NARRATOLOGY AND FANFICTION

Sri Sanjeevini Devi Ganni

A THESIS

in

Data Science

Presented to the Faculties of the University of Pennsylvania in Partial Fulfillment of the Requirements for the degree of Master of Science in Engineering

2021

Prof. Chris Callison-Burch
Supervisor of Thesis

Prof. Susan Davidson
Graduate Group Chairperson

ACKNOWLEDGEMENT

I would like to thank Prof. Chris Callison-Burch for supervising the thesis and helping me all along.

I would also like to thank Dr. Lara J Martin for her amazing support and assistance without which this thesis would not be possible.

Lastly, I would like to thank my family and friends for their constant encouragement and support.

ABSTRACT

NARRATOLOGY AND FANFICTION

Sri Sanjeevini Devi Ganni

Chris Callison-Burch

Narrative shapes our perception of the world. Narratology deals with the structure and function of narrative. Understanding narrative and predicting future narrative is an interesting area of Natural Language Processing. In this thesis, we aim to study narrative and predict future narrative in the domain of Fanfiction. Harry Potter Fanfiction uses characters, locations and expressions from the original work of Harry Potter books. The dataset consists of stories written by enthusiasts of the HarryPotter books. It is a rich collection of text specific to the Harry Potter domain.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

Chapter 1

Introduction

Narrative is a sequence of events recounted by a narrator to a narratee. For a given chain of events, narrative is fluid and alters with change in point of view or the point of time of narration. Switching the perspective or choosing events to exclude or include can drastically change the narrative. Narratology is the study of the narrative structure. A narrative is tied together by correlating events and interacting characters.

Narrative understanding and Narrative generation are interesting areas of research. Subsequent events in a narrative are usually related by a common motive or logical sequence or as an interaction or as a reaction to an external agent. Hence, without an underlying plot, story telling can be complex and be affected by a multitude of factors. This makes automatic narrative generation challenging.

Extracting event chains from a narrative helps depict the story of a character(agent or object) in the story. Narrative event chains are partially ordered set of events related to a character in the story. Story of a single character, does not direct the narrative. But, multiple event chains can be joined together to create a narrative schema (Chambers and Jurafsky, 2009). A Narrative schema is a set of typed narrative chains, that is not specific to one actor. Typed narrative chains consist of partially ordered events and their likely participants. Hence, narrative schema can help model a narrative discourse.

A typical narrative schema for the event 'arrest', would involve the Police or a government authority to arrest and then charge the defendant. This will be followed by the defendant pleading guilty or innocent. Then the suspect will be convicted or sentenced by the judge or jury.

In general, stories tend to have other intermediate steps or obstacles faced. A narrative tends to be way more dynamic and descriptive compared to a fixed schema. Hence, we try to model our problem as text generation task. In this work we attempt to predict the future events based on past events with the help of a pre-trained transformer based language model like GPT-2 (Radford et al., 2019). Transformer based language models have been popular lately for seq2seq and text generation tasks.

Language models for text generation are probabilistic and the resulting stories might not be coherent. The generated text is usually grammatically correct and gives the impression of the target domain but fails to follow a logical sequence of events. The future events can be predicted better if we know the intention of the characters and the rationale behind their actions. To understand the implicit knowledge in a event we use COMET (Bosselut et al., 2019), a language model that generates inferred commonsense knowledge graphs.

The idea behind this thesis is to be able to develop a model that understands the story and predicts the future narrative. The model should be able to identify the events in a narrative and also predict the future events that would eventually follow.

In the story generation world, there are two broad interpretations- open world and closed world. In closed world, a statement that is not explicitly known to be True is considered False. That is if the new statement does not fit in with the previously defined rules, it is considered False. On the Contrary, in the Open world the assump-

tion is True unless it is explicitly mentioned to be False. We also attempt both open world story generation and closed world story generation.

Unless it is explicitly mentioned, the usage of the word Character refers to an agent or object in the story. The words Character, Agent, Protagonist, Actor and Participant are used interchangeably through out the document and they all refer to the same entity.

1.1. Dataset

This work in particular focuses on the Harry Potter fanfiction stories. Our dataset is fanfiction stories collection annotated with AllenNLP (Gardner et al., 2017). AllenNLP is a platform that has implementation approaches for different Natural Language understanding tasks.

