GENERATING A NARRATIVE WITH EXCELLENT STORY TELLING CAPABILITIES BY GIVING PROMPTS TO PRE-TRAINED MODELS

MOHAMMED AAMER  
(Student ID : PN1130101)

Interim Report

July 2024

# DEDICATION

# ACKNOWLEDGEMENTS

# ABSTRACT

# TABLE OF CONTENTS

[DEDICATION 1](#_Toc685544928)

[ACKNOWLEDGEMENTS 2](#_Toc1348095451)

[ABSTRACT 3](#_Toc2132963568)

[TABLE OF CONTENTS 4](#_Toc2044964634)

[LIST OF TABLES 7](#_Toc1574810178)

[LIST OF FIGURES 8](#_Toc279142929)

[LIST OF ABBREVIATIONS 9](#_Toc774127971)

[CHAPTER 1: INTRODUCTION 10](#_Toc1365356800)

[1.1 Background of the Study 11](#_Toc984350137)

[1.2 Research Questions 12](#_Toc1358486203)

[1.3 Aim & Objectives 12](#_Toc389038508)

[1.4 Significance of the Study 13](#_Toc751710371)

[1.5 Scope of the Study 13](#_Toc1062898036)

[1.6 Structure of the Study 14](#_Toc2068340653)

[CHAPTER 2: LITERATURE REVIEW 14](#_Toc1394020312)

[2.1 Automatic Story Generation 15](#_Toc2145768095)

[2.1.1 Introduction 15](#_Toc807839344)

[2.1.2 Structural Models 15](#_Toc948702747)

[2.1.2.1 Graph-Based Approaches 16](#_Toc1111422160)

[2.1.2.2 Grammar-Based Approaches 16](#_Toc1384145753)

[2.1.3 Planning Based Models 17](#_Toc965511201)

[2.1.4 ML Models 17](#_Toc1814778435)

[2.1.4.1 Story Abstraction 17](#_Toc1075589289)

[2.1.4.2 Script Learning and Generation 18](#_Toc389448381)

[2.1.4.3 Story Completion 20](#_Toc1330043707)

[2.1.4.4 Story Generation 22](#_Toc1978132616)

[2.1.5 Story Evaluation 25](#_Toc1248066672)

[2.1.6 Challenges in Automatic Story Generation 28](#_Toc693211711)

[2.2 Prompt Based Learning 30](#_Toc276314918)

[2.2.1 Brief History of NLP 31](#_Toc1083325890)

[2.2.2 Introduction to Prompting 32](#_Toc1954299913)

[2.2.2.1 Prompt Addition 32](#_Toc1038939926)

[2.2.2.2 Answer Search 32](#_Toc58683428)

[2.2.2.3 Answer Mapping 33](#_Toc1483462202)

[2.2.2.4 Design Considerations for Prompting 33](#_Toc765130583)

[2.2.3 Pre-trained Language Models 34](#_Toc607553140)

[2.2.3.1 Training Objectives 34](#_Toc1856758867)

[2.2.3.2 Noising Functions 34](#_Toc1076777778)

[2.2.3.3 Directionality of Representations 35](#_Toc369516428)

[2.2.3.4 Typical Pre-training Methods 35](#_Toc1661814254)

[2.2.4 Prompt Engineering 37](#_Toc23126687)

[2.2.4.1 Prompt Shape 37](#_Toc1363976326)

[2.2.4.2 Manual Template Engineering 37](#_Toc901991562)

[2.2.4.3 Automated Template Engineering 38](#_Toc408445568)

[2.2.5 Answer Engineering 40](#_Toc1790107174)

[2.2.5.1 Answer Shape 40](#_Toc872946957)

[2.2.5.2 Answer Shape Design Methods 41](#_Toc440128871)

[2.2.6 Training Strategies for Prompting Methods 42](#_Toc940814387)

[2.2.6.1 Training Settings 42](#_Toc318026337)

[2.2.6.2 Parameter Update Methods 43](#_Toc657530009)

[2.2.7 Prompt-relevant Topics 45](#_Toc2112832159)

[2.2.7.1 Few-shot Learning 45](#_Toc1276759940)

[2.2.7.2 QA-based Task Formulation 46](#_Toc332432820)

[2.2.7.3 Controlled Generation 46](#_Toc515059268)

[2.2.7.4 Data Augmentation 47](#_Toc809596006)

[2.2.8 Challenges in Prompting 47](#_Toc1603561539)

[2.2.8.1 Prompt Design 47](#_Toc4737187)

[2.2.8.2 Answer Engineering 48](#_Toc2058250863)

[2.2.8.3 Selection of Tuning Strategy 48](#_Toc836769974)

[2.2.8.4 Multiple Prompt Learning 49](#_Toc1007119775)

[2.2.8.5 Selection of Pre-trained Models 49](#_Toc636523240)

[2.2.8.6 Theoretical and Empirical Analysis of Prompting 49](#_Toc475897150)

[2.2.8.7 Transferability of Prompts 50](#_Toc1867068165)

[2.2.8.8 Combination of Different Paradigms 50](#_Toc870500709)

[2.2.8.9 Calibration of Prompting Methods 50](#_Toc86177093)

[2.3 Summary 51](#_Toc1055275576)

[CHAPTER 3: RESEARCH METHODOLOGY 51](#_Toc1752369079)

[3.1 Introduction 52](#_Toc12295305)

[3.2 Theoretical Background 52](#_Toc1630303715)

[3.3 Methodology 52](#_Toc707701410)

[3.3.1 End-to-End Pipeline 52](#_Toc1444532036)

[3.3.3 Data Preparation 53](#_Toc9736789)

[3.3.4 Implementation 54](#_Toc1685047017)

[3.3.5 Evaluation 55](#_Toc914356339)

[3.4 Tools 56](#_Toc964984672)

[3.5 Summary 56](#_Toc1524769939)

[REFERENCES 56](#_Toc4200066)

[APPENDIX A: RESEARCH PLAN 63](#_Toc360995990)

[APPENDIX B: RESEARCH PROPOSAL 64](#_Toc1141024679)

[APPENDIX C: ETHICS FORMS 65](#_Toc1467331277)

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

# CHAPTER 1: INTRODUCTION

## 1.1 Background of the Study

The quality of machine authoring has increased in the past decade with the invention of large-scale learning models (Rashkin et al., 2018). Today, they can produce texts that are similar to texts written by humans. A more comprehensive review of the published literature revealed issues such as conflicting topics and conflicting personalities (Dziri etal., 2021). These deficits are especially important for open reading activities that require collaboration, like as story construction.

Stories created using these models lack dialogue and global planning cognitive skills. Although separately the sentences in the text may seem meaningful & flowing, when combined, there isn’t much snese in the whole plot. Repetition of sentences in long texts will reduce the quality of the story (Yao et al., 2019). Preparation of content in different formats. Use the instructions & Use keywords and keywords using semantic frames. To get the most out of these projects, PLM often needs to fine-tune the data. One of the challenges in optimizing PLM is that in addition to the required information, the model also tends to examine events arising from the planned content and provide useful information from them. The absence of diversity in generated narratives is a common issue. Special alerts can be employed to address textual challenges. Studies have demonstrated that by adhering to specific guidelines, PLM (Probabilistic Language Models) can effectively tackle existing or upcoming tasks without necessitating optimization. However, challenges persist as cues are task-specific and pose difficulties in adaptation or repurposing for novel tasks. Even within the same task, instructions may not apply to all situations in large.

