some traditional metrics that have been used for text generation. However, while these metrics work well for tasks such as Machine-translation of Abstractive Summarization, they do not translate very well to a less well-defined task such as Open-World Story Generation. One problem is that unlike machine-translation and

summarization, generated stories tend to lack a reference text to be compared against, which these metrics expect. Second, these metrics cannot capture features specific to the task of story generation such as narrative, discourse, creativity, suspense, etc.

Recent works have introduced benchmarks specifically for story generation task such as OpenMEVA (Guan et al., 2021) and HANNA (Chhun et al., 2022). In future work, these benchmarks can be tried out.

REFERENCES

Aghajanyan, A., Okhonko, D., Lewis, M., Joshi, M., Xu, H., Ghosh, G. and Zettlemoyer, L., (2021) HTLM: Hyper-Text Pre-Training and Prompting of Language Models. [online] Available at: https://arxiv.org/abs/2107.06955v1 [Accessed 10 Feb. 2023].

Alhussain, A.I. and Azmi, A.M., (2021) Automatic Story Generation. *ACM Computing Surveys (CSUR)*, [online] 54(5), pp.1-38. Available at: https://dl.acm.org/doi/10.1145/3453156 [Accessed 7 Oct. 2022].

Ammanabrolu, P., Tien, E., Cheung, W., Luo, Z., Ma, W., Martin, L.J. and Riedl, M.O., (2019) Guided Neural Language Generation for Automated Storytelling. [online] pp.46–55. Available at: https://aclanthology.org/W19-3405 [Accessed 5 Jan. 2023].

Ammanabrolu, P., Tien, E., Cheung, W., Luo, Z., Ma, W., Martin, L.J. and Riedl, M.O., (2020) Story Realization: Expanding Plot Events into Sentences. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 3405, pp.7375–7382. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/6232 [Accessed 5 Jan. 2023].

Araz, K., (2020) Transformer Neural Networks for Automated Story Generation. *Dissertations*. [online] Available at: https://arrow.tudublin.ie/scschcomdis/211 [Accessed 5 Jan. 2023].

Bach, S.H., Sanh, V., Yong, Z.-X., Webson, A., Raffel, C., Nayak, N. V., Sharma, A., Kim, T., Bari, M.S., Fevry, T., Alyafeai, Z., Dey, M., Santilli, A., Sun, Z., Ben-David, S., Xu, C., Chhablani, G., Wang, H., Fries, J.A., Al-shaibani, M.S., Sharma, S., Thakker, U., Almubarak, K., Tang, X., Radev, D., Jiang, M.T.-J. and Rush, A.M., (2022) PromptSource: An Integrated Development Environment and Repository for Natural Language Prompts. [online] pp.93–104. Available at: https://arxiv.org/abs/2202.01279v3 [Accessed 16 Feb. 2023].

Bahdanau, D., Cho, K.H. and Bengio, Y., (2014) Neural Machine Translation by Jointly Learning to Align and Translate. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. [online] Available at: https://arxiv.org/abs/1409.0473v7 [Accessed 12 Dec. 2022].

Bakhtin, A., Deng, Y., Ott, M., Ranzato, M.' A. and Szlam, A., (2021) Residual Energy-Based Models for Text. *Journal of Machine Learning Research*, [online] 22, pp.1–41. Available at: http://jmlr.org/papers/v22/20-326.html. [Accessed 23 Oct. 2022].

Balasubramanian, N., Soderland, S. and Etzioni, O., (2012) Rel-grams: A Probabilistic Model of Relations in Text. pp.101–105.

Bao, H., Dong, L., Wei, F., Wang, W., Yang, N., Liu, X., Wang, Y., Gao, J., Piao, S., Zhou, M. and Hon, H.-W., (2020) *UniLMv2: Pseudo-Masked Language Models for Unified Language Model Pre-Training*. pp. 642-652. Available at: https://proceedings.mlr.press/v119/bao20a.html [Accessed 10 Feb. 2023].

Baumgartner, J., Zannettou, S., Keegan, B., Squire, M. and Blackburn, J., (2020) The Pushshift Reddit Dataset. Proceedings of the International AAAI Conference on Web and Social Media,

[online] 14, pp.830–839. Available at: https://ojs.aaai.org/index.php/ICWSM/article/view/7347 [Accessed 16 Feb. 2023].

Behrooz, M., Robertson, J. and Jhala, A., (2019) Story Quality as a Matter of Perception: Using Word Embeddings to Estimate Cognitive Interest. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, [online] 151, pp.3–9. Available at: https://ojs.aaai.org/index.php/AIIDE/article/view/5217 [Accessed 5 Jan. 2023].

Ben-David, E., Oved, N. and Reichart, R., (2021) PADA: Example-based Prompt Learning for on-the-fly Adaptation to Unseen Domains. *Transactions of the Association for Computational Linguistics*, [online] 10, pp.414–433. Available at: https://arxiv.org/abs/2102.12206v4 [Accessed 10 Feb. 2023].

Bengio, Y., Courville, A. and Vincent, P., (2013) Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 358, pp.1798–1828.

Bertinetto, L., Henriques, J.F., Valmadre, J., Torr, P.H.S. and Vedaldi, A., (2016) Learning feed-forward one-shot learners. *Advances in Neural Information Processing Systems*, 29.

BigScience, Le Scao, T., Fan, A., Akiki, C., Pavlick, E., Ili, S., Hesslow, D., Castagné, R., Sasha Luccioni, A., Yvon, F., Gallé, M., Tow, J., Rush, A.M., Biderman, S., Webson, A., SasankaAmmanamanchi, P., Wang, T., Sagot, B., Muennighoff, N., Villanova del Moral, A., Ruwase, O., Bawden, R., Bekman, S., McMillan-Major, A., Wolf, T., Beltagy, I., Nguyen, H., Saulnier, L., Tan, S., Ortiz Suarez, P., Sanh, V., Laurençon, H., Jernite, Y., Launay, J., Mitchell, M., Raffel Dataset Aaron Gokaslan, C., Simhi, A., Soroa, A., Fikri Aji, A., Alfassy, A., Rogers, A., KreisbergNitzav, A., Xu, C., Mou, C., Emezue, C., Klamm, C., Leong, C., Raffel, C., van Strien, D., IfeoluwaAdelani, D., Radev, D., González Pon-ferrada, E., Levkovizh, E., Kim, E., Bar Natan, E., De Toni, F., Dupont, G., Kruszewski, G., Pistilli, G., Elsahar, H., Benyamina, H., Tran, H., Yu, I., Abdulmumin, I., Johnson, I., Gonzalez-Dios, I., de la Rosa, J., Chim, J., Dodge, J., Zhu, J., Chang, J., Frohberg, J., Tobing, J., Bhattachar-jee, J., Almubarak, K., Chen, K., Lo, K., Von Werra, L., Weber, L., Phan, L., Ben allal, L., Tanguy, L., Dey, M., Romero Muñoz, M., Masoud, M., Grandury, M., Huang, M., Coavoux, M., Singh, M., Tian-Jian Jiang, M., Chien Vu, M., Jauhar, M.A., Ghaleb, M., Subramani, N., Kassner, N., Khamis, N., Nguyen, O., Espejel, O., de Gibert, O., Villegas, P., Henderson, P., Colombo, P., Amuok, P., Lhoest, Q., Harliman, R., Bommasani, R., Luis López, R., Ribeiro, R., Osei, S., Pyysalo, S., Nagel, S., Bose, S., Hassan Muhammad, S., Sharma, S., Longpre, S., Nikpoor, S., Silberberg, S., Pai, S., Zink, S., Timponi Torrent, T., Schick, T., Thrush, T., Danchev, V., Nikoulina, V., Laippala, V., Lepercq, V., Prabhu, V., Alyafeai, Z., Talat Tokenization Arun Raja, Z., Heinzerling, B., Si, C., Emre Ta, D., Salesky, E., Mielke, S.J., Lee, W.Y., Sharma, A., Santilli, A., Chaffin, A., Stiegler, A., Raja, A., Datta, D., Szczechla, E., Chhablani, G., Wang, H., Pandey, H., Strobelt, H., Alan Fries, J., Rozen, J., Gao, L., Sutawika, L., Saiful Bari, M., Al-shaibani, M.S., Manica, M., Nayak, N., Teehan, R., Albanie, S., Shen, S., Ben-David, S., Bach, S.H., Kim, T., Bers, T., Fevry, T., Neeraj, T., Thakker, U., Raunak, V., Tang, X., Yong, Z.-X., Sun, Z., Brody, S., Uri, Y., Tojarieh, H., Roberts, A., Won Chung, H., Tae, J., Phang, J., Muen-nighoff, N., Press, O., Li, C.,

Narayanan, D., Bourfoune, H., Casper, J., Rasley, J., Ryabinin, M., Mishra, M., Zhang, M., Shoeybi, M., Peyrounette, M., Patry, N., Tazi, N., Sanseviero, O., von Platen, P., Cor-nette, P., François Lavallée, P., Lacroix, R., Rajbhandari, S., Gandhi, S., Smith, S., Requena, S., Patil, S., Dettmers, T., Baruwa, A., Singh, A., Cheveleva, A., Ligozat, A.-L., Subramonian, A., Névéol, A., Lovering, C., Garrette, D., Tunuguntla, D., Reiter, E., Taktasheva, E., Voloshina, E., Bogdanov, E., Indra Winata, G., Schoelkopf, H., Kalo, J.-C., Novikova, J., Zosa Forde, J., Clive, J., Kasai, J., Kawamura, K., Hazan, L., Carpuat, M., Clinciu, M., Kim, N., Cheng, N., Serikov, O., Antverg, O., van der Wal, O., Zhang, R., Zhang, R., Gehrmann, S., Mirkin, S., Pais, S., Shavrina, T., Scialom, T., Yun, T., Limisiewicz, T., Rieser, V., Protasov, V., Mikhailov, V., Pruksachatkun, Y., Belinkov, Y., Bamberger, Z., Kasner, Z., Talat, Z., Gokaslan, A., Rueda, A., Pestana, A., Feizpour, A., Khan, A., Faranak, A., Santos, A., Hevia, A., Unldreaj, A., Aghagol, A., Abdollahi, A., Tammour, A., HajiHosseini, A., Behroozi, B., Ajibade, B., Saxena, B., Muñoz Ferrandis, C., Contractor, D., Lansky, D., David, D., Kiela, D., Nguyen, D.A., Tan, E., Baylor, E., Ozoani, E., Mirza, F., Ononiwu, F., Rezanejad, H., Jones, H., Bhattacharya, I., Solaiman, I., Sedenko, I., Nejadgholi, I., Passmore, J., Seltzer, J., Bonis Sanz, J., Dutra, L., Samagaio, M., Elbadri, M., Ma-soud, M., Mieskes, M., Gerchick, M., Akinlolu, M., McKenna, M., Qiu, M., Ghauri, M., Burynok, M., Abrar, N., Rajani, N., Elkott, N., Fahmy, N., Samuel, O., An, R., Kromann, R., Hao, R., Alizadeh, S., Shubber, S., Wang, S., Roy, S., Viguier, S., Le, T., Oyebade, T., Le, T., Yang, Y., Nguyen, Z., Yong Applications Abhinav Ramesh Kashyap, Z.-X., Palasciano, A., Callahan, A., Shukla, A., Miranda-Escalada, A., Singh, A., Beilharz, B., Wang, B., Brito, C., Muñoz Ferrandis, B., Zhou, C., Jain, C., Xu, C., Fourrier, C., León Periñán, D., Molano, D., Yu, D., Manjavacas, E., Barth, F., Fuhrimann, F., Altay, G., Bayrak, G., Burns, G., Vrabec, H.U., Bello, I., Dash, I., Kang, J., Giorgi, J., Golde, J., David Posada, J., RangasaiSivaraman, K., Bulchandani, L., Liu, L., Shinzato, L., Hahn de Bykhovetz, M., Takeuchi, M., Pàmies, M., Castillo, M.A., Nezhurina, M., Sänger, M., Samwald, M., Cullan, M., Weinberg, M., De Wolf, M., Mihaljcic, M., Liu, M., Freidank, M., Kang, M., Seelam, N., Dahlberg, N., Michio Broad, N., Muellner, N., Fung, P., Haller, P., Chandrasekhar, R., Eisenberg, R., Martin, R., Canalli, R., Su, R., Su, R., Cahyawijaya, S., Garda, S., Deshmukh, S.S., Mishra, S., Kiblawi, S., Ott, S., Sang-aroonsiri, S., Kumar, S., Schweter, S., Bharati, S., Laud, T., Gigant, T., Kainuma, T., Kusa, W., Labrak, Y., Shailesh Bajaj, Y., Venkatraman, Y., Xu, Y., Xu, Y., Xu, Y., Tan, Z., Xie, Z., Ye, Z., Bras, M. and Belkada, Y., (2022) BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. [online] Available at: https://arxiv.org/abs/2211.05100v2 [Accessed 16 Feb. 2023].