Fan fiction is great resource for domain specific, character rich collection of stories. The characters in fanfiction follow the rules of the original Harry Potter story world. But, their character relations, importance and perception differ from the original work (Milli and Bamman, 2016). Sometimes fan fiction works transgress boundaries of the original world and the characters are highly influenced by their own interpretation of the original work (Thomas, 2011). This makes the plots of fanfiction different and diverse. They explore different interpersonal relations and different goals. Like, one fanfiction story might be about an orphan muggle girl making it in the wizard world, another one can be about a love affair between Draco Malfoy and Ginny Weasley, two characters that are considered rivals in the original work.

1.2. Document Structure

The rest of the thesis is structured as follows:

1. Chapter 2 contains a literature review of relevant works to the thesis.
2. Chapter 3 deals with understanding narrative. Creating Narrative chains to understand the story of a character.
3. In Chapter 4, we use the pre-trained language models like GPT-2 to generate the future events.
4. In Chapter 5, we propose a system for integrating the commonsense inferences into pre-trained language models.
5. Chapter 6 summarises the thesis and discusses future work.

Chapter 2

Related Work

2.1. Narratology

(Abbott, 2008) talks about the universality of narrative and how it shapes our life. It details how time, frequency, perception and other factors effect our understanding of a narrative discourse.

In (Montfort, 2011), Nick Montfort introduces Curveship, a framework for developing interactive fiction with narrative variation. For a set of events and characters, Curveship can retell the story by changing the order of events and focalising different characters. It simulates a world with characters, locations and objects and uses it to generate a story with high-level narrative parameters. Narrative style is dependent on which character the narrator is and also the point in time.

2.2. Events and Narrative Chains

In (Chambers and Jurafsky, 2008), Chambers. et. al, propose a three step method for learning narrative event chains. A narrative event is defined as tuple of verb or action and its dependency on the participant. They isolate chains of events(verbs) that correspond to a character with the help of coreference resolutions and partially order the connected events. They find the PMI(Pointwise Mutual Information) between

pairs of events and use it to build narrative event chains. They also introduce an evaluation method called the Narrative Close test. They remove one a narrative event from a chain and try to predict the missing event and its dependency. The predictions are ranked, and the rank of the expected event is averaged to get the final score. In (Chambers and Jurafsky, 2009), they extend this work to present an unsupervised system for learning narrative schemas. While narrative chains model the events of one agent in the narrative, the narrative schemas prototype the entire narrative. They are types narrative chains, where along with the event chains the role of the agent is specified.

Algorithms based on Associate Rule Mining and weighted set covers are used in (Belyy and Van Durme, 2020) to learn narrative chains. They conclude that using Associative Rule Mining to predict the missing event works better than the narrative cloze test. The authors in (Rudinger et al., 2015) use a variety of methods like ordered and unordered events, N-grams and count thresholds for calculating pointwise mutual information for a pair of narrative events. They develop NaChos, a python implementation for all the methods with different parameters for learning and evaluating narrative chains.

In (Martin et al., 2018), the authors define an event as a tuple representation of subject, verb, object and modifier. The different components of the event representation were extracted through dependency parsing. They deal with the assumption that a sentence may contain one or more events. They use these event representations to predict the future event with a recurrent multi-layer encoder-decoder network.

2.3. Interactive Story Generation

In (Yao et al., 2019), the authors propose an open domain story generation algorithm. Given a topic, the algorithm generates a set of key words or a storyline with the help of seq2seq model. The topic and the storylines were further passed through another seq2seq model to generate the story. The authors in (Fan et al., 2018) also use similar multi step Hierarchical Text Generation process to generate stories. They fuse two convolutional seq2seq model with multi headed attention to generate the stories.

(Rashkin et al., 2020) also focuses on story generation that is conditioned on outline. The outline consists of key characters and events. Rashkin et. al. propose a neural narrative model PLOTMACHINES that tracks the state information of the story. It uses the tracked state information to generate narrative. They also implement memory to maintain coherence across the story. In (Fang et al., 2021), the authors work on controllable story generation. Plot is generated from a prompt of cascading events. For the story generation, events-story pairs are finetuned on pre-trained language models like GPT-2 (Radford et al., 2019) with special tokens.