## 1.2 Research Questions

We try to address the below points in this thesis:

1. Narrative creation process along with control needs to be optimized in PLM. Can this technique be used with rapid learning to create stories with fewer shots without requiring much tuning?
2. The previous system only used GPT2. Can text be loaded using this model of GPT3?
3. Learning using prompts has been used in many ways to generate text. Is possible to extend its use for story-making work?

## 1.3 Aim & Objectives

We attempt to find several potential images of GPT3 in long-term document creation projects.

Objectives:

* A comprehensive review of the existing literature on the construction of a long story, a timely study and several published articles.
* Figure out what's possible and then create a way to create short and long stories using a few tenses and instructions.
* Evaluate stories created using automated test creation and compare the design to current technology.

## 1.4 Significance of the Study

## Story creation is an active area of ​​research. Although the short story has been studied extensively, its longer history has been less researched. Although optimization has been used as a method in previous studies, there is no research on creating stories without optimization. This study also explores recent developments and some inflections in cognitive-based learning. This can help writers get new ideas or overcome writer's block.

## 1.5 Scope of the Study

## The parameters of this thesis endeavor are outlined as follows:

## The completion timeframe for the thesis is set at 17 weeks subsequent to the submission of the research proposal.

## Open-source software and models will be utilized for experimentation purposes.

## Experimentation will be carried out utilizing publicly accessible GPU resources like Google Colab.

## Human assessment of the produced narrative is excluded from the thesis scope. Evaluation will solely center on automated metrics.1.6 Structure of the Study

The study's framework is structured as follows:

* Chapter 1 – Introduction: Offering an overview and background of the research.
* Chapter 2 – Literature Review: Examining related works in Automatic Story Generation and Prompt-Learning.
* Chapter 3 – Methodology: Providing a comprehensive explanation of the methodology applied during experimentation.
* Chapter 4 – Implementation: Detailing various experiments conducted for the story generation task.
* Chapter 5 – Results: Discussing the outcomes of the experiments outlined in Chapter 4.
* Chapter 6 – Conclusion: Summarizing the thesis work and exploring potential future enhancements.

# CHAPTER 2: LITERATURE REVIEW

## 2.1 Automatic Story Generation

### 2.1.1 Introduction

Automated story generation involves the creation of a series of events or actions that fulfill specific criteria and can be narrated as a story using a computer program (source). These narratives often encompass settings, characters, and objects, sometimes conveying a particular message or purpose intended by the author. Despite the development of story generation techniques since the 1960s, they have yet to achieve the level of creativity and complexity inherent in human-crafted stories, thus being classified as weak-intelligent machines.

For a computer to exhibit true creativity, it must generate stories distinct from those previously encountered. This entails considering various aspects of characters, including the story's setting, characters' desires and motivations, and their interactions and conflicts. The multitude of potential story characters can pose challenges for computers in effectively crafting unique narratives. Moreover, the purpose, beliefs, and interests underlying a story further influence the storytelling process. Open-ended story creation, which involves generating stories without relying on pre-existing models, presents additional hurdles: automatically devising a structured framework and evaluating story progress to inform the creation process.

Research endeavors have sought to dissect and explore the intricacies of the story design process. Gervás (2009) delved into how such machines attempt to foster creativity in individuals, emphasizing the role of computation. Subsequent advancements have been made in the realm of automatic storytelling. Yang et al. (source) conducted a study on planning and reasoning in computer-generated narratives, while Kybartas and Bidarra (source) categorized narrative generation into four tiers based on the degree of automation in story and space production: manual, story generation, spatial production, and automated story and location elements.

### 2.1.2 Structural Models

#### In the field of story creation, patterns are used to create stories by dividing stories into slices that follow a predetermined pattern. These gaps are filled by adding similar gaps from previously written and recorded stories, including the interaction between elements of the developed stories.

#### 2.1.2.1 Graph-Based Approaches

#### There are many ways to create stories using computers, including creating graphics and using templates. In the story mapping approach, a branching graph is created that represents all possible stories in a particular location, and stories are created by selecting linear paths throughout the graph. The quality of the story created depends on the quality of the image created, and adding constraints to the search process can improve the results. Maranda [104] developed a diagram for the creation of legends based on Propp's story model, looking for pieces of information that match the functions represented by the nodes in the diagram and connecting them together to form a story. The SCHEHERAZADE system was proposed by Li et al. [90] collected human knowledge about a particular type of text and studied drawings based on these notes, then cross-referenced the drawings to create stories. SCHEHERAZADE's images are similar to Maranda's images, but they are not cyclical and are linked to specific events in the story.

#### 2.1.2.2 Grammar-Based Approaches

The use of grammar to create stories began with Lakoff [85], who transformed Propp's story structure into story grammar and used the rewriting process to create stories, using Propp's work as a text for explanation. This has inspired the development of other grammars, such as Pemberton's old French grammar [127] used in the GESTER program [128] which creates good people with a beginning, middle and end. BRUTUS [31] is another system that creates narratives based on story writing by using patterns to group stories into themes. But this particular grammar is limited only to specific details and can only produce a small number of stories, it is necessary to have a more general grammar. Rumelhart [149] proposed the first universal story grammar and has since proposed many other grammars, including the Thorndyke proposal [170].

While story templates are easy to use and can create great stories, they have limitations in their ability to create stories with multiple characters and personas. The personas they create will be inconsistent or unreliable due to lack of focus on the subject. semantics. They will also face the problem of over-generation by producing non-stories articles that are considered stories.

### 2.1.3 Planning Based Models

Story writing theories have been criticized for their inability to understand stories [23, 24] because they focus on the content of stories rather than their theory and cannot be used for conflicting stories or multiple actors. In response, story-oriented theory emerged, which views story as a chain of interrelated events that serve a specific purpose [181]. Cook's [46] Conspiracy: The Master Book of All Plots, which includes the process of writing sentences and instructions on how to combine them to form a complete plot, is based on stories that guide thinking and are used as the basis for calculations. However, the plotter technique proposed by Egger et al. [53] Create a plot from the chapters of Plotto, which have limitations of the same story because it is based on the current state of the story rather than its previous state. The use of artificial intelligence algorithms in automatic story generation should provide reasons along with the initial situation and goals for determining actions that will ensure that the story achieves its goals, and possibly add techniques that will improve the quality of the created story.