Bisk, Y., Holtzman, A., Thomason, J., Andreas, J., Bengio, Y., Chai, J., Lapata, M., Lazaridou, A., May, J., Nisnevich, A., Pinto, N. and Turian, J., (2020) Experience Grounds Language. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.8718–8735. Available at: https://arxiv.org/abs/2004.10151v3 [Accessed 23 Oct. 2022].

Black, J.B. and Bower, G.H., (1980) Story understanding as problem-solving. *Poetics*, 91–3, pp.223–250.

Black, J.B. and Wilensky, R., (1979) An Evaluation of Story Grammars*. *Cognitive Science*, [online] 33, pp.213–229. Available at: https://onlinelibrary.wiley.com/doi/full/10.1207/s15516709cog0303_2 [Accessed 5 Jan. 2023].

Black, S., Gao, L., Wang, P., Leahy, C. and Biderman, S., (2021) GPT-Neo: Large scale autoregressive language modeling with mesh tensorflow. [online] Available at: https://zenodo.org/record/5551208 [Accessed 13 Feb. 2023].

Bosselut, A., Celikyilmaz, A., He, X., Gao, J., Huang, P. sen and Choi, Y., (2018) Discourse-Aware Neural Rewards for Coherent Text Generation. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, [online] 1, pp.173–184. Available at: https://arxiv.org/abs/1805.03766v1 [Accessed 23 Oct. 2022].

Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., Mccandlish, S., Radford, A., Sutskever, I. and Amodei, D., (2020) Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems*, [online] 33, pp.1877–1901. Available at: https://commoncrawl.org/the-data/ [Accessed 10 Feb. 2023].

Chai, D., Wu, W., Han, Q., Wu, F. and Li, J., (2020) *Description Based Text Classification with Reinforcement Learning, pp. 1371-1382. PMLR*. Available at: https://proceedings.mlr.press/v119/chai20a.html [Accessed 10 Feb. 2023].

Chambers, N. and Jurafsky, D., (2008) Unsupervised Learning of Narrative Event Chains. *In Proceedings of ACL-08: HLT* (pp. 789-797).

Chambers, N. and Jurafsky, D., (2009) Unsupervised Learning of Narrative Schemas and their Participants. *ACL and AFNLP*, pp.602–610.

Chambers, N., Wang, S. and Jurafsky, D., (2007) Classifying Temporal Relations Between Events. [online] pp.173–176. Available at: http://nlp.stanford.edu/software/lex-parser.shtml [Accessed 5 Jan. 2023].

Chaturvedi, S., Peng, H. and Roth, D., (2017) Story Comprehension for Predicting What Happens Next. *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.1603–1614. Available at: https://aclanthology.org/D17-1168 [Accessed 5 Jan. 2023].

Chen, G., Liu, Y., Luan, H., Zhang, M., Liu, Q. and Sun, M., (2021) Learning to Generate Explainable Plots for Neural Story Generation. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 29, pp.585–593.

Chen, J., Chen, J. and Yu, Z., (2019) Incorporating Structured Commonsense Knowledge in Story Completion. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 3301,

pp.6244–6251. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/5183 [Accessed 5 Jan. 2023].

Chhun, C., Colombo, P., Clavel, C. and Suchanek, F.M., (2022) Of Human Criteria and Automatic Metrics: A Benchmark of the Evaluation of Story Generation. [online] Available at: https://arxiv.org/abs/2208.11646v4 [Accessed 13 Feb. 2023].

Choi, Y., Kim, S. and Lee, J.H., (2016) Recurrent Neural Network for Storytelling. *Proceedings - 2016 Joint 8th International Conference on Soft Computing and Intelligent Systems and 2016 17th International Symposium on Advanced Intelligent Systems, SCIS-ISIS 2016*, pp.841–845.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H.W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A.M., Pillai, T.S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S. and Fiedel, N., (2022) PaLM: Scaling Language Modeling with Pathways. [online] Available at: https://arxiv.org/abs/2204.02311v5 [Accessed 16 Feb. 2023].

Christiano, P.F., Leike, J., Brown, T.B., Martic, M., Legg, S. and Amodei, D., (2017) Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems*, [online] 2017-December, pp.4300–4308. Available at: https://arxiv.org/abs/1706.03741v3 [Accessed 16 Feb. 2023].

Chu, C., Dabre, R. and Kurohashi, S., (2017) An Empirical Comparison of Domain Adaptation Methods for Neural Machine Translation. *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, [online] 2, pp.385–391. Available at: https://aclanthology.org/P17-2061 [Accessed 10 Feb. 2023].

Chung, J., Gulcehre, C., Cho, K. and Bengio, Y., (2014) Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. [online] Available at: https://arxiv.org/abs/1412.3555v1 [Accessed 12 Dec. 2022].

Clark, E., August, T., Serrano, S., Haduong, N., Gururangan, S. and Smith, N.A., (2021) All That's 'Human' Is Not Gold: Evaluating Human Evaluation of Generated Text. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.7282–7296. Available at: https://arxiv.org/abs/2107.00061v2 [Accessed 23 Oct. 2022].

Collobert, R., Weston, J., Com, J., Karlen, M., Kavukcuoglu, K. and Kuksa, P., (2011) Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12, pp.2493–2537.

Commoncrawl.org, (2023) *So you're ready to get started. – Common Crawl*. [online] Available at: https://commoncrawl.org/the-data/get-started/ [Accessed 16 Feb. 2023].

Cook, W.W., (2011) Plotto : the master book of all plots. p.40.

Cui, L., Wu, Y., Liu, J., Yang, S. and Zhang, Y., (2021) Template-Based Named Entity Recognition Using BART. *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, [online] pp.1835–1845. Available at: https://arxiv.org/abs/2106.01760v1 [Accessed 10 Feb. 2023].

cs.rochester.edu, (2022) *ROCStories and the Story Cloze Test*. [online] Available at: https://cs.rochester.edu/nlp/rocstories/ [Accessed 26 Oct. 2022].

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., (2019) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, [online] 1, pp.4171–4186. Available at: https://arxiv.org/abs/1810.04805v2 [Accessed 10 Feb. 2023].

Doddapaneni, S., Ramesh, G., Khapra, M.M., Kunchukuttan, A., Kumar, P. and Madras, I., (2021) A Primer on Pretrained Multilingual Language Models. [online] Available at: https://arxiv.org/abs/2107.00676v2 [Accessed 10 Feb. 2023].

Dodge, J., Ilharco, G., Schwartz, R., Farhadi, A., Hajishirzi, H. and Smith, N., (2020) Fine-Tuning Pretrained Language Models: Weight Initializations, Data Orders, and Early Stopping. [online] Available at: https://arxiv.org/abs/2002.06305v1 [Accessed 10 Feb. 2023].

Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M. and Hon, H.-W., (2019a) Unified Language Model Pre-training for Natural Language Understanding and Generation. *Advances in Neural Information Processing Systems*, [online] 32. Available at: https://github.com/microsoft/unilm. [Accessed 12 Dec. 2022].

Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M. and Hon, H.-W., (2019b) Unified Language Model Pre-training for Natural Language Understanding and Generation. *Advances in Neural Information Processing Systems*, [online] 32. Available at: https://github.com/microsoft/unilm. [Accessed 10 Feb. 2023].

Dou, Y., Forbes, M., Koncel-Kedziorski, R., Smith, N.A. and Choi, Y., (2021) Is GPT-3 Text Indistinguishable from Human Text? Scarecrow: A Framework for Scrutinizing Machine Text. [online] pp.7250–7274. Available at: https://arxiv.org/abs/2107.01294v3 [Accessed 23 Oct. 2022].

Dou, Z.Y., Liu, P., Hayashi, H., Jiang, Z. and Neubig, G., (2020) GSum: A General Framework for Guided Neural Abstractive Summarization. *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, [online] pp.4830–4842. Available at: https://arxiv.org/abs/2010.08014v3 [Accessed 10 Feb. 2023].

Dziri, N., Madotto, A., Zaiane, O. and Bose, A.J., (2021) Neural Path Hunter: Reducing Hallucination in Dialogue Systems via Path Grounding. *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.2197–2214. Available at: https://arxiv.org/abs/2104.08455v2 [Accessed 23 Oct. 2022].

Eger, M., Potts, C.M., Barot, C. and Young, R.M., (2015) Plotter: Operationalizing the Master Book of All Plots. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, [online] 114, pp.30–33. Available at: https://ojs.aaai.org/index.php/AIIDE/article/view/12835 [Accessed 9 Feb. 2023].

Fadaee, M., Bisazza, A. and Monz, C., (2017) Data Augmentation for Low-Resource Neural Machine Translation. *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, [online] 2, pp.567–573. Available at: http://arxiv.org/abs/1705.00440 [Accessed 10 Feb. 2023].

Fan, A., Lewis, M. and Dauphin, Y., (2018) Hierarchical Neural Story Generation. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, [online] 1, pp.889–898. Available at: https://arxiv.org/abs/1805.04833v1 [Accessed 9 Feb. 2023].

Fan, A., Lewis, M. and Dauphin, Y., (2019) Strategies for Structuring Story Generation. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, [online] pp.2650–2660. Available at: https://arxiv.org/abs/1902.01109v2 [Accessed 9 Feb. 2023].

Fang, L., Zeng, T., Liu, C., Bo, L., Dong, W. and Chen, C., (2021) Outline to Story: Fine-grained Controllable Story Generation from Cascaded Events. Available at: https://arxiv.org/abs/2101.00822

Finn, C., Abbeel, P. and Levine, S., (2017) Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *In International conference on machine learning* (pp. 1126-1135). PMLR. Available at: https://proceedings.mlr.press/v70/finn17a.html [Accessed 10 Feb. 2023].

Gao, J., Peng, B., Li, C., Li, J., Shayandeh, S., Liden, L. and Shum, H.-Y., (2020a) Robust Conversational AI with Grounded Text Generation. [online] Available at: https://arxiv.org/abs/2009.03457v1 [Accessed 23 Oct. 2022].

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S. and Leahy, C., (2020b) The Pile: An 800GB Dataset of Diverse Text for Language Modeling. [online] Available at: https://arxiv.org/abs/2101.00027v1 [Accessed 16 Feb. 2023].

Gao, T., Fisch, A. and Chen, D., (2020c) Making Pre-trained Language Models Better Few-shot Learners. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.3816–3830. Available at: https://arxiv.org/abs/2012.15723v2 [Accessed 10 Feb. 2023].

Gervás, P., (2009) Computational Approaches to Storytelling and Creativity. *AI Magazine*, [online] 303, pp.49–49. Available at: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2250 [Accessed 9 Feb. 2023].

Goldfarb-Tarrant, S., Chakrabarty, T., Weischedel, R. and Peng, N., (2020) Content Planning for Neural Story Generation with Aristotelian Rescoring. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.4319–4338. Available at: https://arxiv.org/abs/2009.09870v2 [Accessed 23 Oct. 2022].

Grangier, D. and Auli, M., (2017) QuickEdit: Editing Text & Translations by Crossing Words Out. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, [online] 1, pp.272–282. Available at: https://arxiv.org/abs/1711.04805v2 [Accessed 10 Feb. 2023].

Guan, J., Huang, F., Zhao, Z., Zhu, X. and Huang, M., (2020) A Knowledge-Enhanced Pretraining Model for Commonsense Story Generation. *Transactions of the Association for Computational Linguistics*, [online] 8, pp.93–108. Available at: https://aclanthology.org/2020.tacl-1.7 [Accessed 31 Aug. 2022].

Guan, J., Wang, Y. and Huang, M., (2019) Story Ending Generation with Incremental Encoding and Commonsense Knowledge. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 3301, pp.6473–6480. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/4612 [Accessed 9 Feb. 2023].