In (Martin et al., 2018) and (Martin, 2021), the authors works on causally consistent neural story generation in an open world. They use GPT-2 (Radford et al., 2019) for story generation and VerbNet to keep track of the physical state of the event. They use the state information to prune and select the generated story lines.

2.4. CommonSense Reasoning

ConceptNet (Speer et al., 2017) and ATOMIC (Sap et al., 2019) are both a rich source of knowledge graphs. ATOMIC is a knowledge graph for if-then commonsense reasoning. Given an event, ATOMIC will be able to predict the required precondi-

tions, reactions, intent and effects. Whereas ConceptNet connects words and phrases with labeled edges. Its knowledge graph focuses on commonsense meanings of words. In (Mostafazadeh et al., 2020), the authors introduce GLUCOSE, a commonsense knowledge base with Causal-Inference rules. It generalises the event to get rule based inferences.

COMET(Bosselut et al., 2019) (Commonsense Transformers for Knowledge Graph Construction) is a generative model for automatically constructing the knowledge base. It is trained as a masked language model on tuples in the subject, relation, object format. It learns to produce the object tuple given a subject and relation.

2.5. Fanfiction

In (Milli and Bamman, 2016), the authors explore how fan fiction differs from the original work with respect to characteristics of the story. They analyse the fan fiction data to investigate the differences in character emphasis and how gender is presented. They find that the fan fiction prioritizes the protagonists in the original work and has greater attention allocated to female characters.

In (Frens et al., 2018), the authors talk about distributed mentoring and lexical diversity in fan fiction works. Most of the fan fiction works are peer reviewed through distributed mentoring. There is difference in the lexical diversity of the first chapter to the subsequent chapters. Lexical diversity increases with increase in reviews for the previous works and also as author matures.

(Bamman et al., 2013) uses LDA to identify the personas of characters in movies.

Looks into studying the dynamic changing relationships between characters in stories over a period of time. This work is particularly interesting as it does not consider relations static and homogeneous(Sometimes love may lead to murder). The authors design Relationship Modeling Networks to learn relationship descriptors.

Chapter 3

Understanding the FanFiction Narrative

In this chapter, we want to understand the underlying narrative chains in the fan fiction stories. Also, we want to try predicting the future events based on narrative chains.

3.1. Narrative Chains

To understand the narrative in the Harry Potter Fan fiction stories, we start with extracting Narrative chains. Narrative event chains are set of events related to a character in the story. The major inspiration of extracting narrative chains comes from (Chambers and Jurafsky, 2008) and (Chambers and Jurafsky, 2009). The event chain extraction they define works with the assumption of Narrative Coherence. It states that verbs connected by coreferring arguments are more likely to participate in a narrative chains compared to those not sharing. Coreference resolution refers to the task of finding all the mentions or expressions related to one entity within the whole document.

(Chambers and Jurafsky, 2008) and (Chambers and Jurafsky, 2009) also use semantic role labelling to identify the participants and their corresponding events(verbs). In this thesis, we instead made use of dependency parsing as we discovered that it better

helps relate characters and their actions, specially in story lines(sentences) containing multiple agents. Dependency parsing analyses the dependency between words in a sentence with the help of tags.

The dataset is annotated with AllenNLP platform. It uses the algorithm in (Lee et al., 2018) for Coreference Resolution and (Dozat and Manning, 2017) for Dependency Parsing. Figure 1 is an example output for dependency parsing.