### 2.1.4 ML Models

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

#### 2.1.4.1 Story Abstraction

To increase the effectiveness of learning and thinking in machine learning (ML), it is necessary to create simple story models (so-called story abstractions) that focus on the importance of events and organizations in the story and enable possible overlaps between stories [105]. There are many ways to create story abstractions, and each has its own advantages and disadvantages. Chambers and Jurafsky [36, 37] and Jans et al. [76] represent a story as a sequence of events using (indicator, dependency) pairs to associate each event with a grammatical role performed by an actor. While this representation can capture the relationship between expressions and their arguments, it can also introduce conflicts between expression-object tuples and limit the representation of an actor's action. [18, 19] use the Open IE system to extract Rel-gram patterns, which are ternary relationships of the form (arg1, Relation, arg2). This representation can show interaction between entities, but may differ from other methods. Pichotta and Mooney [131] proposed the four-event representation of verbs (element, object, preposition), which can show the interaction between entities and was later developed by teaching prepositions [133]. Martin et al. [105] and Tambwekar et al. [167] also used the 4-event representation of the form (learning, teaching, product, change), while Ammanabrolu et al. [6] use the five-fold notation (object, verb, preposition, object, M). Yao et al. [186]

#### 2.1.4.2 Script Learning and Generation

The learning and generation process will use statistical models to analyze relationships between events in a story and predict new events that will occur in a series of events. These methods often include the use of parsers to extract expressions and arguments, the use of kernel reference parsers to identify sentences that refer to the same entity, and techniques such as context interfacing or cross-n-grams to create chain-of-event patterns. This model can be used to create a list of possible events that will fit into the chain of events. The relationship between the events of the hero. They then used vector machines to describe the physical relationship between two moments, as described by Chambers et al. [38] and create a list of possible events that could be part of the chain. This study improves on the previous method by using social relations to extract the state of individual actors and PMI to extract relations between states. [76] proposed the use of crossed n-grams to learn event chain statistics by combining each event with the next three events in the chain; developed the PMI method by thus reducing data sparsity and improving the training process [36]. The binary probabilistic ranking function used in this method models the chain of events sequentially, scoring events according to their position in the chain, taking into account the pre- and post-developmental events fixed in the previous method.

Balasubramanian et al. [18, 19] introduced the Rel-grams system, a Markov model similar to Jans et al. [76] But focus on social cohesion rather than conflict. This improvement allows the system to estimate one of the parameters (if there is a relationship) and the other parameter. and model interactions between different parts. This improvement allows the system to create event chains for the entire story, rather than separate organizations based on event chains formed by verb dependency pairs. However, the complexity of the process also increases the complexity of the statistical model. This system demonstrated greater accuracy of prediction compared to the system with instruction-dependent pairs, but the complexity of the process also increased the complexity of the statistical model. Estimating the situation compared to previous calculation methods. Ruedinger et al. [148] trained a log-bilinear model to predict story events, arguing that the predictive model can outperform linguistic modelling. The language discrimination model performs better compared to previous calculations. Pichotta and Mooney [133] used short-term (LSTM) RNN to learn stories based on statistics. Their model is capable of predicting the name or important information of the bad situation and shows that the performance is better than many other sources. They also continued their work to predict events directly from raw text without using explicit event models [132] and found that the difference between the raw text model and event model testing was small; This shows that subtraction can be useful for prediction: it is not necessary, especially in encoder-decoder setups. Granroth-Wilding and Clark [63] compared different methods to provide vector representations of event predictions and parameters and used these representations in an integrated neural network model that predicts the outcome of two events in Success in the chain of events. [114] introduced the Story Completion Test (SCT) and created the ROCStories corpus to test the machine's ability to choose the correct story endings for the story. Some researchers are using ROCStories to create classification models that can pick the ending of a story based on a variety of factors, including modeling and neural patterns. Chaturvedi et al. [39] proposed a learning model based on three assumptions: event sequence, logic theory, and semantic consistency. When their work is better than previous methods, researchers suggest a good analysis of human behavior and social culture to improve education. Lin et al. [95] proposed a similar model. Mustafazadeh et al. [116] trained a simple embedding model to predict the correct story ending based on the embedding of story content and two endings. Wang et al. [175] used generative adversarial networks where the model generates false models based on the background story and the discriminative model separates the model from the spurious model. The discriminator has three models: an LSTM-RNN model that represents sentences, an LSTM-RNN model that represents data, and a bilinear model that calculates the similarity between information content and articles. Recent studies [41, 74, 91, 93] show a significant improvement in SCT results when training large datasets.

#### 2.1.4.3 Story Completion

Unlike previous research on predicting new events by evaluating familiar events, the story serves the purpose of executing the plan given the context of the story [ 65 ]. Most systems in this category end a story by creating a story ending based on previous events. [145] used Children's Book Test (CBT) data as a narrative corpus. The first generation story takes the first story of 20 sentences as input and generates the next sentence based on CBR. RNN is used to generate the final sentence, which is compared word for word with the original sentence 21 as the gold standard. The main purpose of this study is to use various language measures to measure story production. Additionally, Hu et al. [72] reported a context-aware hierarchical LSTM model that can predict future events given past events. This model creates a series of narratives about future events. It considers the order of events at two levels: the order of words and the time order of events. It also treats story content as additional content. [92] proposed a Seq2Seq model that learns to use feedback to generate different stories. They believe that the traditional Seq2Seq model trained with maximum likelihood is suitable for gold-standard work. But this is not the case for the end of the generation story, and it is an ending that deserves to be accepted. To improve the quality of the results, electronic machines are encouraged to produce results similar to the results of human-written stories. Therefore, a binary classifier is trained to classify the output as human-generated or machine-generated. This distribution is used as a reward for the generator in reinforcement learning algorithms. Zhao et al. [189] improved the accuracy and quality of generated stories using the copying and masking process for the traditional Seq2Seq model proposed by See et al. [156]. To avoid word of mouth (OOV) problems, the published technique is used to create story endings directly from the previous events of the marker. The scope method is used to overcome the word equation problem by checking the maintenance history to adjust the service vector of future maintenance. A new intent function for semantic impact loss has been added to maximize the impact of the final design and story. It is calculated as the cosine similarity between the graphic meaning vector and the meaning vector created at the end. Although the resulting semantic vector is the final latent output of the encoder, the drawing semantic vector is calculated according to the method proposed by Ma and Sun [102]. The machine is trained using a reinforcement learning technique that uses different metrics as the reward function to simulate the human story process. [65] proposed a neural model that considers two assumptions for generating story endings: story coherence and tacit knowledge of the story. The events of each story, its characters, and the relationships between events play a role in the overall story. Therefore, incremental coding is used to ensure consistency in the content of the story. The model uses ConceptNet as a cognitive and control system that knows this from many angles, to follow the way the human brain understands stories and presents information based on background knowledge. The model can create the same story. [176] proposed a model based on GPT-2 [137] to form the missing part of the incomplete story by correcting the sentences formed by the previous and next sentences. Their model can create a composite story that fits given objectives. Similarly, Wang and Wan [178] proposed a model to generate the incomplete story at each point of the incomplete story. Unlike the model of Wang et al. [176], this pattern can form a sentence at the end of the story. It was adapted from Transformer [174] using a common algorithm for encoder and decoder. BERT (Bidirectional Encoder Represented by Transformers) is used as a separate encoder. BERT is a new representation language developed for pre-training deep two-sentence representation from anonymous text [51].

#### 2.1.4.4 Story Generation

Researchers are keen to use Seq2Seq models to create success stories due to their success in various NLP projects. [75] combined two off-the-shelf machines to create a narrative that creates stories while providing a series of short narratives. First, the words in a sentence are automatically translated using machine translation (SMT) and then a deep RNN is used to encode each sentence as a unit and decide the heart of the story. However, the resulting content was not completely semantically related to the description, and the overall score of the evaluation was not very high. [44] trained an RNN model to generate stories by predicting the next sentence. The model has two models: RNN Encoder-Decoder (RNNED), which shows expressions for vector representation and vice versa; . The model can create sentences that are grammatically correct and have all the correct content, but some words are used incorrectly in the created sentences. Harrison et al. [66] used an RNN to perform Markov Chain Monte Carlo (MCMC) sampling to generate stories, similar to the two-stage process used by Choi et al. [44].