Guan, J., Zhang, Z., Feng, Z., Liu, Z., Ding, W., Mao, X., Fan, C. and Huang, M., (2021) OpenMEVA: A Benchmark for Evaluating Open-ended Story Generation Metrics. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.6394–6407. Available at: https://arxiv.org/abs/2105.08920v1 [Accessed 13 Feb. 2023].

Guyon, I., Weston, J. and Barnhill, S., (2002) Gene Selection for Cancer Classification using Support Vector Machines. 46, pp.389–422.

Hambardzumyan, K., Khachatrian, H. and May, J., (2021) WARP: Word-level Adversarial ReProgramming. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.4921–4933. Available at: https://arxiv.org/abs/2101.00121v2 [Accessed 10 Feb. 2023].

Han, X., Zhao, W., Ding, N., Liu, Z. and Sun, M., (2022) PTR: Prompt Tuning with Rules for Text Classification. *AI Open*, 3, pp.182–192.

Harrison, B., Purdy, C. and Riedl, M.O., (2017) Toward Automated Story Generation with Markov Chain Monte Carlo Methods and Deep Neural Networks. In *Proceedings of the AAAI*

Conference on Artificial Intelligence and Interactive Digital Entertainment (Vol. 13, No. 2, pp. 191-197). [online] Available at: www.aaai.org [Accessed 12 Feb. 2023].

Hochreiter, S. and Schmidhuber, J., (1997) Long Short-Term Memory. *Neural Computation*, 98, pp.1735–1780.

Holtzman, A., Buys, J., Du, L., Forbes, M. and Choi, Y., (2019) The Curious Case of Neural Text Degeneration. *CEUR Workshop Proceedings*, [online] 2540. Available at: https://arxiv.org/abs/1904.09751v2 [Accessed 9 Feb. 2023].

Hu, L., Li, J., Nie, L., Li, X.L. and Shao, C., (2017) What Happens Next? Future Subevent Prediction Using Contextual Hierarchical LSTM. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 311, pp.3450–3456. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/11001 [Accessed 9 Feb. 2023].

Ippolito, D., Grangier, D., Eck, D. and Callison-Burch, C., (2020) Toward Better Storylines with Sentence-Level Language Models. [online] pp.7472–7478. Available at: https://arxiv.org/abs/2005.05255v1 [Accessed 9 Feb. 2023].

Iyer, S., Lin, X.V., Pasunuru, R., Mihaylov, T., Simig, D., Yu, P., Shuster, K., Wang, T., Liu, Q., Koura, P.S., Li, X., O'Horo, B., Pereyra, G., Wang, J., Dewan, C., Celikyilmaz, A., Zettlemoyer, L. and Stoyanov, V., (2022) OPT-IML: Scaling Language Model Instruction Meta Learning through the Lens of Generalization. [online] Available at: https://arxiv.org/abs/2212.12017v3 [Accessed 16 Feb. 2023].

Jain, P., Agrawal, P., Mishra, A., Sukhwani, M., Laha, A. and Sankaranarayanan, K., (2017) Story Generation from Sequence of Independent Short Descriptions. [online] 7. Available at: https://arxiv.org/abs/1707.05501v2 [Accessed 9 Feb. 2023].

Jans, B., Bethard, S., Vulic, I., Moens, M.-F., Daelemans, W., Lapata, M. and Màrquez, L., (2012) Skip n-grams and ranking functions for predicting script events. *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, (pp. 336-344). ACL. [online] Available at: https://lirias.kuleuven.be/1572145 [Accessed 9 Feb. 2023].

Ji, H. and Huang, M., (2021) DiscoDVT: Generating Long Text with Discourse-Aware Discrete Variational Transformer. *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.4208–4224. Available at: https://arxiv.org/abs/2110.05999v1 [Accessed 23 Oct. 2022].

Ji, H., Ke, P., Huang, S., Wei, F., Zhu, X. and Huang, M., (2020) Language Generation with Multi-Hop Reasoning on Commonsense Knowledge Graph. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.725–736. Available at: https://arxiv.org/abs/2009.11692v1 [Accessed 23 Oct. 2022].

Jiang, Z., Anastasopoulos, A., Araki, J., Ding, H. and Neubig, G., (2020a) X-FACTR: Multilingual Factual Knowledge Retrieval from Pretrained Language Models. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the*

Conference, [online] pp.5943–5959. Available at: https://arxiv.org/abs/2010.06189v3 [Accessed 10 Feb. 2023].

Jiang, Z., Xu, F.F., Araki, J. and Neubig, G., (2020b) How can we know what language models know? *Transactions of the Association for Computational Linguistics*, [online] 8, pp.423–438. Available at: https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00324/96460/How-Can-We-Know-What-Language-Models-Know [Accessed 10 Feb. 2023].

Jurafsky, D. and Martin, J.H., (2000) Book Reviews Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. [online] Available at: www.cs.colorado.edu/~martin/slp.html. [Accessed 10 Feb. 2023].

Kaiser, Ł., Brain, G., Nachum, O., Google Brain, †, Roy, A., Tech, G. and Bengio, S., (2017) Learning to Remember Rare Events. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*. [online] Available at: https://arxiv.org/abs/1703.03129v1 [Accessed 10 Feb. 2023].

Kalchbrenner, N., Grefenstette, E. and Blunsom, P., (2014) A Convolutional Neural Network for Modelling Sentences. *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference*, [online] 1, pp.655–665. Available at: https://arxiv.org/abs/1404.2188v1 [Accessed 12 Dec. 2022].

Kartal, B., Koenig, J. and Guy, S.J., (2014) User-Driven Narrative Variation in Large Story Domains using Monte Carlo Tree Search. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. [online], pp. 69-76. Available at: www.ifaamas.org [Accessed 9 Feb. 2023].

Keskar, N.S., Mccann, B., Varshney, L.R., Xiong, C., Socher, R. and Research, S., (2019) CTRL: A Conditional Transformer Language Model for Controllable Generation. [online] Available at: https://arxiv.org/abs/1909.05858v2 [Accessed 23 Oct. 2022].

Khandelwal, U., He, H., Qi, P. and Jurafsky, D., (2018) Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, [online] 1, pp.284–294. Available at: https://arxiv.org/abs/1805.04623v1 [Accessed 9 Feb. 2023].

Khashabi, D., Min, S., Khot, T., Sabharwal, A., Tafjord, O., Clark, P. and Hajishirzi, H., (2020) UnifiedQA: Crossing Format Boundaries With a Single QA System. *Findings of the Association for Computational Linguistics Findings of ACL: EMNLP 2020*, [online] pp.1896–1907. Available at: https://arxiv.org/abs/2005.00700v3 [Accessed 10 Feb. 2023].

Kikuchi, Y., Neubig, G., Sasano, R., Takamura, H. and Okumura, M., (2016) Controlling Output Length in Neural Encoder-Decoders. *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.1328–1338. Available at: https://arxiv.org/abs/1609.09552v1 [Accessed 10 Feb. 2023].

Kotsiantis, S.B., (2007) Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, 31, pp.249–268.

Kumar, A., Irsoy, O., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., Zhong, V., Paulus, R. and Socher, R., (2016) Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. In *International conference on machine learning* (pp. 1378-1387). Available at: https://proceedings.mlr.press/v48/kumar16.html [Accessed 10 Feb. 2023].

Kumar, S. and Talukdar, P., (2021) Reordering Examples Helps during Priming-based Few-Shot Learning. *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, [online] pp.4507–4518. Available at: https://arxiv.org/abs/2106.01751v1 [Accessed 10 Feb. 2023].

Kybartas, B. and Bidarra, R., (2017) A Survey on Story Generation Techniques for Authoring Computational Narratives. *IEEE Transactions on Computational Intelligence and AI in Games*, 93, pp.239–253.

Lafferty, J., McCallum, A. and Pereira, F., (2001) Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Departmental Papers (CIS)*. [online] Available at: https://repository.upenn.edu/cis_papers/159 [Accessed 12 Dec. 2022].

León, C. and Gervás, P., (2010) The role of evaluation-driven rejection in the successful exploration of a conceptual space of stories. *Minds and Machines*, [online] 204, pp.615–634. Available at: https://link.springer.com/article/10.1007/s11023-010-9205-z [Accessed 9 Feb. 2023].

Lester, B., Al-Rfou, R. and Constant, N., (2021) The Power of Scale for Parameter-Efficient Prompt Tuning. *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.3045–3059. Available at: https://arxiv.org/abs/2104.08691v2 [Accessed 10 Feb. 2023].

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L. and Ai, F., (2020) BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. [online] pp.7871–7880. Available at: https://aclanthology.org/2020.acl-main.703 [Accessed 13 Feb. 2023].

Li, B., Lee-Urban, S., Johnston, G. and Riedl, M.O., (2013) Story Generation with Crowdsourced Plot Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 271, pp.598–604. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/8649 [Accessed 9 Feb. 2023].

Li, Q., Li, Z., Wei, J.-M., (顾彦慧) Y.G., Jatowt, A. and Yang, Z., (2018a) A Multi-Attention based Neural Network with External Knowledge for Story Ending Predicting Task. In *Proceedings of the 27th International Conference on Computational Linguistics* (pp. 1754-1762). Available at: https://aclanthology.org/C18-1149 [Accessed 9 Feb. 2023].

Li, X.L. and Liang, P., (2021) Prefix-Tuning: Optimizing Continuous Prompts for Generation. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and*

the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference, [online] pp.4582–4597. Available at: https://arxiv.org/abs/2101.00190v1 [Accessed 10 Feb. 2023].

Li, Z., Ding, X. and Liu, T., (2018b) Generating Reasonable and Diversified Story Ending Using Sequence to Sequence Model with Adversarial Training. In *Proceedings of the 27th International Conference on Computational Linguistics* (pp. 1033-1043). Available at: https://aclanthology.org/C18-1088 [Accessed 9 Feb. 2023].

Li, Z., Ding, X., Liu, T., Hu, J.E. and van Durme, B., (2021) Guided Generation of Cause and Effect. *IJCAI International Joint Conference on Artificial Intelligence*, [online] 2021-January, pp.3629–3636. Available at: http://arxiv.org/abs/2107.09846 [Accessed 9 Feb. 2023].

Liu, C.W., Lowe, R., Serban, I. v., Noseworthy, M., Charlin, L. and Pineau, J., (2016) How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation. *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.2122–2132. Available at: https://arxiv.org/abs/1603.08023v2 [Accessed 9 Feb. 2023].

Lin, C.-Y., (2004) ROUGE: A Package for Automatic Evaluation of Summaries. In *Text summarization branches out* (pp. 74-81). Available at: https://aclanthology.org/W04-1013 [Accessed 29 Oct. 2022].

Liu, J., Shen, D., Zhang, Y., Dolan, B., Carin, L. and Chen, W., (2021a) What Makes Good In-Context Examples for GPT-3? *DeeLIO 2022 - Deep Learning Inside Out: 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, Proceedings of the Workshop*, [online] pp.100–114. Available at: https://arxiv.org/abs/2101.06804v1 [Accessed 10 Feb. 2023].

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H. and Neubig, G., (2021b) Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Computing Surveys*, , 55(9), pp.1-35.. [online] Available at: https://arxiv.org/abs/2107.13586v1 [Accessed 10 Nov. 2022].

Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z. and Tang, J., (2021c) GPT Understands, Too. [online] Available at: https://arxiv.org/abs/2103.10385v1 [Accessed 10 Feb. 2023].

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M. and Zettlemoyer, L., (2020) Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, [online] 8, pp.726–742. Available at: https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00343/96484/Multilingual-Denoising-Pre-training-for-Neural [Accessed 10 Feb. 2023].

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V. and Allen, P.G., (2019) RoBERTa: A Robustly Optimized BERT Pretraining Approach. [online] Available at: https://arxiv.org/abs/1907.11692v1 [Accessed 10 Feb. 2023].

Liu, R. and Nikolic, P.K., (2021) Mutltimodal AI Companion for Interactive Fairytale Co-creation. [online] Available at: https://arxiv.org/abs/2112.00331v1 [Accessed 6 Sep. 2022].

Logan Iv, R.L., Balaževi´cbalaževi´c, I., Wallace, E., Petroni, F., Singh, S. and Riedel, S., (2021) Cutting Down on Prompts and Parameters: Simple Few-Shot Learning with Language Models. [online] pp.2824–2835. Available at: https://arxiv.org/abs/2106.13353v2 [Accessed 10 Feb. 2023].