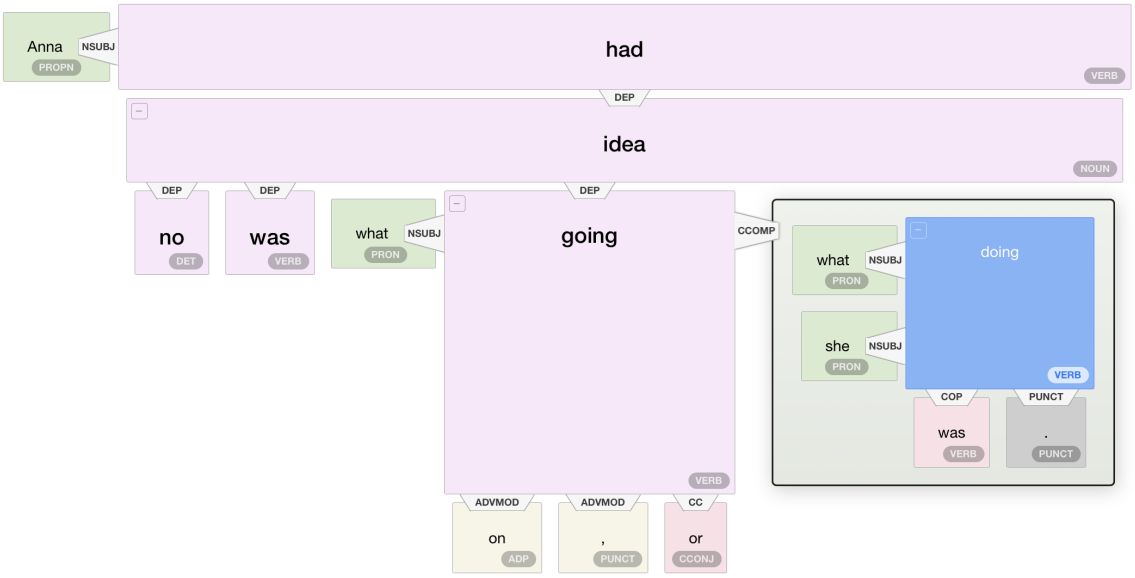

Figure 1: Dependency Parsing for "Anna had no idea what was going on or what she was doing." from AllenNLP Demo(Gardner et al., 2017)

In the sentence, "Anna had no idea what was going on or what she was doing.", the root dependency is the Verb "had". "Anna" is dependent on "had" through the dependency tag "nsubj". In corefernce resolution all mentions of an entity are clustered together. Each entity is tagged by a unique cluster number. Figure 2 is an example of Coreference Resolution for an example document from the fanfiction stories.

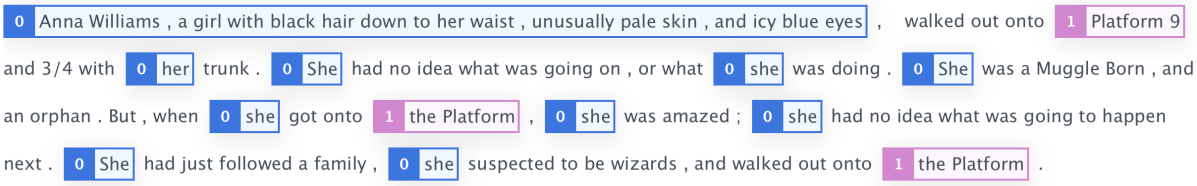

Figure 2: Coreference Resolution from AllenNLP Demo(Gardner et al., 2017)

To extract the event chains for an entity, we go through each of its mention and find the events. Since, narrative events are a tuple of verb and its dependency on the participant, we use dependency parsing to obtain the dependencies. Through dependency parsing we get the word the entity is dependent on (the parent of the word) as well as the dependency tag. Sometimes the parent turns out to be a complement or a helping verb. Hence, we obtain the grandparent(word the parent is dependent on) as well if it exists. We filter through the tags and match them to a predefined set of tags to get the action items. Figure 3 is an example of event chain of one of the characters from a fanfiction story. Here each event is represented with an action and its dependency on the character. Some lines have two events in them. The second event means that the grandparent dependency exists.

3.2. Predicting the Next Event

Using the event chains we generate ordered bigrams of events. That is each event in an event chain with every other event (in the order they occurred in the chain). From the event chain [(Follow, nsubj), (Suspect, nsubj), (Shock, nsubj)], we obtain the bigrams -

(Follow, nsubj), (Suspect, nsubj)
(Follow, nsubj), (Shock, nsubj)
(Suspect, nsubj), (Shock, nsubj)

Figure 3: Event Chains of a character

Using ordered bigrams will create partially ordered narrative chains. We use these bigrams to calculate the Pointwise Mutual Information(PMI) between sets of events. PMI is the logarithm of joint probability of the bigram divided by the individual probabilities. It calculates the measure of association between co-occurring events. We go through all the narrative chains created through the dataset and calculate PMI for all existing event pairs.