Although RNNs are successful in many Seq2Seq problems, they do not follow the expectations in the story but often fail to make the story connection a few sentences later. According to Khandelwal and others, this is because a story is a longer sequence of events than RNN can support. [81] found that the prediction of RNN is based on a small number of previous signals. Therefore, according to story generation, RNN loses the connection between the current event and the distant event, which affects the coherence and integration of the story. [167] developed a controllable RNN generator that provides start and end words and uses additive learning to guide the RNN from start to finish. The design of the reward function depends on two factors: the distance between the next event and the last event, and the frequency with which the next event occurs before the last event in the story body. Amanabrolu et al. [6] used the gradient deep learning method of Tambwekar et al. [167] improved event generation and introduced four event-to-model transition methods to improve the quality and satisfaction of the written story. These models include retrieval and repair models, sample collection models, coupled models with Monte Carlo beam decoding, and Seq2Seq with a finite state machine. The results show that the overall performance of this model is better than a single model. Fan et al. [54] divided story generation into two stages to improve the consistency of generated stories. They created the story space using the convolutional language model, converted the space into text using the Seq2Seq model with the fusion mechanism to develop the relationship between the story and the space, and guided the idea of ​​modeling in context. This happened after research that divided story formation into two phases. [186] proposed a hierarchical generator that uses plot and text structure to generate stories based on names. They extract the story from each story in the body using the RAKE algorithm [147] to determine the most important words in each sentence, and then use two strategies to create ghost People: dynamic and static. The dynamic mode creates the next word in the story and the next sentence in the story at each step, while the static mode creates complete sentences that are then translated into text. They used the Seq2Seq model to transcribe stories, and the results showed that organizing stories led to better stories in terms of honesty, relationships, satisfaction, and liking of all users, but such models created worse stories of the same, and consistent models [183] ​​included the most important phrases in the sentence (skeleton He used additive learning to learn different sentences in a story and then trained the Seq2Seq model to generate sequences based on the sequence. However, integration may be negatively affected by the length of ideas and lack of familiarity with the ideas. Chen et al. [40] also generates context as an intermediate step before story generation, using external annotations to generate high-level plans from the training corpus, and then uses natural annotation to illustrate the preparation of the structure on how to create the process. They used the Seq2Seq model to generate new names and patterns, and while this system outperformed previous methods [54, 183, 186], the authors suggested that more robust methods were required to improve story-level consistency. Zhai et al. [188] proposed a hybrid model that can prepare stories from a small organization using a generator to prepare stories by taking the physical model of body graphs extracted from the story corpus, and generate a written story based on the stories using the neural surface module. story plan. They analyzed the story of global integration in terms of inclusivity, specificity, and intersectionality. Araz [10] proposed a modified neural network for inspiration-based story generation that creates new and possible stories, while also generating repetitive and incorrect words, and the instructions are not taken into account as carefully as expected.

Previously learned language patterns indicate good language processing ability. Text models created by the GPT-2 model [137] show that this model can produce texts similar to human handwriting. This led researchers to use pre-trained models to create stories. Saw et al. [157] proposed two models: a preliminary study of the Fusion model [125] and a minimal version of GPT2 called GPT2-117 [137]. Like other works [10, 54], this model is trained to generate inspiration-based stories. Overall, the authors found that the GPT2-117 model outperformed the Fusion model in many aspects but produced repetitive and less diverse forms when using the decision-making algorithm. Holtzman et al. [70] found that such patterns produce boring, inconsistent, or repetitive texts. [64] believe that the poor performance of pre-trained models is due to lack of information. Story creation, as an open-ended creation project, does not provide gold-standard results against which to compare models, unlike other creation tasks such as writing. No amount of creativity hinders the learning process. Therefore, awareness raising activities should be carried out. To solve this problem, the authors suggest using information from external information to create good stories and using various learning projects to capture the relationship between the relationship and the body of the sentence in the story. Their models create better stories in terms of logic and international relations than the underlying models. Inspired by Guan et al. [64], Xu YJ et al. [184] proposed a story-based control system that allows the integration of perceptual information into language structures. At each generation step, the model predicts a set of elements based on the context of the story and then uses these elements to query information regarding the understanding of the relevant elements. The next paragraph of the story is created with the GPT-2 model, reaching the main story and the climax. [94] proposed an open-source design based on Transformer. They use causal corpus to train causal models and event models, and to establish correlations and multivariate relationships. To encourage diversity, they also developed a method for conflicting lexical constraints that allows the decoder to choose one of the given words or phrases to include in its output. This method is used to choose between different morphological variants of the same lemma.

### 2.1.5 Story Evaluation

Automatic evaluation of stories is a difficult task in story creation due to the content of the story, the diversity of the measurement model, and the height of the story content. Most systems rely on human judgment to evaluate the creation of stories, but this method is not easy, time-consuming, content, and there is no gold standard for comparison. Human auditors can also use their knowledge and intuition to supplement inconsistent stories and rate inconsistent stories higher than they should be. [108, 129, 90, 144, 126]

There are many ways to evaluate the performance of the manufacturing process. One method is the narrative completion test proposed by Chambers and Jurafsky [36] to assess unsupervised learning and production. In this experiment, the events in the story are given by removing one event, and the system is asked to create a prediction list for the missing events according to the conditions. SCT proposed by Mostafazadeh et al. [114] is a variant of the descriptive completion test designed for educational auditing. It presents a set of four sentences and two alternatives to the fifth sentence, labeled "true ending" and "false ending", and evaluates the system's ability to choose an outcome for each story. Granroth-Wilding and Clark [63] proposed the Multiple-Choice Narrative Completion (MCNC) test, which provides a system with five decisions that can select the missing event, allowing the system to use more information about the context and choice list. Rich data and better comparison of different story-making processes.

**Statistical models.** This model predicts event conditions based on various statistics:

* **N-gram overlap**: As with other NLG functions, story generation quality can be evaluated by calculating the n-gram overlap between predicted and expected events. This includes measures such as BLEU, METEOR, CIDER and ROUGE. However, BLEU is the most commonly used metric in story creation; see for example [7, 65, 105, 183, 186].
* **Perplexity**: Perplexity is a frequently used metric to measure the quality of language structures. It measures how well the model predicts given previous data, where less stress indicates the true level of accuracy.
* **Pointwise Mutual Information**: PMI is used when choosing an event among many others. It is based on the common word count and selects cases whose common entities in the description have the highest PMI score (see Section 5.2). It was first proposed by Chambers and Jurafsky [36] and has been adopted by others such as [76, 148].