Mark, G.W. and Clark, S., (2016) What Happens Next? Event Prediction Using a Compositional Neural Network Model. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 301, pp.2727–2733. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/10344 [Accessed 9 Feb. 2023].

Martin, L.J., Ammanabrolu, P., Wang, X., Hancock, W., Singh, S., Harrison, B. and Riedl, M.O., (2018) Event Representations for Automated Story Generation with Deep Neural Nets. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 321. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/11430 [Accessed 9 Feb. 2023].

McCann, B., Keskar, N.S., Xiong, C. and Socher, R., (2018) The Natural Language Decathlon: Multitask Learning as Question Answering. [online] Available at: https://arxiv.org/abs/1806.08730v1 [Accessed 10 Feb. 2023].

McCloskey, M. and Cohen, N.J., (1989) Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. *Psychology of Learning and Motivation - Advances in Research and Theory*, 24C, pp.109–165.

Mcintyre, N. and Lapata, M., (2010) Plot Induction and Evolutionary Search for Story Generation. [online] pp.11–16. Available at: http://opennlp.sourceforge.net/. [Accessed 9 Feb. 2023].

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P. and Allen, J., (2016a) A Corpus and Cloze Evaluation for Deeper Understanding of Commonsense Stories. pp.839–849.

Mostafazadeh, N., Vanderwende, L., Yih, W.-T., Kohli, P. and Allen, J., (2016b) Story Cloze Evaluator: Vector Space Representation Evaluation by Predicting What Happens Next. [online] pp.24–29. Available at: http://cs.rochester.edu/ [Accessed 9 Feb. 2023].

Muennighoff, N., Wang, T., Sutawika, L., Roberts, A., Biderman, S., Scao, T. Le, Bari, M.S., Shen, S., Yong, Z.-X., Schoelkopf, H., Tang, X., Radev, D., Aji, A.F., Almubarak, K., Albanie, S., Alyafeai, Z., Webson, A., Raff, E. and Raffel, C., (2022) Crosslingual Generalization through Multitask Finetuning. [online] Available at: https://arxiv.org/abs/2211.01786v1 [Accessed 16 Feb. 2023].

Och, F.J. and Ney, H., (2004) The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, [online] 304, pp.417–449. Available at: https://direct.mit.edu/coli/article/30/4/417/1869/The-Alignment-Template-Approach-to-Statistical [Accessed 12 Dec. 2022].

O'Neill, B. and Riedl, M., (2014) Dramatis: A Computational Model of Suspense. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 281, pp.944–950. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/8836 [Accessed 10 Feb. 2023].

OpenAI, (2023a) *ChatGPT: Optimizing Language Models for Dialogue*. [online] Available at: https://openai.com/blog/chatgpt/ [Accessed 17 Feb. 2023].

OpenAI, (2023b) *Model index for researchers - OpenAI API*. [online] Available at: https://platform.openai.com/docs/model-index-for-researchers [Accessed 16 Feb. 2023].

Openai, A.R., Openai, K.N., Openai, T.S. and Openai, I.S., (2018) Improving Language Understanding by Generative Pre-Training. [online] Available at: https://gluebenchmark.com/leaderboard [Accessed 12 Dec. 2022].

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D. and Auli, M., (2019) fairseq: A Fast, Extensible Toolkit for Sequence Modeling. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Demonstrations Session*, [online] pp.48–53. Available at: https://arxiv.org/abs/1904.01038v1 [Accessed 10 Feb. 2023].

Ouyang, X., Wang, S., Pang, C., Sun, Y., Tian, H., Wu, H. and Wang, H., (2020) ERNIE-M: Enhanced Multilingual Representation by Aligning Cross-lingual Semantics with Monolingual Corpora. *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.27–38. Available at: https://arxiv.org/abs/2012.15674v4 [Accessed 10 Feb. 2023].

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C.L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J. and Lowe, R., (2022) Training language models to follow instructions with human feedback. [online] Available at: https://arxiv.org/abs/2203.02155v1 [Accessed 16 Feb. 2023].

Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J., (2002) BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics* (pp. 311-318).

Peinado, F. and Gervás, P., (2006) Evaluation of automatic generation of basic stories. *New Generation Computing*, [online] 243, pp.289–302. Available at: https://link.springer.com/article/10.1007/BF03037336 [Accessed 10 Feb. 2023].

Perez, E., Kiela, D. and Cho, K., (2021) True Few-Shot Learning with Language Models. *Advances in Neural Information Processing Systems*, 34, pp.11054–11070.

PÉrez, R.Pé.Ý. and Sharples, M., (2010) MEXICA: A computer model of a cognitive account of creative writing. *http://dx.doi.org/10.1080/09528130010029820*, [online] 132, pp.119–139. Available at: https://www.tandfonline.com/doi/abs/10.1080/09528130010029820 [Accessed 10 Feb. 2023].

Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. and Zettlemoyer, L., (2018) Deep Contextualized Word Representations. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, [online] 1, pp.2227–2237. Available at: https://aclanthology.org/N18-1202 [Accessed 12 Dec. 2022].

Peters, M.E., Ruder, S. and Smith, N.A., (2019) To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks. *ACL 2019 - 4th Workshop on Representation Learning for NLP, RepL4NLP 2019 - Proceedings of the Workshop*, [online] pp.7–14. Available at: https://arxiv.org/abs/1903.05987v2 [Accessed 10 Feb. 2023].

Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A.H. and Riedel, S., (2019) Language Models as Knowledge Bases? *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.2463–2473. Available at: https://arxiv.org/abs/1909.01066v2 [Accessed 10 Feb. 2023].

Pichotta, K. and Mooney, R.J., (2014) Statistical Script Learning with Multi-Argument Events. pp.220–229.

Pichotta, K. and Mooney, R.J., (2016a) Learning Statistical Scripts with LSTM Recurrent Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 301, pp.2800–2806. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/10347 [Accessed 10 Feb. 2023].

Pichotta, K. and Mooney, R.J., (2016b) Using Sentence-Level LSTM Language Models for Script Inference. *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, [online] 1, pp.279–289. Available at: https://arxiv.org/abs/1604.02993v2 [Accessed 10 Feb. 2023].

Purdy, C., Wang, X., He, L. and Riedl, M., (2018) Predicting Generated Story Quality with Quantitative Measures. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (Vol. 14, No. 1, pp. 95-101). [online] Available at: https://dl.acm.org/doi/10.5555/3505378.3505392 [Accessed 13 Feb. 2023].

Qiu, X.P., Sun, T.X., Xu, Y.G., Shao, Y.F., Dai, N. and Huang, X.J., (2020) Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, [online] 6310, pp.1872–1897. Available at: https://link.springer.com/article/10.1007/s11431-020-1647-3 [Accessed 10 Feb. 2023].

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I., (2019) Language Models are Unsupervised Multitask Learners. OpenAI blog [online] , 1(8), p.9. Available at: https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf [Accessed 10 Feb. 2023].

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. and Liu, P.J., (2020) Exploring the limits of transfer learning with a unified text-to-text transformer. *The*

Journal of Machine Learning Research, [online] 21, pp.1–67. Available at: https://dl.acm.org/doi/10.5555/3455716.3455856 [Accessed 10 Feb. 2023].

Rashkin, H., Smith, E.M., Li, M. and Boureau, Y.L., (2018) Towards Empathetic Open-domain Conversation Models: a New Benchmark and Dataset. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, [online] pp.5370–5381. Available at: https://arxiv.org/abs/1811.00207v5 [Accessed 23 Oct. 2022].

Ratner, A.J., Ehrenberg, H.R., Hussain, Z., Dunnmon, J. and Ré, C., (2017) Learning to Compose Domain-Specific Transformations for Data Augmentation. *Advances in Neural Information Processing Systems*, 30.

Roemmele, M. and Gordon, A.S., (2015) Creative help: A story writing assistant. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, [online] 9445, pp.81–92. Available at: https://link.springer.com/chapter/10.1007/978-3-319-27036-4_8 [Accessed 10 Feb. 2023].

Roemmele, M., Gordon, A.S. and Swanson, R., (2017a) Evaluating Story Generation Systems Using Automated Linguistic Analyses. UNIVERSITY OF SOUTHERN CALIFORNIA LOS ANGELES.

Roemmele, M., Kobayashi, S., Inoue, N. and Gordon, A.M., (2017b) An RNN-based Binary Classifier for the Story Cloze Test. *LSDSem 2017 - 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-Level Semantics, Proceedings of the Workshop*, [online] pp.74–80. Available at: https://aclanthology.org/W17-0911 [Accessed 10 Feb. 2023].

Rose, S., Engel, D., Cramer, N. and Cowley, W., (2010) Automatic Keyword Extraction from Individual Documents. *Text Mining: Applications and Theory*, [online] pp.1–20. Available at: https://onlinelibrary.wiley.com/doi/full/10.1002/9780470689646.ch1 [Accessed 10 Feb. 2023].

Rudinger, R., Rastogi, P., Ferraro, F. and Durme, B. van, (2015) Script Induction as Language Modeling Center for Language and Speech Processing. pp.17–21.

Ryan, J., (2017) Grimes' Fairy Tales: A 1960s Story Generator. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, [online] 10690 LNCS, pp.89–103. Available at: https://link.springer.com/chapter/10.1007/978-3-319-71027-3_8 [Accessed 10 Feb. 2023].

Sagarkar, M., Wieting, J., Tu, L. and Gimpel, K., (2018) Quality Signals in Generated Stories. *NAACL HLT 2018 - Lexical and Computational Semantics, SEM 2018, Proceedings of the 7th Conference*, [online] pp.192–202. Available at: https://aclanthology.org/S18-2024 [Accessed 10 Feb. 2023].

Saito, I., Nishida, K., Nishida, K. and Tomita, J., (2020) Abstractive Summarization with Combination of Pre-trained Sequence-to-Sequence and Saliency Models. [online] Available at: https://arxiv.org/abs/2003.13028v1 [Accessed 10 Feb. 2023].

Saunshi, N., Malladi, S. and Arora, S., (2020) A Mathematical Exploration of Why Language Models Help Solve Downstream Tasks. [online] Available at: https://arxiv.org/abs/2010.03648v2 [Accessed 10 Feb. 2023].

Le Scao, T. and Rush, A.M., (2021) How Many Data Points is a Prompt Worth? *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, [online] pp.2627–2636. Available at: https://arxiv.org/abs/2103.08493v2 [Accessed 10 Feb. 2023].

Schick, T. and Schütze, H., (2020a) Exploiting Cloze Questions for Few Shot Text Classification and Natural Language Inference. *EACL 2021 - 16th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, [online] pp.255–269. Available at: https://arxiv.org/abs/2001.07676v3 [Accessed 10 Feb. 2023].

Schick, T. and Schütze, H., (2020b) Few-Shot Text Generation with Pattern-Exploiting Training. [online] Available at: https://arxiv.org/abs/2012.11926v2 [Accessed 10 Feb. 2023].

Schick, T. and Schütze, H., (2020c) It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners. *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, [online] pp.2339–2352. Available at: https://arxiv.org/abs/2009.07118v2 [Accessed 10 Feb. 2023].

Schick, T., Udupa, S. and Schütze, H., (2021) Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in nlp. *Transactions of the Association for Computational Linguistics*, [online] 9, pp.1408–1424. Available at: https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00434/108865/Self-Diagnosis-and-Self-Debiasing-A-Proposal-for [Accessed 10 Feb. 2023].

See, A., Pappu, A., Saxena, R., Yerukola, A. and Manning, C.D., (2019) Do Massively Pretrained Language Models Make Better Storytellers? *CoNLL 2019 - 23rd Conference on Computational Natural Language Learning, Proceedings of the Conference*, [online] pp.843–861. Available at: https://arxiv.org/abs/1909.10705v1 [Accessed 10 Feb. 2023].

Sennrich, R., Haddow, B. and Birch, A., (2016) Controlling Politeness in Neural Machine Translation via Side Constraints. [online] pp.35–40. Available at: http://www.opensubtitles.org [Accessed 10 Feb. 2023].