For our task of narrative completion we want to predict the next event given initial event or events. Let the initial set of events be $I = e_1, ..., e_k$. To guess the next event

we score all the events in the vocabulary and find the one with the highest score. To calculate the score of the event e, add the PMI of the event e with each event in the set I. In case, for an event e_i in I, if (e, e_i) has no occurrence in the PMI bigrams, penalise the score. This helps in removing improbable events.

In table 1, are the examples of narrative chains created. It has predictions of the next three events given a seed event.

Initial Events	*Predicted Events*
(follow, nsubj)	('have', 'nsubj'), ('look', 'nsubj'), ('say', 'nsubj')
('suspect','nsubj'), ('follow', 'nsubj')	('have', 'nsubj'), ('look', 'nsubj'), ('say', 'nsubj')
('love', 'nsubj')	('say', 'nsubj'), ('want', 'nsubj'), ('think', 'nsubj')
('love', 'nsubj'), ('propose', 'nsubj')	('want', 'nsubj'), ('widen', 'poss'), ''
('love', 'nsubj'), ('propose', 'dobj')	('say', 'nsubj'), ('want', 'nsubj'), ('think', 'nsubj')
('think','nsubj')	('want', 'nsubj'), ('widen', 'poss'), ('try', 'nsubj')
('think','nsubj'), ('interrupt', 'dobj')	('try', 'nsubj'), ('want', 'nsubj'), ('widen', 'poss')

Table 1: Examples of Narrative Chains

3.3. Analysis

The generated event chains as seen in figure 3 are not specific. Most of the actions repeat like $Born \rightarrow nsubj$, $sigh \rightarrow nsubj$, $say \rightarrow nsubj$. The fanfiction articles often do not posses highest quality of writing in terms of lexical diversity or sophistication ((Thomas, 2011) (Frens et al., 2018)).

Here are the observations on Predicted Events as seen in the table:

1. The predicted narrative chains are generalised. They are heavily dependent on co-occurrence of events, with some pairs being tightly coupled.((follow, nsubj) has high PMI with ('have', 'nsubj'), ('look', 'nsubj'), ('say', 'nsubj'))

2. Addition of an event or changing the dependency on the event does not make much difference in the predicted event chains. (('love', 'nsubj') and ('love',

'nsubj'), ('propose', 'dobj') have the same event chains)

3. There is no comprehensible rationale behind few event chains. We can not make sense of some of the predicted events like ('think','nsubj'), ('want', 'nsubj'), ('widen', 'poss'), ('try', 'nsubj'). In the story, "Anna thinks about an ice-cream. She wants an ice-cream. She widens her eyes. She tries.", 'widen' does not seem to fit in the context.

Chapter 4

Generating Narrative

Recently, large scale language models like GPT-2 (Radford et al., 2019) have gained widespread attention. GPT-2 is a Transformer model trained on millions of webpages and is efficient in generating coherent text. In this chapter we describe finetuning GPT-2 for generating narrative and future events.

4.1. Event Generation

In the previous chapter, the obtained narrative chains were generic. They were heavily dependent on co-occurrence of events. The events were a tuple of verb and dependency and belong to a particular participant in the story. In an interactive world this information is not enough to predict the next event. The generative language models are robust and work better with more information. Hence, we use GPT-2 to predict future events and adapt the event representation to include more arguments(complete sentences).

We modify the event definition to include a four tuple representation as in (Martin, 2021). Each event tuple has a subject, verb, object and modifier extracted through dependency parsing. Modifier can be propositional object or indirect object or complement. If the sentence misses an object or modifier, "EmptyParameter" is used in the position. For the sentence, "Harry loves Ginny", the event representation would be $< Harry, love, Ginny, EmptyParameter >$.

4.1.1. Experiments

We use the medium sized GPT-2 model that has 345 million parameters. The data is roughly split into 90-10 split for Training and Validation data. We Also held out around 1 percent of the data for test set. The events are finetuned for 3 epochs with a block size of 512.

The perplexity on the Validation data: 2.9922

Below are the events generated for an initial prompt event. All the prompts that were used are from the held out test data.