**Embeddings models.** Embedding models predict events based on word-level or sentence-level embeddings. Different measures can be used, including Skip Thought Cosine Similarity (STCS), Embedding Mean Cosine Similarity (EACS), Vector Extremal Cosine Similarity (VECS), and Greedy Matching Score (GMS). The average maximum likelihood model proposed by Roemmele et al. [146] is a word-level embedding model that calculates the average of similar embeddings for each trailing word and then selects the ending with the highest average. Deep semantic model is another design model used in SCT by Mostafazadeh [114]. Conditionally generative adversarial network models have also been used to generate stories; here a separator is used to select the correct story ending [92]. > One problem is that addressing the problem of story generation, the classification problem will lead to the quality of classification, but does not understand the content of the story and may not have enough ideas through the generation. Also, the creative nature of the story means that it will be uninteresting if it is too predictable. Additionally, there is no word “rule” in story generation that could affect the evaluation criteria, requiring the system to choose a particular response or be penalized for not doing so. The resulting stories are compared to spoken words, descriptions of the stories are analyzed, statistical analysis is used, details of the stories are evaluated based on emotional or social evaluation information, and the level of tension in the stories is evaluated. . . Romell et al. [145] used lexical agreement, style matching, and entity co-reference as language indicators to create stories, while Purdy et al. [136] analyzed grammar, chronology, local context and descriptive production. Kartal et al. [80] calculated the belief of the story created by dividing the belief in each action in the story, while León and Gervás [89] used 13 variables such as satisfaction, stress, thoughts, and emotions to tell the story to create a collective action of cooperation, the emergence of patterns and emotions. Yao et al. [186] used interactive stories and in-story retellings to measure differences in story production. Wang et al. [177] used social media likes as a predictive measure of story quality, and Sagarkar et al. The crowdsourced interestingness rating of the story continues. Behruz et al. [21, 22] evaluated interesting stories based on the unexpected and ability to generate predictions, while O'Neill and Riedl [122] measured interest in stories by calculating the evacuation cost at different times and finding suspicious locations.

### 2.1.6 Challenges in Automatic Story Generation

Although significant progress has been made in the field of automatic story generation, the field has not been as successful as expected and still faces many challenges and limitations such as.

* **Dispersion**.
* **Domain knowledge**.
* **Seq2Seq models**.
* **Pre-trained language models**.
* **Story interestingness**.
* **Objective evaluation**.

## 2.2 Prompt Based Learning

### 2.2.1 Brief History of NLP

The lack of an information recording and evaluation system makes it difficult to compare the performance of different systems. This can lead to a system's success being attributed to experience rather than appropriate resources, and systems being rated based on individual opinion rather than objective measurement of the brand. Standardizing these elements will allow more accurate assessment of the strengths and weaknesses of different models and will facilitate the development and improvement of previous work. However, despite the long history of the development of automated stories, no organization or existing measurement system has yet been widely accepted as a standard. In recent years, some researchers have begun to reuse existing stories and common-sense experiences, but more work is needed to develop a resource and measurement system to evaluate the design process.

### 2.2.2 Introduction to Prompting

#### One of the main problems of supervised learning is the large amount of recorded data required to train a model to perform a specific task. The NLP method uses instructions to solve this problem by training a language model (LM) to model the outcome of the text itself (not the outcome of the output of specific quote ideas). This allows the use of LM to predict expected results for a particular task without needing a large amount of recorded data.

#### 2.2.2.1 Prompt Addition

The prompt function is applied to the input text to create the result, causing the text to change. The process generally follows a two-step process: (1) uses a template, which is a string with spaces for input and intermediate response, which should ultimately do what is needed; (2) fills the input text placeholder with the actual input. For example, in an emotional review, the example would be the headline "Overall, [X] is the movie [Z]" and "I liked this movie." "I liked this movie. Overall, it's a [Z] movie." In machine translation, the pattern where input and response are marked by language would be "Finnish: [X] English: [Z]". names. These prompts can be "close" prompts (with a space between the text) or "prefix" prompts (enter text before the space). In some cases, fields may be represented by numerical IDs or continuous vectors rather than symbols. The number of parts [X] and [Z] can be adjusted according to the needs of different functions. (Kumar et al., 2016; McCann et al., 2018; Radford et al., 2019; Schick and Schütze, 2021a)

#### 2.2.2.2 Answer Search

The aim is then to find the text with the highest score, which is considered as á°, which increases the score of the language model (LM). To do this, we define a set of allowed mean responses generated including Z to represent the set in Y = {++, +,. ) ", "good", "OK", "bad", "terrible" } -, -- }).The answer may be in Z, it fills the field immediately, the command "fill" has a definite answer, to which it immediately "answers" We name it. Finally, we search for possible answers in Z using a pre-trained LM to calculate the probability of corresponding instructions. This search could be an argmax search to find the maximum value or a random sampling to maximize the output according to the LM distribution.

#### 2.2.2.3 Answer Mapping

Finally, the goal is to convert the answer with the highest score (á) into the highest score (Å·). In some cases, the response itself is the output (for example, in language-based tasks such as translation), but in other cases multiple responses may correspond to the same output. For example, multiple emotion words (e.g., “excellent,” “excellent,” “excellent”) can represent a category (e.g., “++”). In this case, the search results for the desired results must be specified.

#### 2.2.2.4 Design Considerations for Prompting

In this section, we explain several design considerations when creating NLP workflows:

* Pre-trained Model Choice
* Prompt Engineering
* Answer Engineering
* Expanding the Paradigm
* Prompt-based Training Strategies

### 2.2.3 Pre-trained Language Models

There have been many comprehensive evaluations of the effects of prelearning models (LM) on language processing (NLP) in the “pretraining and remediation” paradigm (Raffel et al., 2020; Qiu et al., 2020). Xu et al., 2021; Doddapaneni et al., 2021). This section provides an overview of various pre-LM studies focusing on factors related to motivation.

#### 2.2.3.1 Training Objectives

#### The main training goal of the prior learning model (LM) is usually to predict the probability of text x. The goal of language modeling (SLM) is to optimize the probability P(x) of the text in the training corpus ( Radford et al., 2019 ). The text is usually estimated autoregressively; This means that the model predicts tokens one by one, usually from left to right (although there may be other orders). Another approach to the SLM objective is the denoising objective, which applies some noise function xÌ = fnoise(x) to the input sentence and tries to predict the input sentence given the noise P(xxÌ). There are two types of denoising targets: corrupted text reconstruction (TO) targets, which provide only the noise of the input sentence to its pristine state, and full-text reconstruction (FTR) targets, which reconstruct the entire text, whether loud or voiced. loud. No (Lewis et al., 2020a).

#### 2.2.3.2 Noising Functions

For reconstruction purposes, the type of damage used to generate the noisy text xÌ can affect the performance of the learning algorithm. Additionally, preliminary information can be included by controlling noise. For example, noise can be applied to entities in the sentence to encourage the model to better estimate location. Various types of noise can be used, as shown in Table 4. Masks may be different or specifically designed to display foreknowledge. Replacement (e.g., Raffel et al., 2020 ) is similar to a mask, but the token or polysymbols are replaced with other symbols or words, such as image fields ( Su et al., 2020 ). Deletion (e.g., Lewis et al., 2020a ) involves removing one or more tokens from the text without adding [MASK] or another token. This work is often combined with a reconstruction of the full text. Planning (e.g., Liu et al., 2020a ) involves splitting the text into different parts (tokens, sentence intervals, or sentences) and rearranging them into new text.

#### 2.2.3.3 Directionality of Representations

The introduction of computational representation is an important aspect to consider when first understanding LMs. Generally speaking, there are two ways to count these representations: from left to right, where the representation of each word is based on the word itself and all previous words that occur in the sentence; words in sentence All words, including words to the left of the current word. It is also possible to combine these ideas or present the events in one decision, but this process is rare. Representing the direction of computation is often achieved through face tracking, which covers important aspects in tracking models such as Transformer architecture ( Vaswani et al., 2017 ).