Shin, T., Razeghi, Y., Logan, R.L., Wallace, E. and Singh, S., (2020) AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.4222–4235. Available at: https://arxiv.org/abs/2010.15980v2 [Accessed 10 Feb. 2023].

Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J. and Catanzaro, B., (2019) Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism. [online] Available at: https://arxiv.org/abs/1909.08053v4 [Accessed 16 Feb. 2023].

Song, K., Tan, X., Qin, T., Lu, J. and Liu, T.Y., (2019) MASS: Masked Sequence to Sequence Pre-training for Language Generation. *36th International Conference on Machine Learning, ICML 2019*, [online] 2019-June, pp.10384–10394. Available at: https://arxiv.org/abs/1905.02450v5 [Accessed 10 Feb. 2023].

Sun, X., Fan, C., Sun, Z., Meng, Y., Wu, F. and Li, J., (2020) Summarize, Outline, and Elaborate: Long-Text Generation via Hierarchical Supervision from Extractive Summaries. [online] Available at: https://arxiv.org/abs/2010.07074v2 [Accessed 23 Oct. 2022].

Sun, Y., Wang, S., Feng, S., Ding, S., Pang, C., Shang, J., Liu, J., Chen, X., Zhao, Y., Lu, Y., Liu, W., Wu, Z., Gong, W., Liang, J., Shang, Z., Sun, P., Liu, W., Ouyang, X., Yu, D., Tian, H., Wu, H. and Wang, H., (2021) ERNIE 3.0: Large-scale Knowledge Enhanced Pre-training for Language Understanding and Generation. [online] Available at: https://arxiv.org/abs/2107.02137v1 [Accessed 10 Feb. 2023].

Sun, Y., Wang, S., Li, Y., Feng, S., Chen, X., Zhang, H., Tian, X., Zhu, D., Tian, H. and Wu, H., (2019) ERNIE: Enhanced Representation through Knowledge Integration. [online] Available at: https://arxiv.org/abs/1904.09223v1 [Accessed 10 Feb. 2023].

Tambwekar, P., Dhuliawala, M., Martin, L.J., Mehta, A., Harrison, B. and Riedl, M.O., (2018) Controllable Neural Story Plot Generation via Reward Shaping. *IJCAI International Joint Conference on Artificial Intelligence*, [online] 2019-August, pp.5982–5988. Available at: http://arxiv.org/abs/1809.10736 [Accessed 10 Feb. 2023].

Tan, B., Yang, Z., Al-Shedivat, M., Xing, E.P. and Hu, Z., (2020) Progressive Generation of Long Text with Pretrained Language Models. *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, [online] pp.4313–4324. Available at: https://arxiv.org/abs/2006.15720v2 [Accessed 23 Oct. 2022].

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., Li, Y., Lee, H., Zheng, H.S., Ghafouri, A., Menegali, M., Huang, Y., Krikun, M., Lepikhin, D., Qin, J., Chen, D., Xu, Y., Chen, Z., Roberts, A., Bosma, M., Zhao, V., Zhou, Y., Chang, C.-C., Krivokon, I., Rusch, W., Pickett, M., Srinivasan, P., Man, L., Meier-Hellstern, K., Morris, M.R., Doshi, T., Santos, R.D., Duke, T., Soraker, J., Zevenbergen, B., Prabhakaran, V., Diaz, M., Hutchinson, B., Olson, K., Molina, A., Hoffman-John, E., Lee, J., Aroyo, L., Rajakumar, R., Butryna, A., Lamm, M., Kuzmina, V., Fenton, J., Cohen, A., Bernstein, R., Kurzweil, R., Aguera-Arcas, B., Cui, C., Croak, M., Chi, E. and Le, Q., (2022) LaMDA: Language Models for Dialog Applications. [online] Available at: https://arxiv.org/abs/2201.08239v3 [Accessed 16 Feb. 2023].

Trinh, T.H., Brain, G. and Le Google Brain, Q. V, (2018) A Simple Method for Commonsense Reasoning. [online] Available at: https://arxiv.org/abs/1806.02847v2 [Accessed 10 Feb. 2023].

Tsimpoukelli, M., Menick, J.L., Cabi, S., Eslami, S.M.A., Vinyals, O. and Hill, F., (2021) Multimodal Few-Shot Learning with Frozen Language Models. *Advances in Neural Information Processing Systems*, 34, pp.200–212.

Vaswani, A., Brain, G., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I., (2017) Attention is All you Need. *Advances in Neural Information Processing Systems*, 30.

Wang, B., Liu, K. and Zhao, J., (2017a) Conditional Generative Adversarial Networks for Commonsense Machine Comprehension. In *IJCAI*. [online], (pp. 4123-4129) Available at: https://code.google.com/archive/p/word2vec/ [Accessed 10 Feb. 2023].

Wang, S., Durrett, G. and Erk, K., (2020) Narrative Interpolation for Generating and Understanding Stories. [online] Available at: https://arxiv.org/abs/2008.07466v1 [Accessed 10 Feb. 2023].

Wang, T., Chen, P. and Li, B., (2017b) Predicting the Quality of Short Narratives from Social Media. *IJCAI International Joint Conference on Artificial Intelligence*, [online] 0, pp.3859–3865. Available at: https://arxiv.org/abs/1707.02499v1 [Accessed 10 Feb. 2023].

Wang, T. and Wan, X., (2019) T-CVAE: Transformer-Based Conditioned Variational Autoencoder for Story Completion. [online] Available at: https://github.com/sodawater/T-CVAE. [Accessed 10 Feb. 2023].

Wang, Y., Mishra, S., Alipoormolabashi, P., Kordi, Y., Mirzaei, A., Arunkumar, A., Ashok, A., Dhanasekaran, A.S., Naik, A., Stap, D., Pathak, E., Karamanolakis, G., Lai, H.G., Purohit, I., Mondal, I., Anderson, J., Kuznia, K., Doshi, K., Patel, M., Pal, K.K., Moradshahi, M., Parmar, M., Purohit, M., Varshney, N., Kaza, P.R., Verma, P., Puri, R.S., Karia, R., Sampat, S.K., Doshi, S., Mishra, S., Reddy, S., Patro, S., Dixit, T., Shen, X., Baral, C., Choi, Y., Smith, N.A., Hajishirzi, H. and Khashabi, D., (2022) Super-NaturalInstructions: Generalization via Declarative Instructions on 1600+ NLP Tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. [online], (pp. 5085-5109) Available at: https://arxiv.org/abs/2204.07705v3 [Accessed 16 Feb. 2023].

Wang, Y., Yao, Q., Kwok, J.T. and Ni, L.M., (2020) Generalizing from a Few Examples. *ACM Computing Surveys (CSUR)*, [online] 533. Available at: https://dl.acm.org/doi/10.1145/3386252 [Accessed 10 Feb. 2023].

Wei, C., Xie, S.M. and Ma, T., (2021a) Why Do Pretrained Language Models Help in Downstream Tasks? An Analysis of Head and Prompt Tuning. *Advances in Neural Information Processing Systems*, 34, pp.16158–16170.

Wei, J., Bosma, M., Zhao, V.Y., Guu, K., Yu, A.W., Lester, B., Du, N., Dai, A.M. and Le, Q. V., (2021b) Finetuned Language Models Are Zero-Shot Learners. [online] Available at: https://arxiv.org/abs/2109.01652v5 [Accessed 16 Feb. 2023].

Wilensky, R., (1983) Story grammars versus story points. *Behavioral and Brain Sciences*, [online] 64, pp.579–591. Available at: https://www.cambridge.org/core/journals/behavioral-and-brain-sciences/article/abs/story-grammars-versus-story-points/5DA9B2DFF9F8811942256C97230304B9 [Accessed 10 Feb. 2023].

Xu, J., Ren, X., Zhang, Y., Zeng, Q., Cai, X. and Sun, X., (2018) A Skeleton-Based Model for Promoting Coherence Among Sentences in Narrative Story Generation. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, [online] pp.4306–4315. Available at: https://arxiv.org/abs/1808.06945v2 [Accessed 10 Feb. 2023].

Xu, P., Patwary, M., Shoeybi, M., Puri, R., Fung, P., Anandkumar, A. and Catanzaro, B., (2020) MEGATRON-CNTRL: Controllable Story Generation with External Knowledge Using Large-Scale Language Models. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.2831–2845. Available at: https://arxiv.org/abs/2010.00840v1 [Accessed 10 Feb. 2023].

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R. and Le, Q. v., (2019) XLNet: Generalized Autoregressive Pretraining for Language Understanding. *Advances in Neural Information Processing Systems*, [online] 32. Available at: https://github.com/zihangdai/xlnet [Accessed 12 Dec. 2022].

Yao, L., Peng, N., Weischedel, R., Knight, K., Zhao, D. and Yan, R., (2019) Plan-and-Write: Towards Better Automatic Storytelling. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 3301, pp.7378–7385. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/4726 [Accessed 10 Feb. 2023].

Yin, W., Hay, J. and Roth, D., (2019) Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.3914–3923. Available at: https://arxiv.org/abs/1909.00161v1 [Accessed 10 Feb. 2023].

Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., Choi, Y. and Allen, P.G., (2019) Defending Against Neural Fake News. *Advances in Neural Information Processing Systems*, [online] 32. Available at: https://rowanzellers.com/grover [Accessed 23 Oct. 2022].

Zhai, F., Demberg, V., Shkadzko, P., Shi, W. and Sayeed, A., (2019) A Hybrid Model for Globally Coherent Story Generation. [online] pp.34–45. Available at: https://aclanthology.org/W19-3404 [Accessed 10 Feb. 2023].

Zhang, J., Zhao, Y., Saleh, M. and Liu, P.J., (2020) PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. In *International Conference on Machine Learning* (pp. 11328-11339). PMLR. Available at: https://proceedings.mlr.press/v119/zhang20ae.html [Accessed 12 Dec. 2022].

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X.V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P.S., Sridhar, A., Wang, T. and Zettlemoyer, L., (2022) OPT: Open Pre-trained Transformer Language Models. [online] Available at: https://arxiv.org/abs/2205.01068v4 [Accessed 16 Feb. 2023].

Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q. and Artzi, Y., (2019a) BERTScore: Evaluating Text Generation with BERT. [online] Available at: https://arxiv.org/abs/1904.09675v3 [Accessed 16 Feb. 2023].

Zhang, Y. and Nivre, J., (2011) Transition-based Dependency Parsing with Rich Non-local Features. [online] pp.188–193. Available at: http://w3.msi.vxu.se/ [Accessed 12 Dec. 2022].

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J. and Dolan, B., (2019b) DialoGPT: Large-Scale Generative Pre-training for Conversational Response Generation. [online] pp.270–278. Available at: https://arxiv.org/abs/1911.00536v3 [Accessed 23 Oct. 2022].

Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M. and Liu, Q., (2019c) ERNIE: Enhanced Language Representation with Informative Entities. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, [online] pp.1441–1451. Available at: https://arxiv.org/abs/1905.07129v3 [Accessed 10 Feb. 2023].

Zhao, T.Z., Wallace, E., Feng, S., Klein, D. and Singh, S., (2021) Calibrate Before Use: Improving Few-shot Performance of Language Models. In *International Conference on Machine Learning* (pp. 12697-12706). PMLR. Available at: https://proceedings.mlr.press/v139/zhao21c.html [Accessed 10 Feb. 2023].

Zhu, C., Hinthorn, W., Xu, R., Zeng, Q., Zeng, M., Huang, X. and Jiang, M., (2020) Enhancing Factual Consistency of Abstractive Summarization. *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, [online] pp.718–733. Available at: https://arxiv.org/abs/2003.08612v8 [Accessed 10 Feb. 2023].

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A. and Fidler, S., (2015) Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. In *Proceedings of the IEEE international conference on computer vision* (pp. 19-27).

Zweig, G., Platt, J.C., Meek, C., Burges, C.J.C., Yessenalina, A. and Liu, Q., (2012) Computational Approaches to Sentence Completion. pp.8–14.

APPENDIX A: RESEARCH PROPOSAL

Generating Stories by Prompting Pre-trained Language Models

Paresh Pradhan

Research Proposal for
Master of Science in Machine Learning & Artificial Intelligence

Liverpool John Moores University & upGrad

November 2022

Abstract

Text generation problems have seen remarkable advancements with the advent of pre-trained language models (PLMs). These models can only influence a few broad characteristics of the output text, though. PLMs are unable to produce long-form stories as they are ignorant of the narrative structure. Recent studies on tale generation have made use of explicit content planning, which can result in stories with more logical event sequences. However, a shortage of training data makes it challenging to fine-tune PLMs. It is difficult to achieve precise control, even with fine-tuning. Therefore, building a model that can produce long-form stories is not simple. Recent prompt-based learning provides a potential answer for this problem. This thesis work proposes a method to use prompt-based learning to generate stories while maintaining fine-grained control.