Prompt 1:

$< Draco, have, problem, EmptyParameter >$

$< Draco, got, idea, EmptyParameter >$

Generated Event Chain 1:

$< Draco, have, problem, EmptyParameter >$

$< Draco, got, idea, EmptyParameter >$

$< Draco, do, EmptyParameter, EmptyParameter >$

$< Draco, came, EmptyParameter, EmptyParameter >$

$< Draco, love, Draco, EmptyParameter >$

Prompt 2:

$< Draco, have, problem, EmptyParameter >$

$< Draco, got, idea, EmptyParameter >$

$< Draco, realised, EmptyParameter, EmptyParameter >$

Generated Event Chain 2:

$< Draco, have, problem, EmptyParameter >$

$< Draco, got, idea, EmptyParameter >$

$< Draco, realised, EmptyParameter, EmptyParameter >$

$< Draco, raised, eyebrow, EmptyParameter >$

$< Draco, smiled, EmptyParameter, EmptyParameter >$

$< Draco, gave, look, EmptyParameter >$

$< Draco, smiled, EmptyParameter, EmptyParameter] >$

Prompt 3:

$< Albus, feel, good, EmptyParameter >$

Generated Event Chain 3:

$< Albus, feel, good, EmptyParameter >$

$< Albus, cursed, EmptyParameter, EmptyParameter >$

$< Albus, retreated, EmptyParameter, EmptyParameter >$

$< Albus, took, lunch, EmptyParameter >$

$< Albus, had, take, EmptyParameter >$

$< Sadie, beginning, think, EmptyParameter >$

4.1.2. Analysis

Even though the perplexity on the validation set is low, the generations have repetitions in them. As seen in Generations 1 and 2, increasing the Prompt does not seem to have made much difference in the quality of generations.

In generation 3, we can see that 'feels good' is followed by 'cursed', which is followed by retreating and getting lunch. The chances of these events happening in succession are very low. Even in the first generation, idea is followed by doing something which is followed by Draco loving himself.

4.2. Story Generation

Given a part of the story, we want to predict what a particular character would do next. The input will be the first few sentences of the story and the character to predict the story on. The output will be what the character would do next. GPT-2 is trained on huge diverse corpus of data, which makes it ideal for text generation.

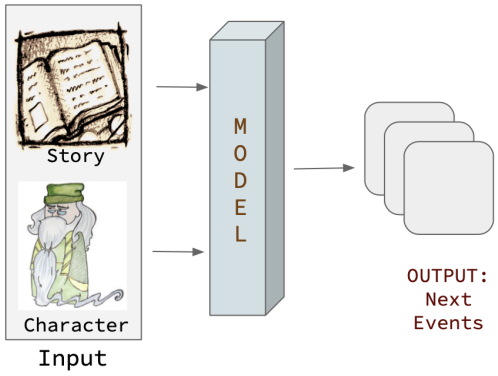

Figure 4: Problem Illustration for Story Generation

To finetune stories on GPT-2, the dataset was modified according to the task. The main subject in the story line was identified and coreference resolution was used to replace pronouns and other mentions with proper nouns.

4.2.1. Experiments

We finetuned the medium sized GPT-2 model, for 3 epochs with a block size of 1024.

Perplexity on Validation set: 16.2914

Below are the stories generated for a given story prompt.

Prompt 1:

Severus:Severus shouted as he flicked his wand and passed out books to the class of Gryffindors and Slytherin 's .
Severus:" today we will be doing theory work " he said as he started to write on the board behind him .

Generated Story 1:

Severus:Severus shouted as he flicked his wand and passed out books to the class of Gryffindors and Slytherin 's .
Severus:" today we will be doing theory work " he said as he started to write on the board behind him .
Severus:He paused to grab the back of his trousers and wrote on the board under his clothes saying the next class would be Hufflepuff theory, the next class will be Ravenclaw theory and the next class will be Hufflepuff Charms.

The songs he played with the same cadence as James were playing in his head.

Prompt 2:

Severus:Severus shouted as he flicked his wand and passed out books to the class of Gryffindors and Slytherin 's .

Generated Story 2:

Severus:Severus shouted as he flicked his wand and passed out books to the class of Gryffindors and Slytherin 's .
Severus:He woke up with a start as he felt his legs slumping.
It was the fourth class, and he remembered to have extra practice.
Severus:He stumbled over to his bed and fell asleep.