#### 

#### 2.2.3.4 Typical Pre-training Methods

Some of the popular pre-training methods are - Left-to-Right Language Model, Masked Language Models, Prefix and Encoder-Decoder

### 2.2.4 Prompt Engineering

Just-in-time engineering is creating a function to optimize the performance of a specific downstream job. This may include finding the best model for the job using a manual or machine tool, including an instruction sample.

#### 2.2.4.1 Prompt Shape

The choice of instructions can affect the performance of previous models of basic tasks. Completion prompts involve filling in gaps in the text and are effective for problem-solving tasks using cover language patterns ( Petroni et al., 2019 ; Cui et al., 2021 ). On the other hand, preorders containing continuations of strings are suitable for tasks with structure or autoregressive language (Li and Liang, 2021; Lester et al., 2021). The new structure model can be used with two types of prompts. When working with tasks that require multiple inputs (such as sorting pairs), there must be room for multiple inputs in the current model.

#### 2.2.4.2 Manual Template Engineering

One way to create reports is to manually generate patterns based on human perception. This approach was used to create the LAMA dataset ( Petroni et al., 2019 ), which provides guidelines for searching for information in language structures. Manually prepared prefix verbs are also used for a variety of tasks, including question answering, translation, and reasoning ( Brown et al., 2020 ). Pre-designed models have also been used in multishot studies for tasks such as text classification and legal text (Schick and Schütze, 2020, 2021a, b).

#### 2.2.4.3 Automated Template Engineering

There are many problems with self-designing instant models, including the time and experience required to do so and the fact that even experienced designers may not see the best (Jiang li al., 2020c). To solve these problems, automated processes have been developed that enable the rapid creation of prototypes. These methods can be divided into those that use discrete hints (real strings) and those that use continuous hints (e.g., in the context of the underlying language structure). Additionally, activation can be static (using the same model for each input) or dynamic (creating a specific model for each input).

Discrete Prompts

There are several methods for automatically discovering prompts in a discrete space, usually corresponding to natural language phrases.

Continuous Prompts

The purpose of using hints in natural language processing (NLP) is to find ways to allow the language model (LM) to function properly. One way to use cues is to use extended instructions, also known as soft instructions, which carry out the instructions directly to the model rather than using natural human interpretation. The extension directive removes two restrictions: it allows embeddings of language templates to be embeddings of words in a language (not a natural language) and allows templates to contain them; The individual cannot adjust based on information coming from downstream activities rather than activities. Pre-trained LM parameters are parameterized.

There are several methods that use continuous prompts in NLP.

* **Prefix Tuning**
* **Tuning Initialized with Discrete Prompts**
* **Hard-Soft Prompt Hybrid Tuning**

### 2.2.5 Answer Engineering

Response engineering is the process of exploring the response space and raw data to build a good predictive model; Hint engineering involves creating an appropriate strategy for the hint process. For intervention engineering, two dimensions need to be considered: the form of the intervention and the nature of the intervention design.

#### 2.2.5.1 Answer Shape

The form of the answer refers to the level of detail, which can be different in natural language processing (NLP) tasks. One option is to use determiners, which can be single words from the word structure before the spoken word or one of the words. Another option is to use intervals, which are short multi-marker intervals often used with closing prompts. Sentences or documents can also be used as common responses to prefix prompts. The choice of the correct answer depends on the task being performed. Token or response responses are frequently used in classification tasks such as classification theory ( Yin et al., 2019 ) and relation extraction ( Petroni et al., 2019 ), also known as recognition ( Cui et al., 2021 ). Answers or sentences are often used in word formation tasks (Radford et al., 2019) and multiple choice questions where multiple sentence scores are compared (Khashabi et al., 2020).

#### 2.2.5.2 Answer Shape Design Methods

To obtain a good engineering response, not only the shape of the response must be considered, but also how to generate the appropriate response and how to map the output location if the response is not used as the final output.

Manual Answer Search

There are two main concepts in the design book for creating response space and drawing for output space: unlimited space and bounded space. In the case of unrestricted space, the response space is usually the space of all tokens (Petroni et al., 2019), a long-distance range (Jiang et al., 2020a), or the location of token arrays (Radford et al., 2019) and the identity map to direct the response to the final version. use it. Limited space is generally used for activities where label space is limited, such as classifying text or recognizing organization or answering multiple questions. For example, Yin et al. (2019) and Cui et al. (2021) Manually generated words or labels to be used as response fields for text classification and task recognition tasks. In multiple choice question answering, language models are often used to calculate the outcome probability in multiple choice question answering (Zweig et al., 2012).

Discrete Answer Search

Compared to speed searching in natural language processing (NLP), searching for automatic answers is less researched, although some methods have been developed for regular and unstructured answers. Response paraphrasing involves expanding the initial response space by creating a paraphrased response using methods such as back-translation (Jiang et al., 2020b). The pruning-before-search method first creates the pruned answer space and then uses the algorithm to search in this space to select the final answer. Schick and Schütze (2021a) and Schick et al. (2020) defined characters containing at least two characters of the alphabet as the first response field in the anonymous dataset, while Shin et al. (2020) Logical classifier using content representation based on [Z] tokens. Gao et al. (2021) selects the top k words based on their probability and continues to cut the response space based on the correct zero-throw of the training samples. Label parsing used in the relationship extraction study by Chen et al. (2021b), including decomposing each relationship into its components and using them as responses; The result of multiple responses becomes the sum of events for each form.

Continuous Answer Search

Hambazumyan et al. (2021) is one of several studies investigating the use of continuous or “soft” response signals that can be optimized from low-level parameters. They assign a virtual token to each tag and optimize token placement for each category with instant token placement. Unlike other methods, this method does not use embeddings learned from language models, but instead learns new embeddings for each tag from scratch.

### 2.2.6 Training Strategies for Prompting Methods

The methods described in the previous section provide a way to obtain appropriate instructions and consistent responses. The next step is to consider a training strategy that has a clear training model as well as a hint system.

#### 2.2.6.1 Training Settings

In many cases, a quick method without specifying the language pattern can be used for the following tasks by using previously learned language patterns to write in cloze or prefix prompts. Since there is no information about the relevant function, it is called a zero-throw field. However, there are still methods that use information along with instructions to teach the model. These methods fall into two categories: full data learning, where a large number of training samples are used to train the model, and several hours of training, where some samples are used to train the model. Suggestions are particularly useful in learning multiple times because often there are not enough instructions to specify the desired behavior and the instructions can be used to guide the model to the right. It is important to note that for most of the cue design methods described in the previous section, although the instructional model is not explicitly used for training models for the water function below, it is often used to create or implement the instructions that the function below will use. . . This means that this is not true zero-sum learning for basic tasks (Perez et al., 2021).

#### 2.2.6.2 Parameter Update Methods

There are generally two types of limitations in the cue-based downward learning process: not exceeding the previous model and those that cannot be obtained from the cue. Determining parameters is an important design decision that can affect how the model will perform under different conditions. There are five recommended corrections, including (i) whether the negative of the basic language structure has been changed, (ii) whether there are additional related words, and (iii) whether there are more related words.