1. Background

In recent years, with the emergence of large pre-trained language models, the quality of machine-generated text has improved significantly (Rashkin et al., 2018; Radford et al., 2019; Zhang et al., 2019; Brown et al., 2020; Guan et al., 2020; Bakhtin et al., 2021). Today, models can generate text that is indistinguishable from human-written text (Clark et al., 2021).

Controlling the generation is still a challenge however, despite the fact that large-scale PLMs have demonstrated excellent capabilities in producing coherent and intelligible text (Keskar et al., 2019; Radford et al., 2019; Zellers et al., 2019). A more thorough examination of generated text reveals problems like topic drift and self-contradiction(Fan et al., 2019; Bisk et al., 2020; Gao et al., 2020a; Tan et al., 2020; Dou et al., 2021; Dziri et al., 2021). These flaws stand out, in particular, for open-ended text generation tasks that require a high level of coherence, such as story generation.

Stories generated using language models have shown to lack discourse coherence (Bosselut et al., 2018; Ji and Huang, 2021), global planning (Hua and Wang, 2020; Tan et al., 2020) and common-sense knowledge (Ji et al., 2020; Xu et al., 2020). While the individual sentences in a generated text seem logical and fluent, when put together, the overall story often does not make much sense (See et al., 2019; Goldfarb-Tarrant et al., 2020). In long-form text generation, sentences tend to repeat which leads to reduction in story quality (Yao et al., 2019).

To provide structure to the generation process, recent works have tried to used explicit content planning. The content plan comes in different forms. (Fan et al., 2018a)used prompts. (Xu et al., 2018; Yao et al., 2019) used keywords and key-phrases. (Fan et al., 2019) used semantic frames. (Sun et al., 2020) used summaries. To make use of these content plans, PLMs generally require fine-tuning on content-plan related data. A challenge with fine-tuning PLMs is that, in addition to needing training data, the model has a tendency to learn frequently occurring events from the content plan and derives common sense information from them (Fan et al., 2019). This leads to lack of variety in generated stories.

Another novel approach to address linguistic issues without fine-tuning is provided by the recently proposed prompt-based learning (Liu et al., 2021). In this framework, task-specific prompts can be used to address text-based problems. Researchers have shown that using

prompts, PLMs can solve existing or new generation tasks without need for fine-tuning (Brown et al., 2020; Li and Liang, 2021).

Although prompt-based learning looks promising, there are still some challenges. Prompts are highly task-specific and are hard to transfer or reuse for new tasks (Gao et al., 2020b). Even for the same task the prompts may not work well for all instances in a large population (le Scao and Rush, 2021).

2. Related Work

Controllable story generation has been studied from different angles. Researchers have focused on controlling story generation using broad thematic elements such as sentiment, genre, style, topic, etc. (Hu et al., 2017; Shen et al., 2017; Zhao et al., 2018; Dathathri et al., 2019; Fang et al., 2019; Keskar et al., 2019). Some works have tried more fine-grained control using story-lines, story-plans and plots (Peng et al., 2018; Yao et al., 2019). These works were benchmarked using relatively short-text datasets such as the five-lines story dataset, ROCStories(Mostafazadeh et al., 2016). Later on, some works have tried to controllable story generation with long-form text (Fan et al., 2018a, 2019; Rashkin et al., 2020; Fang et al., 2021).

Similar to short story generation, researchers have tried using fine-grained control to drive long-form story generation as well. (Fang et al., 2021) proposed generation of story given an outline of story events/phrases. (Rashkin et al., 2020) presented a comparable technique with a specialized architecture and memory mechanism. (Sun et al., 2020) created an outline of the story by generating summaries for each segment of the story. Then each summary is extrapolated to generate the full story.

Most of the research in the field is based on fine-tuning transformer-based Pre-trained Language Models (PLM) (Vaswani et al., 2017; Devlin et al., 2018; Radford et al., 2019) with curated or generated datasets (Conneau and Lample, 2019; Dong et al., 2019; Keskar et al., 2019; Song et al., 2019). Because of its specialized architecture for unconditional text generation, GPT2 (Radford et al., 2019), in particular, has attracted much attention in this field (Mao et al., 2019; See et al., 2019; Ziegler et al., 2019). And lately, after the availability of its API, GPT3 (Brown

et al., 2020) has seen increasing usage for text generation (Dou et al., 2021; Shakeri et al., 2021) .

In the absence of sufficient data, fine-tuning PLMs is challenging (Chen et al., 2019; Li et al., 2021). To resolve that, researchers have tried Plug-and-Play methods to control story generation without fine-tuning (Dathathri et al., 2019; Pascual et al., 2020, 2021; Lin and Riedl, 2021; Jin et al., 2022; Mori et al., 2022).

Prompt-based learning is another approach that does not require fine-tuning. Some works have used hand-crafted prompts for various generation tasks (Brown et al., 2020; Raffel et al., 2020; Zou et al., 2021). Others have tried to automatically generate discrete prompts (Gao et al., 2020b; Shin et al., 2020) and continuous prompts (Li and Liang, 2021; Liu et al., 2021). Some have tried to generate prompts for target task using source task (Su et al., 2021; Vu et al., 2021).

3. Research Questions

This thesis tries to answer the following questions:

1. The approaches for story generation with fine-grained control require fine-tuning of PLMs. Can these approaches be used with Prompt-based learning to generate stories in a Few-Shot manner without fine-tuning?
2. The previous methods largely use GPT2 as base model. Can using the latest generation GPT3 (or alternatives) improve the text generation capabilities?
3. Prompt-based learning has been used to generate text in few-shot manner. Can this be extended to story generation task?

4. Aim and Objectives

This work tries to explore the Few-shot capabilities of GPT3 for long-form controllable story generation task.

Objectives:

- To conduct a comprehensive review of available literature with regards to Long-form story generation, Prompt-learning and Few-Shot text generation.
- To explore the viability and then develop a method to generate short and long form stories using few-shot generation and prompting.
- To evaluate the generated stories using automated story generation evaluation metrics and compare the developed method against existing methods.

5. Significance of the Study

Story Generation is a field under active research. While short-form story generation has been studied extensively, long-form story generation is relatively under-explored. Although fine-tuning based approaches have been used in previous works, there is a lack of research in generating stories without fine-tuning.

This work tries to fill these gaps by adding to the existing literature, providing benchmarks and contributing code. This work also explores recent developments in Prompt-based learning and Few-Shot generation.

In terms of application, this work helps story writers write better stories in conjunction with AI. This can help writers get new ideas or get over the writer's block.

6. Scope of the Study

The scope of this thesis work is defined as follows:

- The thesis work is to be completed within 17 weeks after submission of research proposal.
- The experimentation will be conducted using open-source software and models.
- The experimentation will be conducted using publicly available GPU such as Google-Colab.

- Human evaluation of the generated story is not a part of this thesis work. The evaluation will only focus on automated metrics.

7. Research Methodology

This work focuses on the text generation and few-shot learning capabilities of PLMs. Given an outline as a control-mechanism, the model should generate a story conditioned on the outline.

7.1 Dataset Description

This work makes use of two standard story generation datasets:

- **ROCStories**: Introduced by (Mostafazadeh et al., 2016), this dataset contains ~98K 5-sentence long stories along with story titles. This dataset is widely used for short-form story generation tasks.
- **WritingPrompts**: Introduced by (Fan et al., 2018b), this dataset contains ~300K human-written stories along with the starting prompt used to write the story. These stories were collected from the Reddit, an online social media forum. These stories are long-form multi-paragraph stories, and hence useful for more complex task of long-form story generation.

7.2 Data Preparation

The proposed method requires sample pairs of outline-instance to paragraph. While paragraph text can be derived from the ROCStories and WritingPrompts datasets, there is no dataset of outlines readily available. Hence, the outlines need to extracted from the story datasets and then mapped to corresponding paragraph text. These outline-paragraph pairs can then be sampled during the few-shot inference.

The outline instances can take one of two forms:

- **Summary** – Here the outline instance is a short extractive summary of the paragraph. The paragraph is expanded from the summary. For the summary extraction, TextRank(Mihalcea and Tarau, 2004) is proposed to be used to extract the most informative sentence from the paragraph.

- **Keywords/Keyphrases** - Here the outline instance is a set of keywords and phrases that are present in the paragraph. The paragraph text is generated conditioned on these keywords/keyphrases. For the outline extraction, RAKE (Rose et al., 2010) is proposed to be used to extract keyphrases from the paragraph.

7.3 Algorithms & Techniques Description

7.3.1 Pre-trained Language Models (PLMs)

Pre-trained models originate from the idea of transfer learning. Transfer learning refers to the process of applying previously acquired knowledge to new tasks. Traditional transfer learning used large volume of annotated data points for supervised training. Pre-training with self-supervised learning on vast amounts of unlabelled data has emerged as the most popular transfer learning strategy in deep learning. Pre-training methods differ from other approaches in that they use unlabelled data for self-supervised training and can be used for a number of downstream tasks using fine-tuning or few-shot learning.

Language modelling, in NLP, refers to the task of predicting the next character/word/sentence in a text. Language models are trained in a self-supervised manner using large corpora of unstructured text. These models can then be used for a number of natural language tasks, such as question answering, text generation, and text classification.

Pre-trained language models combine the tasks of language modelling and transfer learning leading to the creation of large language models which can be fine-tuned for many downstream tasks. Some of the most well-known language models are:

- BERT (Devlin et al., 2018)
- GPT3 (Brown et al., 2020)

7.3.2 Few-Shot Learning (FSL)

With the aid of a small number of samples and previously acquired knowledge, humans can quickly recognise new classes in data. This is called meta-learning. Few-Shot Learning is a type of meta-learning. In this method, to efficiently generalise to new (but related) tasks with a small number of instances during the meta-testing phase, a learner is taught on a number of related tasks during the meta-training phase. One effective approach towards solving Few-Shot Learning problems is to learn a common representation for numerous tasks and then train task-specific classifiers on top of this representation. FSL is a solution to the problem of traditional supervised learning methods requiring large quantities of labelled data for training.

7.3.3 Prompt-Learning

Prompt-based learning is a new class of techniques for training ML models. When prompting, users directly state in natural language the task they want the pre-trained language model to understand and complete. In contrast, conventional Transformer training first pre-trained models using unlabelled data before fine-tuning them using labelled data for the desired downstream task. A prompt is basically a user-written natural language instruction that the model is supposed to follow. There may be a need for multiple prompts, depending on how difficult the task is that is being trained for. Prompt engineering is the process of selecting the appropriate prompt, or series of prompts, for the required task. Compared to the conventional pre-train & fine-tune method, prompt-based learning has many benefits. The primary benefit is that prompting typically performs quite well with few samples of labelled data.

7.4 Implementation

The proposed implementation can be broadly separated into two major steps:

1. Create prompts for Few-Shot Learning – In this step, a dataset of few-shot sample pairs is created. Each sample pair consists of an outline (o) and corresponding text paragraph (t). The dataset takes the following form:

$[(o_1, t_1), (o_2, t_2), (o_3, t_3), \ldots, (o_n, t_n)]$

2. Use the sample pairs as few-shot prompts to generate missing story paragraph for a new outline – The prompt, few-shot samples and the query outline are passed to the model as input for inference. The model returns the generated story paragraph corresponding to the query outline as prediction.