Prompt 3:

Emily Granger: Emily Granger held her daughter close to her chest as she stumbled through the dark woods .
path:The path that she had made slowly disappeared as she descended deeper into the woods .
wind:Gnarled and twisted trees groan as an icy wind tears away the last of autumn 's golden leaves .

Generated Story 3:

Emily Granger: Emily Granger held her daughter close to her chest as she stumbled

through the dark woods .

path:The path that she had made slowly disappeared as she descended deeper into the woods .

wind:Gnarled and twisted trees groan as an icy wind tears away the last of autumn 's golden leaves .

The warmth of Hermione heated Emily 's almost blue hands , that warmth wo n't be there for long though Emily thought to herself as she wiped away a warm tear that hung on her pale thin face .

Emily Granger:Emily Granger looked down upon the sandy ground in front of her. There was only the smidgen of a trickle of water on the ground and it covered a muggle child's hair. The path that she had made slowly disappeared as she descended deeper into the woods.

There was a deafening thundering as a lightning bolt shot through the air.

The mere sight of it brought all the little creatures that live on the planet into a frenzy .

4.2.2. Analysis

In the first generation, particularly, "He paused to grab the back of his trousers and wrote on the board under his clothes ", board and clothes seem to have some relation, which is not causally coherent. But, the generation does seem to get the theme of the prompt, that is the classroom. It uses different houses of the Harry Potter world and the subjects taught at Hogwarts.

The second generation does not seem to recognise the location of the original prompt, that is classroom. The generation alternates between bed and classroom.

The third generation is highly coherent compared to the previous generations. We

can say that the generations work better on scenic descriptions and the finetuned model does not seem to fully understand the rules of the Wizard world.

Chapter 5

Narrative Generation with Commonsense Knowledge Graphs

In this Chapter, we describe a novel method for integrating Story generation and Commonsense Reasoning. We introduce our system 'GPTCOM', that combines GPT-2(Radford et al., 2019) and COMET (Commonsense Transformers for Knowledge Graph Construction) (Bosselut et al., 2019).

In (Martin, 2021), the authors propose a casually consistent story generation pipeline ASTER-X. The pipeline has many components, starting with extracting events from the story and training a seq2seq model for story generation. With the help of VerbNet (Schuler, 2005), the authors kept track of the physical state of the event. This helped in selecting the next event and finally the events were converted to sentences. The story thus generated had poor readability, which prompted them to use GPT-2 for story generation. The story generation through finetuning GPT-2 had similar pipeline. The GPT-2 was used to generate sentences directly instead of an event-to-event generator. Through keeping track of the state and pruning, the next sentence was selected.

VerbNet (Schuler, 2005) gives the syntactic frame and the semantic arguments of a sentence. These can be considered physical aspects of an event and they do not consider the unsaid implicit inferences. Similar to the ASTER-X system proposed in (Martin, 2021), we use commonsense information for keeping track of the state.

Tracking state information is used to select the next story line.

5.1. COMET

COMET is a language model that constructs Commonsense knowledge graphs. Given an event COMET identifies the implicit conditions and abstractions of an event. In the normal world, when we narrate a story or even in a conversation there are a lot of sub-events that are inferred. These inferred events are not explicitly mentioned. The conditions and abstractions include the possible causes for the event, the effect on the subject or objects(others) and also the implicit attributes of the subject. For example, for the event "PersonX wants to see a movie", COMET predicts that the cause of the event PersonX wanted to be entertained and also that tickets and money were needed by PersonX. It also goes on to guess the after effects of watching the movie. The relations are fixed during training and the commonsense attributes are generated accordingly for each relation. COMET does a good job of understanding the implicit intent or cause in an event.

COMET is trained on both ATOMIC and ConceptNet knowledge bases. We use the model trained on ConceptNet data for our experiments. COMET has around 32 specified relations. We group the relations into 8 broader categories, namely *MadeOf, Location, Subevent, IsA, Desires, HasA, UsedFor, CreatedBy.* These relations also contain negative relations(IsNotA, NotHasA, etc).

5.1.1. Narration State

We define the State information to consist of Conditions and Restrictions. Conditions are the categorical information and restrictions are the negatory categorical

information stored for each character in the story.