Promptless Fine-tuning

Tuning-free Prompting

Fixed-LM Prompt Tuning

Fixed-prompt LM Tuning

Prompt+LM Tuning

### 2.2.7 Prompt-relevant Topics

In this section, we discuss the connection between real-time learning and other learning methods.

#### 2.2.7.1 Few-shot Learning

Few-Shot Learning is a method of learning machine learning in a situation where the number of training examples is small. There are many ways to use it:

* Model agnostic meta-learning
* Embedding learning
* Memory-based learning

Immediate reinforcement is another way to achieve multiple-shot learning, also known as priming-based multiple-shot learning. It involves adding sample text to an existing model to gain information from previously learned language models without a test score. (Finn et al., 2017b; Bertinetto et al., 2016; Kaiser et al., 2017; Wang et al., 2020; Kumar and Talukdar, 2021).

#### 2.2.7.2 QA-based Task Formulation

#### Previous QA-based design work involved conceptualizing various NLP tasks as question-answer questions with the aim of integrating the various activities into a framework. This approach is similar to prompting, uses questions to guide what to do, and focuses on the effective use of information from previously learned language structures. However, previous studies on QA design have not focused on the use of pre-trained models. For multitasking NLP (Kumar et al., 2016; McCann et al., 2018), for data extraction (Li et al., 2020; Wu et al., 2020) and (Chai et al., 2020) for classifying text.

#### 2.2.7.3 Controlled Production

#### Controlled production is a method with additional guidance beyond the input in the design to control the output (Yu et al., 2020). These instructions include tags (Sennrich et al., 2016b; Fan et al., 2018), long specifications (Kikuchi et al., 2016), field tags (Chu et al., 2017), keywords (Saito et al., Content or other types of information). relationship triples (Zhu et al., 2020), keywords or phrases (Grangier and Auli, 2018; Liu et al., 2021c) Some prompts can be considered a type of production control as they use instructions to specify tasks and add additional information to the text. Both controlled generation and cueing methods involve additional, unlearned text, and when used with seq2seq-based prelearning, controlled generation can be thought of as a type of cue learning that involves recommendation and the absence of both cues and prior LM learning can be fine-tuned. + LM fine-tuning” strategy. For example, GSum (Dou et al., 2021). First, generation control usually focuses on controlling the structure or content of the text generated while running in the background ( Fan et al., 2018 ; Dou et al., 2021 ) and does not necessarily have a learning model in the first place. In contrast, the main purpose of using instruction for text generation is to analyze the task itself and take advantage of the previous learning model (Li and Liang, 2021). Second, most current research on cue learning in the literature uses dataset- or task-level cues ( Li and Liang, 2021 ) rather than input-dependent cues ( Tsimpoukelli et al., 2021 ), which are information in the control text. Multigenerational systems will become the field of future educational research.

#### 2.2.7.4 Data Augmentation

Product augmentation will modify existing data to make the product suitable for education (Fadaee et al., 2017; Ratner et al., 2017). Scao and Rush (2021) found that adding instructions to a data set yielded an improvement in accuracy similar to adding hundreds of additional points to a classification task, and suggested that using instructions for the following activities is similar to additional information.

### 2.2.8 Challenges in Prompting

#### Although cue-based learning has achieved promising results in a wide range of tasks and situations, there are still some issues that need to be addressed. Some of these challenges are discussed below.

#### 2.2.8.1 Prompt Design

While most current academic research focuses on text classification or generation tasks, little attention is paid to data retrieval and text analysis. This may be because it is difficult to generate cues for such tasks. In the future, it will be necessary to modify these studies according to the classification or design problem, or to use good engineering response to present the design in the appropriate article. Another difficulty is; How to present information such as trees, graphs, tables or relational models in reports. Chen et al. (2021b) use additional symbols to encode lexical words, Aghajayan et al. (2021) use HTML-based guidelines to create web articles, but many models have not been thoroughly investigated. Additionally, there is also the question of how to search or learn the best combination of models and responses simultaneously, since the effectiveness of the model depends on both. While some studies focus on the selection of responses before models ( Gao et al., 2021 ; Shin et al., 2020 ), Hambardzumyan et al. (2021) demonstrated the ability to learn both simultaneously.

#### 2.2.8.2 Answer Engineering

Field engineering faces two major problems in the distribution of water. First, it can be difficult to choose the appropriate area when there are many groups. Second, when multiple token responses are used, it is not clear how to identify multiple tokens using language models, although some methods have been proposed (Jiang et al., 2020a). Appropriate responses for writing tasks can be equally valid but syntactically different, and most studies have focused on using a single response with only a few exceptions (Jiang et al., 2020c). How to achieve a better learning process through the many uses in text study is still an open question.

#### 2.2.8.3 Selection of Tuning Strategy

There are many ways to fix inconsistency of snapshot, language, or both. However, there is no good understanding in the field about the change in these practices. Conducting similar research in pre-training and conditioning would help understand the balances between different strategies (Peters et al., 2019).

#### 2.2.8.4 Multiple Prompt Learning

Cue integration involves the use of multiple cues, increasing space and time complexity. There is no research on the extraction of information from different directions. Schick and Schütze (2020, 2021a, b) used an integrated model to collect big data to extract information from many instructions, but it is not clear how to choose the appropriate integration language or how to use learning integration in text work. Cue synthesis and parsing involves decomposing complex job inputs into many subcues, but it is not clear how to choose between them. As a general rule, cue decomposition is appropriate for token or delay prediction tasks, while cue combination is better for relational prediction tasks. Rapid development is limited by the length of ideas, so it is necessary to learn how to select and sequence the information presented. Finally integration, working on different tasks, names or languages ​​just takes time, but there is little research in this area. It is not clear how to generate different cues or how to regulate the interaction between them. There is no research on the extraction of information from different directions. Schick and Schätze (2020, 2021a, b) used an integrated model to collect large datasets to extract knowledge from various teaching methods, but it is not clear how to choose integration-worthy language design or how to use integrated learning. in text activities. Cue synthesis and parsing involves decomposing complex job inputs into many subcues, but it is not clear how to choose between them. As a general rule, cue decomposition is appropriate for token or delay prediction tasks, while cue combination is better for relational prediction tasks. Rapid development is limited by the length of ideas, so it is necessary to learn how to select and sequence the information presented. Finally integration, working on different tasks, names or languages ​​just takes time, but there is little research in this area. It is not clear how to create different cues or modify the interaction between them.

#### 2.2.8.5 Selection of Pre-trained Models

There are many pre-learned languages ​​available for cue learning, but it is not clear how to choose the best language. While there are suggestions on how to choose a prior model for different NLP tasks, there is little comparison of the results of cue-based learning for different language models.

#### 2.2.8.6 Theoretical and Empirical Analysis of Prompting

Although time-only learning is successful in many cases, this approach lacks theoretical analysis and guarantees. Wei et al. (2021) showed that a small edit can reduce the biases required for retrieval in text classification tasks and facilitate the extraction of specific intended information. Sanghi et al. (2021) determined that the text classification task could be modified into a sentence completion task by modeling the language before expressive training. Scao and Rush (2021) found that inspiration often has hundreds of new products in the business section.