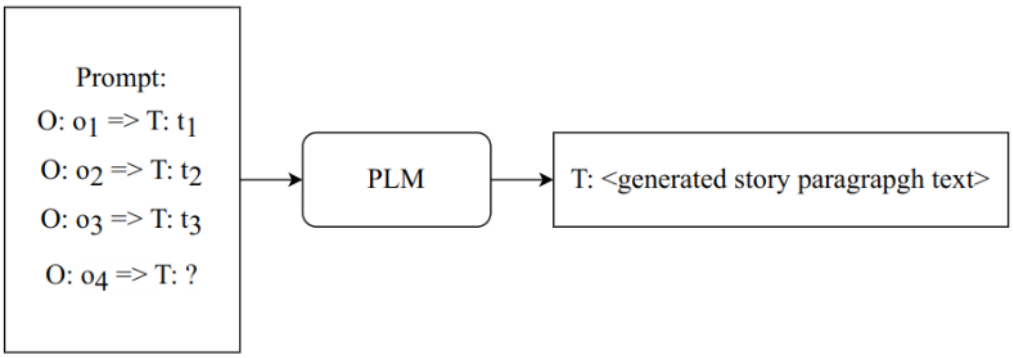

Figure 7.4.1

7.5 Evaluation

The generated stories are to be evaluated using multiple metrics. This work only focuses on evaluation using Automatic Metrics. Human-Evaluation of the generated stories is not within the scope of this work.

The proposed metrics for evaluation are:

- **Perplexity (PPL)** - Similar to(Fang et al., 2021; Jin et al., 2022), PPL is used to compute word-level complexity.
- **DIST/distinct-n**(Li et al., 2015) - DIST measures generation diversity as a ratio of distinct n-grams to all generated n-grams.
- **BLEU**(Papineni et al., 2002) - Measures n-gram overlap between generated text and ground truth.

- **Self-BLEU**(Zhu et al., 2018) - Measures intra-story lexical diversity.
- **ROUGE**(Lin, 2004) - Includes Precision, Recall & F1, where ROUGE Precision has similar interpretation as BLEU score.

This work will be benchmarked against the following baselines:

- Outline-to-Story (**O2S**) (Fang et al., 2021)
- Summarize, Outline and Elaborate (**SOE**) (Sun et al., 2020)
- Prompt Transfer for Text Generation (**PTG**) (Li et al., 2022)

8. Required Resources

8.1 Hardware Requirements

The following hardware requirements must be met for this research work:

- A laptop/desktop computer with internet access capable of browsing, doc-writing and compiling/executing code.
- Access to GPUs to execute CUDA-based deep-learning model training/inference.

8.2 Software Requirements

The following software requirements must be met for this research work:

- Web-browser
- Code IDE
- Python 3.7+
- NVIDIA - CUDA libraries
- Deep Learning libraries such as TensorFlow, PyTorch and HuggingFace

- Other python libraries required for working with data, e.g., Pandas, Numpy, NLTK, etc.

8.3 Dataset Requirements

The following dataset requirements must be met for this research work:

- ROCStories dataset requires a form to be filled and the dataset links are sent via email (ROCStories and the Story Cloze Test, 2022).

9. Research Plan

9.1 Gantt Chart

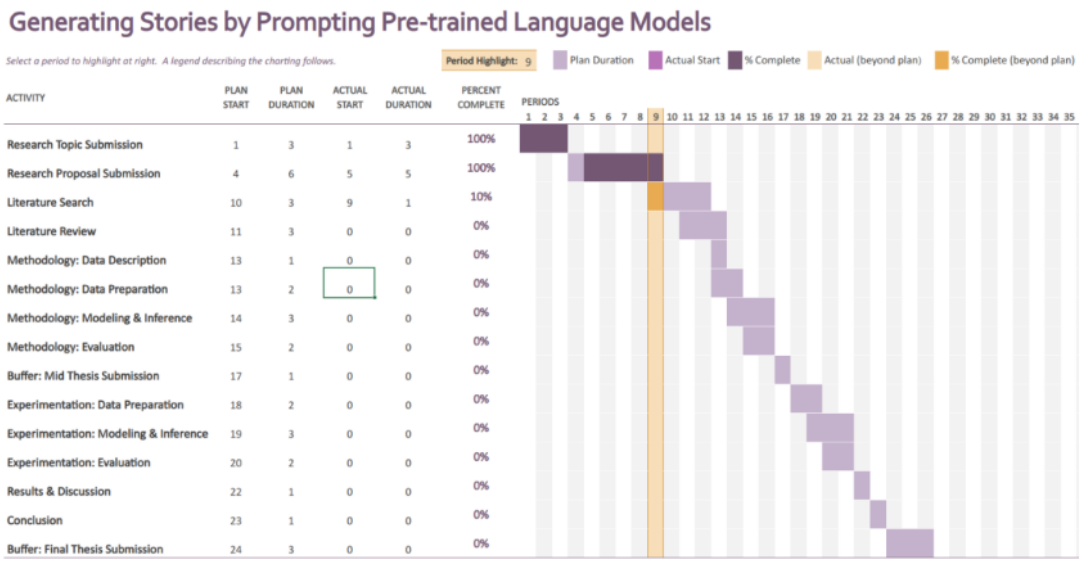

Figure 9.1.1

Note: 1 Period = I Calendar Week

9.2 Risk Mitigation and Contingency Plan

The potential risks to the completion of the thesis work and corresponding contingencies are listed below:

Table 9.2.1

Risk	**Contingency**
Candidate is unable to perform research work due to health issues or personal problems and it affects timelines.	Plan for buffer time in project management. Inform University/Upgrad administration and ask for extension.
Unavailability of specialized hardware such as GPUs.	Use cloud GPUs.

References

Anon (2022) *ROCStories and the Story Cloze Test*. [online] Available at: https://cs.rochester.edu/nlp/rocstories/ [Accessed 26 Oct. 2022].

Bakhtin, A., Deng, Y., Ott, M., Ranzato, M.' A. and Szlam, A., (2021) Residual Energy-Based Models for Text. *Journal of Machine Learning Research*, [online] 22, pp.1–41. Available at: http://jmlr.org/papers/v22/20-326.html. [Accessed 23 Oct. 2022].

Bisk, Y., Holtzman, A., Thomason, J., Andreas, J., Bengio, Y., Chai, J., Lapata, M., Lazaridou, A., May, J., Nisnevich, A., Pinto, N. and Turian, J., (2020) Experience Grounds Language. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.8718–8735. Available at: https://arxiv.org/abs/2004.10151v3 [Accessed 23 Oct. 2022].

Bosselut, A., Celikyilmaz, A., He, X., Gao, J., Huang, P. sen and Choi, Y., (2018) Discourse-Aware Neural Rewards for Coherent Text Generation. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, [online] 1, pp.173–184. Available at: https://arxiv.org/abs/1805.03766v1 [Accessed 23 Oct. 2022].

Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., Mccandlish, S., Radford, A., Sutskever, I. and Amodei, D., (2020) Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems*, [online] 33, pp.1877–1901. Available at: https://commoncrawl.org/the-data/ [Accessed 23 Oct. 2022].

Chen, Z., Eavani, H., Chen, W., Liu, Y. and Wang, W.Y., (2019) Few-Shot NLG with Pre-Trained Language Model. [online] pp.183–190. Available at: https://arxiv.org/abs/1904.09521v3 [Accessed 25 Oct. 2022].

Clark, E., August, T., Serrano, S., Haduong, N., Gururangan, S. and Smith, N.A., (2021) All That's 'Human' Is Not Gold: Evaluating Human Evaluation of Generated Text. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.7282–7296. Available at: https://arxiv.org/abs/2107.00061v2 [Accessed 23 Oct. 2022].

Conneau, A. and Lample, G., (2019) Cross-lingual Language Model Pretraining. *Advances in Neural Information Processing Systems*, [online] 32. Available at: https://arxiv.org/abs/1901.07291v1 [Accessed 25 Oct. 2022].

Dathathri, S., Madotto, A., Lan, J., Hung, J., Ai, U., Frank, E., Molino, P., Yosinski, J. and Liu, R., (2019) Plug and Play Language Models: A Simple Approach to Controlled Text Generation. [online] Available at: https://arxiv.org/abs/1912.02164v4 [Accessed 25 Oct. 2022].

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., (2018) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL HLT 2019 - 2019 Conference*

of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference, [online] 1, pp.4171–4186. Available at: https://arxiv.org/abs/1810.04805v2 [Accessed 23 Oct. 2022].

Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M. and Hon, H.-W., (2019) Unified Language Model Pre-training for Natural Language Understanding and Generation. *Advances in Neural Information Processing Systems*, [online] 32. Available at: https://github.com/microsoft/unilm. [Accessed 25 Oct. 2022].

Dou, Y., Forbes, M., Koncel-Kedziorski, R., Smith, N.A. and Choi, Y., (2021) Is GPT-3 Text Indistinguishable from Human Text? Scarecrow: A Framework for Scrutinizing Machine Text. [online] pp.7250–7274. Available at: https://arxiv.org/abs/2107.01294v3 [Accessed 23 Oct. 2022].

Dziri, N., Madotto, A., Zaiane, O. and Bose, A.J., (2021) Neural Path Hunter: Reducing Hallucination in Dialogue Systems via Path Grounding. *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.2197–2214. Available at: https://arxiv.org/abs/2104.08455v2 [Accessed 23 Oct. 2022].

Fan, A., Lewis, M. and Dauphin, Y., (2018a) Hierarchical Neural Story Generation. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, [online] 1, pp.889–898. Available at: https://arxiv.org/abs/1805.04833v1 [Accessed 23 Oct. 2022].

Fan, A., Lewis, M. and Dauphin, Y., (2018b) Hierarchical Neural Story Generation. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, [online] 1, pp.889–898. Available at: https://aclanthology.org/P18-1082 [Accessed 13 Sep. 2022].

Fan, A., Lewis, M. and Dauphin, Y., (2019) Strategies for Structuring Story Generation. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, [online] pp.2650–2660. Available at: https://arxiv.org/abs/1902.01109v2 [Accessed 23 Oct. 2022].

Fang, L., Li, C., Gao, J., Dong, W. and Chen, C., (2019) Implicit Deep Latent Variable Models for Text Generation. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.3946–3956. Available at: https://arxiv.org/abs/1908.11527v3 [Accessed 25 Oct. 2022].

Fang, L., Zeng, T., Liu, C., Bo, L., Dong, W. and Chen, C., (2021) Outline to Story: Fine-grained Controllable Story Generation from Cascaded Events. [online] Available at: http://arxiv.org/abs/2101.00822 [Accessed 31 Aug. 2022].

Gao, J., Peng, B., Li, C., Li, J., Shayandeh, S., Liden, L. and Shum, H.-Y., (2020a) Robust Conversational AI with Grounded Text Generation. [online] Available at: https://arxiv.org/abs/2009.03457v1 [Accessed 23 Oct. 2022].

Gao, T., Fisch, A. and Chen, D., (2020b) Making Pre-trained Language Models Better Few-shot Learners. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing,*

Proceedings of the Conference, [online] pp.3816–3830. Available at: https://arxiv.org/abs/2012.15723v2 [Accessed 25 Oct. 2022].

Goldfarb-Tarrant, S., Chakrabarty, T., Weischedel, R. and Peng, N., (2020) Content Planning for Neural Story Generation with Aristotelian Rescoring. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.4319–4338. Available at: https://arxiv.org/abs/2009.09870v2 [Accessed 23 Oct. 2022].

Guan, J., Huang, F., Zhao, Z., Zhu, X. and Huang, M., (2020) A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, [online] 8, pp.93–108. Available at: https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00302/43540/A-Knowledge-Enhanced-Pretraining-Model-for [Accessed 23 Oct. 2022].

Hu, Z., Yang, Z., Liang, X., Salakhutdinov, R. and Xing, E.P., (2017) *Toward Controlled Generation of Text*. Available at: https://proceedings.mlr.press/v70/hu17e.html [Accessed 25 Oct. 2022].

Hua, X. and Wang, L., (2020) PAIR: Planning and Iterative Refinement in Pre-trained Transformers for Long Text Generation. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.781–793. Available at: https://arxiv.org/abs/2010.02301v1 [Accessed 23 Oct. 2022].

Ji, H. and Huang, M., (2021) DiscoDVT: Generating Long Text with Discourse-Aware Discrete Variational Transformer. *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, [online] pp.4208–4224. Available at: https://arxiv.org/abs/2110.05999v1 [Accessed 23 Oct. 2022].

Ji, H., Ke, P., Huang, S., Wei, F., Zhu, X. and Huang, M., (2020) Language Generation with Multi-Hop Reasoning on Commonsense Knowledge Graph. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.725–736. Available at: https://arxiv.org/abs/2009.11692v1 [Accessed 23 Oct. 2022].