#### 2.2.8.7 Transferability of Prompts

It is important to understand which cues are specific to a pattern and to improve adaptation of the cues. Perez et al. (2021) found that cue selection in a calibrated few-shot learning scenario (in which a large validation set for selection cues is available) generally performs well on similar models, whereas in a true few-case learning scenario (in which a large validation set for selection cues is available in only a few training samples) similar samples It does not provide a good generalization for the change in cues was not good when sample sizes differed between the two conditions.

#### 2.2.8.8 Combination of Different Paradigms

Instruction-based learning has been successful in part due to the use of learning models that have been previously developed for training and optimization, such as BERT. However, it is unclear whether the pretraining effects of pretraining and conditioning can be applied directly to sign learning or whether pretraining is necessary to improve accuracy or ease of use. to work. This is an important research question but has not been extensively explored in the literature.

#### 2.2.8.9 Calibration of Prompting Methods

Calibration refers to the model's ability to make good predictions. When using the probability generated from a previously learned language model to predict the answer, it is important to consider that the distribution may not be well calibrated. Jiang et al. (2020b) found that the results of the preliminary model for the QA study measured well, but Zhu et al. (2021) identified three pitfalls (usually wrong form, wrong form, and wrong label) that can cause pre-learned language models to provide fast answers but be unfair for some answers. To overcome these pitfalls, Zhao et al. (2021) used a random-access point to obtain the initial probability and used the real input to obtain the second distribution, which could be used to obtain the probability distribution. However, this approach has the overhead of finding appropriate context-independent input, and the possibility of previously learned language patterns has not yet been evaluated. Although the ratio is measurable, caution should be taken when assuming a golden answer to the strategy because different data for the same product will compete for lack of quality (Holtzman et al., 2021). To solve this problem, answer engineering can be done, creating a set of golden answers using the annotation method (§5.2.2), or first evaluating the likelihood of words based on them in the given context (Holtzman et al., 2021).

## 2.3 Summary

# A lot of work has been done in the field of automatic story generation. However, most studies use the previous generation GPT2 as the model before story generation training. A few of them work using the power of a next-generation language model such as GPT3 or its derivatives.

# Although just-in-time learning is a relatively new phenomenon, significant research has been conducted in this area. Just-in-time learning has been used in many NLP applications, including writing. However, studies that use real-time learning to complete the story generation task are lacking.

# This work attempts to fill the above gap by using real-time learning to create stories and contribute to scores using next-generation language models.

# CHAPTER 3: RESEARCH METHODOLOGY

## 3.1 Introduction

## This project focuses on text generation and the various learning capabilities of PLM. Given an explanation as a control, the model should generate a story that exists in the model.

## 3.2 Algorithms & Techniques

3.2.3 State-of-the-Art Language Models

Some of the latest State-of-the-Art Pre-trained Language Models available for text generation tasks are discussed below:

* GPT3
* OPT
* BLOOM
* LaMDA
* PaLM

3.2.3 n-Shot Learning

* Zero-Shot Learning
* One-Shot Learning
* Few-Shot Learning

## 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.

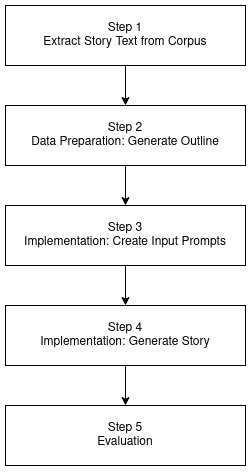


Figure 3.3.1.1 E2E Pipeline

3.3.2 Data Selection

This work makes use of two standard story generation datasets:

* **ROCStories**
* **WritingPrompts**

The datasets are used as a corpus for stories. This corresponds to Step 1 in Figure 3.3.1.1.

### 3.3.3 Data Preparation

The plan should be an example of a sentence. While sending articles and sharing written information via ROCStories, no information is prepared. For this reason, the process needs to be extracted from the story data set and then displayed on the link. These descriptive sentence pairs can be used as examples in many conversations.

The outline instances can take one of two forms:

* **Summary** – An example of the structure here is the summary of the sentence. This article has been expanded from the abstract. For content extraction, it is recommended to use TextRank (Mihalcea and Tarau, 2004) to extract the most informative words from the sentence.
* **Keywords/Keyphrases** - Examples of the structure here are the main subject and the expressions in the sentence. Articles consist of these keywords/keywords. For the extraction process, it is recommended to use RAKE (Rose et al., 2010) to remove important words from the sentence.

This corresponds to Step 2 in Figure 3.3.1.1.

### 3.3.4 Implementation

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

1. Creating prompts for Few-Shot Learning – In this step, information is generated about a small set of sample pairs. Each example has a outline (o) and a paragraph (t).

This corresponds to Step 3 in Figure 3.3.1.1.

1. Use examples of verbs with several conjugations to create incomplete sentences for new verbs; Verbs, examples with few conjugations, and sample questions are sent to the sample channel as emotional input. This formula returns the sentence generated from the query as a prediction.

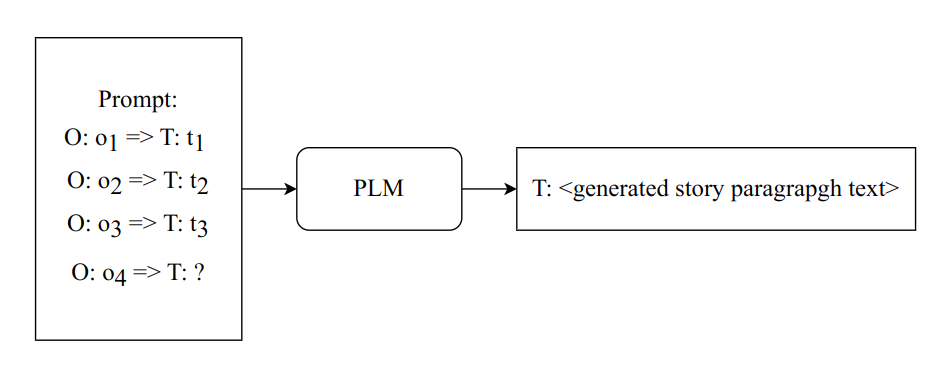


Figure 3.3.4.1 Prompting for Story Generation

This corresponds to Step 4 in Figure 3.3.1.1.

### 3.3.5 Evaluation

The resulting stories will be evaluated with various measurements. This study focuses only on evaluation using automatic indicators. Book review of the resulting stories is beyond the scope of this study.

The proposed metrics for evaluation are:

* **Perplexity (PPL)**
* **DIST/distinct-n**
* **BLEU**
* **Self-BLEU**
* **ROUGE**

These metrics will be compared across different factors.

Based on length of generated story -

* Short-Form Story Generation
* Long-Form Story Generation

Based on Pre-trained Language Model used:

* GPT2
* GPT3
* BLOOM
* OPT

Based on the Inference type:

* Zero-shot Generation
* One-shot Generation
* Few-shot Generation

This work will be benchmarked against the following baselines:

* Outline-to-Story
* Summarize, Outline and Elaborate
* Prompt Transfer for Text Generation

This corresponds to Step 5 in Figure 3.3.1.1.

## 3.4 Tools

## 

## 3.5 Summary

# 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.

# APPENDIX A: RESEARCH PLAN

# APPENDIX B: RESEARCH PROPOSAL

# APPENDIX C: ETHICS FORMS