Jin, Y., Kadam, V. and Wanvarie, D., (2022) Plot Writing From Pre-Trained Language Models. *INLG 2022*. [online] Available at: https://arxiv.org/abs/2206.03021 [Accessed 24 Aug. 2022].

Keskar, N.S., Mccann, B., Varshney, L.R., Xiong, C., Socher, R. and Research, S., (2019) CTRL: A Conditional Transformer Language Model for Controllable Generation. [online] Available at: https://arxiv.org/abs/1909.05858v2 [Accessed 23 Oct. 2022].

Li, J., Galley, M., Brockett, C., Gao, J. and Dolan, B., (2015) A Diversity-Promoting Objective Function for Neural Conversation Models. *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference*, [online] pp.110–119. Available at: https://arxiv.org/abs/1510.03055v3 [Accessed 29 Oct. 2022].

Li, J., Tang, T., Nie, J.-Y., Wen, J.-R. and Zhao, W.X., (2022) Learning to Transfer Prompts for Text Generation. *NAACL 2022*, [online] pp.3506–3518. Available at: https://arxiv.org/abs/2205.01543v2 [Accessed 11 Oct. 2022].

Li, J., Tang, T., Zhao, W.X., Wei, Z., Yuan, N.J. and Wen, J.R., (2021) Few-shot Knowledge Graph-to-Text Generation with Pretrained Language Models. *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, [online] pp.1558–1568. Available at: https://arxiv.org/abs/2106.01623v1 [Accessed 25 Oct. 2022].

Li, X.L. and Liang, P., (2021) Prefix-Tuning: Optimizing Continuous Prompts for Generation. *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.4582–4597. Available at: https://arxiv.org/abs/2101.00190v1 [Accessed 25 Oct. 2022].

Lin, C.-Y., (2004) *ROUGE: A Package for Automatic Evaluation of Summaries*. Available at: https://aclanthology.org/W04-1013 [Accessed 29 Oct. 2022].

Lin, Z. and Riedl, M.O., (2021) Plug-and-Blend: A Framework for Controllable Story Generation with Blended Control Codes. *NAACL | NUSE | 2021*, [online] pp.62–71. Available at: https://aclanthology.org/2021.nuse-1.7 [Accessed 31 Aug. 2022].

Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z. and Tang, J., (2021) GPT Understands, Too. [online] Available at: https://arxiv.org/abs/2103.10385v1 [Accessed 25 Oct. 2022].

Mao, H.H., Majumder, B.P., McAuley, J. and Cottrell, G.W., (2019) Improving Neural Story Generation by Targeted Common Sense Grounding. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, [online] pp.5988–5993. Available at: https://aclanthology.org/D19-1615 [Accessed 31 Aug. 2022].

Mihalcea, R. and Tarau, P., (2004) TextRank: Bringing Order into Texts.

Mori, Y., Yamane, H., Shimizu, R. and Harada, T., (2022) Plug-and-Play Controller for Story Completion: A Pilot Study toward Emotion-aware Story Writing Assistance. [online] pp.46–57. Available at: https://aclanthology.org/2022.in2writing-1.6 [Accessed 1 Oct. 2022].

Mostafazadeh, N., Chambers, N., He, X., Parikh, D., Batra, D., Vanderwende, L., Kohli, P. and Allen, J., (2016) A Corpus and Cloze Evaluation for Deeper Understanding of Commonsense Stories. *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference*, [online] pp.839–849. Available at: https://aclanthology.org/N16-1098 [Accessed 13 Sep. 2022].

Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J., (2002) Bleu: a Method for Automatic Evaluation of Machine Translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, [online] pp.311–318. Available at: https://aclanthology.org/P02-1040 [Accessed 29 Oct. 2022].

Pascual, D., Egressy, B., Bolli, F. and Wattenhofer, R., (2020) Directed Beam Search: Plug-and-Play Lexically Constrained Language Generation. [online] Available at: http://arxiv.org/abs/2012.15416 [Accessed 31 Aug. 2022].

Pascual, D., Egressy, B., Meister, C., Cotterell, R. and Wattenhofer, R., (2021) A Plug-and-Play Method for Controlled Text Generation. *Findings of the Association for Computational*

Linguistics, Findings of ACL: EMNLP 2021, [online] pp.3973–3997. Available at: https://aclanthology.org/2021.findings-emnlp.334 [Accessed 31 Aug. 2022].

Peng, N., Ghazvininejad, M., May, J. and Knight, K., (2018) Towards Controllable Story Generation. [online] pp.43–49. Available at: https://aclanthology.org/W18-1505 [Accessed 25 Oct. 2022].

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I., (2019) Language Models are Unsupervised Multitask Learners. [online] Available at: https://github.com/codelucas/newspaper [Accessed 23 Oct. 2022].

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. and Liu, P.J., (2020) Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, [online] 21, pp.1–67. Available at: http://jmlr.org/papers/v21/20-074.html. [Accessed 25 Oct. 2022].

Rashkin, H., Celikyilmaz, A., Choi, Y. and Gao, J., (2020) PlotMachines: Outline-Conditioned Generation with Dynamic Plot State Tracking. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.4274–4295. Available at: https://aclanthology.org/2020.emnlp-main.349 [Accessed 31 Aug. 2022].

Rashkin, H., Smith, E.M., Li, M. and Boureau, Y.L., (2018) Towards Empathetic Open-domain Conversation Models: a New Benchmark and Dataset. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, [online] pp.5370–5381. Available at: https://arxiv.org/abs/1811.00207v5 [Accessed 23 Oct. 2022].

Rose, S., Engel, D., Cramer, N. and Cowley, W., (2010) Automatic Keyword Extraction from Individual Documents. *Text Mining: Applications and Theory*, [online] pp.1–20. Available at: https://onlinelibrary.wiley.com/doi/full/10.1002/9780470689646.ch1 [Accessed 29 Oct. 2022].

le Scao, T. and Rush, A.M., (2021) How Many Data Points is a Prompt Worth? *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, [online] pp.2627–2636. Available at: https://arxiv.org/abs/2103.08493v2 [Accessed 25 Oct. 2022].

See, A., Pappu, A., Saxena, R., Yerukola, A. and Manning, C.D., (2019) Do Massively Pretrained Language Models Make Better Storytellers? *CoNLL 2019 - 23rd Conference on Computational Natural Language Learning, Proceedings of the Conference*, [online] pp.843–861. Available at: https://arxiv.org/abs/1909.10705v1 [Accessed 23 Oct. 2022].

Shakeri, H., Neustaedter, C. and DiPaola, S., (2021) SAGA: Collaborative Storytelling with GPT-3; SAGA: Collaborative Storytelling with GPT-3. *Companion Publication of the 2021 Conference on Computer Supported Cooperative Work and Social Computing*. [online] Available at: https://doi.org/10.1145/3462204.3481771 [Accessed 1 Oct. 2022].

Shen, T., Lei, T., Barzilay, R., Jaakkola, T. and Csail, M., (2017) Style Transfer from Non-Parallel Text by Cross-Alignment. *Advances in Neural Information Processing Systems*, [online] 30. Available at: https://github.com/shentianxiao/language-style-transfer. [Accessed 25 Oct. 2022].

Shin, T., Razeghi, Y., Logan, R.L., Wallace, E. and Singh, S., (2020) AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.4222–4235. Available at: https://arxiv.org/abs/2010.15980v2 [Accessed 25 Oct. 2022].

Song, K., Tan, X., Qin, T., Lu, J. and Liu, T.Y., (2019) MASS: Masked Sequence to Sequence Pre-training for Language Generation. *36th International Conference on Machine Learning, ICML 2019*, [online] 2019-June, pp.10384–10394. Available at: https://arxiv.org/abs/1905.02450v5 [Accessed 25 Oct. 2022].

Su, Y., Wang, X., Qin, Y., Chan, C.-M., Lin, Y., Wang, H., Wen, K., Liu, Z., Li, P., Li, J., Hou, L., Sun, M. and Zhou, J., (2021) On Transferability of Prompt Tuning for Natural Language Processing. *NAACL 2022 - 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, [online] pp.3949–3969. Available at: http://arxiv.org/abs/2111.06719 [Accessed 25 Oct. 2022].

Sun, X., Fan, C., Sun, Z., Meng, Y., Wu, F. and Li, J., (2020) Summarize, Outline, and Elaborate: Long-Text Generation via Hierarchical Supervision from Extractive Summaries. [online] Available at: https://arxiv.org/abs/2010.07074v2 [Accessed 23 Oct. 2022].

Tan, B., Yang, Z., Al-Shedivat, M., Xing, E.P. and Hu, Z., (2020) Progressive Generation of Long Text with Pretrained Language Models. *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, [online] pp.4313–4324. Available at: https://arxiv.org/abs/2006.15720v2 [Accessed 23 Oct. 2022].

Vaswani, A., Brain, G., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I., (2017) Attention is All you Need. *Advances in Neural Information Processing Systems*, 30.

Vu, T., Lester, B., Constant, N., Al-Rfou, R., Cer, D. and Research, G., (2021) SPoT: Better Frozen Model Adaptation through Soft Prompt Transfer. [online] pp.5039–5059. Available at: https://arxiv.org/abs/2110.07904v2 [Accessed 25 Oct. 2022].

Xu, J., Ren, X., Zhang, Y., Zeng, Q., Cai, X. and Sun, X., (2018) A Skeleton-Based Model for Promoting Coherence Among Sentences in Narrative Story Generation. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, [online] pp.4306–4315. Available at: https://arxiv.org/abs/1808.06945v2 [Accessed 23 Oct. 2022].

Xu, P., Patwary, M., Shoeybi, M., Puri, R., Fung, P., Anandkumar, A. and Catanzaro, B., (2020) MEGATRON-CNTRL: Controllable Story Generation with External Knowledge Using Large-Scale Language Models. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, [online] pp.2831–2845. Available at: https://arxiv.org/abs/2010.00840v1 [Accessed 23 Oct. 2022].

Yao, L., Peng, N., Weischedel, R., Knight, K., Zhao, D. and Yan, R., (2019) Plan-and-Write: Towards Better Automatic Storytelling. *Proceedings of the AAAI Conference on Artificial Intelligence*, [online] 3301, pp.7378–7385. Available at: https://ojs.aaai.org/index.php/AAAI/article/view/4726 [Accessed 23 Oct. 2022].

Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., Choi, Y. and Allen, P.G., (2019) Defending Against Neural Fake News. *Advances in Neural Information Processing Systems*, [online] 32. Available at: https://rowanzellers.com/grover [Accessed 23 Oct. 2022].

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J. and Dolan, B., (2019) DialoGPT: Large-Scale Generative Pre-training for Conversational Response Generation. [online] pp.270–278. Available at: https://arxiv.org/abs/1911.00536v3 [Accessed 23 Oct. 2022].

Zhao, J.J., Kim, Y., Zhang, K., Rush, A.M. and Lecun, Y., (2018) Adversarially Regularized Autoencoders. [online] pp.5902–5911. Available at: https://proceedings.mlr.press/v80/zhao18b.html [Accessed 25 Oct. 2022].

Zhu, Y., Lu, S., Zheng, L., Guo, J., Zhang, W., Wang, J. and Yu, Y., (2018) Texygen: A benchmarking platform for text generation models. *41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018*, pp.1097–1100.

Ziegler, Z.M., Melas-Kyriazi, L., Gehrmann, S. and Rush, A.M., (2019) Encoder-Agnostic Adaptation for Conditional Language Generation. [online] Available at: https://arxiv.org/abs/1908.06938v2 [Accessed 25 Oct. 2022].

Zou, X., Yin, D., Zhong, Q., Yang, H., Yang, Z. and Tang, J., (2021) Controllable Generation from Pre-trained Language Models via Inverse Prompting. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.2450–2460.

Final_Report_draft_final-2.docx

ORIGINALITY REPORT

40%	6%	6%	37%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Liverpool John Moores University Student Paper	36%
2	Daniel Probst. "Growing pains: Reacting to negative impacts of deep learning on machine learning for chemistry", American Chemical Society (ACS), 2022 Publication	2%
3	arxiv.org Internet Source	1%
4	Arwa I. Alhussain, Aqil M. Azmi. "Automatic Story Generation", ACM Computing Surveys, 2021 Publication	1%
5	www.researchgate.net Internet Source	1%

Exclude quotes	Off	Exclude matches	< 1%
Exclude bibliography	On		