The State Conditions for "Fred smiled weakly." are given in table 2:

Fred	MadeOf:	'happiness and happiness', 'emotion', 'pleasure', 'joy', 'happiness'
	Location:	set()
	Subevent:	set()
	IsA:	'facial expression', 'sign of happiness', 'sign of joy', 'expression of happiness', 'sign of pleasure'
	Desires:	'be with you', 'girl', 'please person', 'flirt', 'kiss'
	HasA:	'effect of make person happy', 'effect of relieve stress', 'effect of relieve tension', 'tooth', 'effect of make person feel good'
	UsedFor:	'show appreciation', 'show pleasure', 'flirt', 'show affection', 'show your appreciation'
	CreatedBy:	'light and air', 'smile', 'light and energy', 'light and light', 'light'

Table 2: State Conditions for "Fred smiled weakly."

The State Conditions for "Leaves littered the wet grey pavement as the wind billowed through the trees , shaking their crackling bounty of crimson , ginger and golden leaves ." are given in table 3

Leaves	MadeOf:	'water', 'grass', 'mud', 'flower', 'plant'
	Location:	set()
	Subevent:	set()
	IsA:	'sign of wet weather', 'result of rain', 'sign of wet season', 'scene of crime', 'sign of rain'
	Desires:	'lawn', 'dry'
	HasA:	'stain on your shirt', 'effect of make wet', 'stain on it', 'dandruff', 'stain on your pant'
	UsedFor:	'throw up', 'throw away', 'clean', 'clean up', 'clean up mess'
	CreatedBy:	'raindrop', 'water', 'droplet', 'rain', 'puddle'

Table 3: State Conditions for "Leaves littered the wet grey pavement as the wind billowed through the trees , shaking their crackling bounty of crimson , ginger and golden leaves ."

As we can see that the attributes for each category are a little noisy. For "Fred smiled weakly", the state correctly identifies it as a facial expression and sign of happiness. It also goes on to assume flirtatious behaviour and desires of being with a girl. The CreatedBy category is completely noisy.

In the state conditions for "Leaves littered the wet grey pavement as the wind billowed through the trees , shaking their crackling bounty of crimson , ginger and golden leaves", COMET interprets the sentence as sign of wet weather, result of rain as well as scene of crime. It also has noisy connotations for HasA category.

The attributes for each category are predicted using a language model which explains their noisiness. Hence, we store multiple attributes for each category to make the state intentions more inclusive.

5.2. GPTCOM Algorithm

We finetune GPT-2 with our fanfiction stories. This acts as a generative model for our proposed algorithm. The initial prompt is fed to the generative model to get the required generative sequences. We prune the generated sequences to get the next sentence in the story.

We track the state for prompt and for every sequence and compare them to check the validity. If there is more than one valid sequence, we pick one of the valid sequences at random. The two parameters of the algorithm are n and NS. NS is the number of sentences or story lines to generate. n is the number of sequences to generate from the finetuned GPT2 model for a given prompt. The Algorithm is given below:

Algorithm 1: GPTCOM Algorithm

```
Result: PROMPT
state = State() for i = 0 to NS do
    sequences = getGenerationsFromGPT2(n)
    validStates = []
    for seq in sequences do
        promptState = getState(PROMPT)
        seqState = getState(seq)
        valid, newState = checkStateValidity(promptState, seqState)
        if valid then
            validStates.append(newState)
        end
    end
    selectedState = random(validStates) /* Randomly select a valid state */
    state.update(selectedState)
    PROMPT.update() /* update to PROMPT + selected Sequence */
end
```

5.3. Open World and Closed World Interpretation

Checking the validity of the state, can be done in two different ways. One way, being very stringent is to check if there is an overlap in any of the categories of state. For an event to have valid co-occurrence with another event, it should have some characteristic in common. This way is analogous to the Close World representation. An event is only accepted if it meets any of the preconditions.

The second way is to check if any of the restrictions for one of the event overlaps with the conditions of the other. This is more flexible way of checking the validity and is similar to the Open world interpretation. Here, we reject a generation if and only if

it contradicts with the initial events(prompt).

We implemented both the Open World and Closed World interpretations and compared the results. Figures 5 and 6 refer to the sentence selection in Closed world and Open world systems.

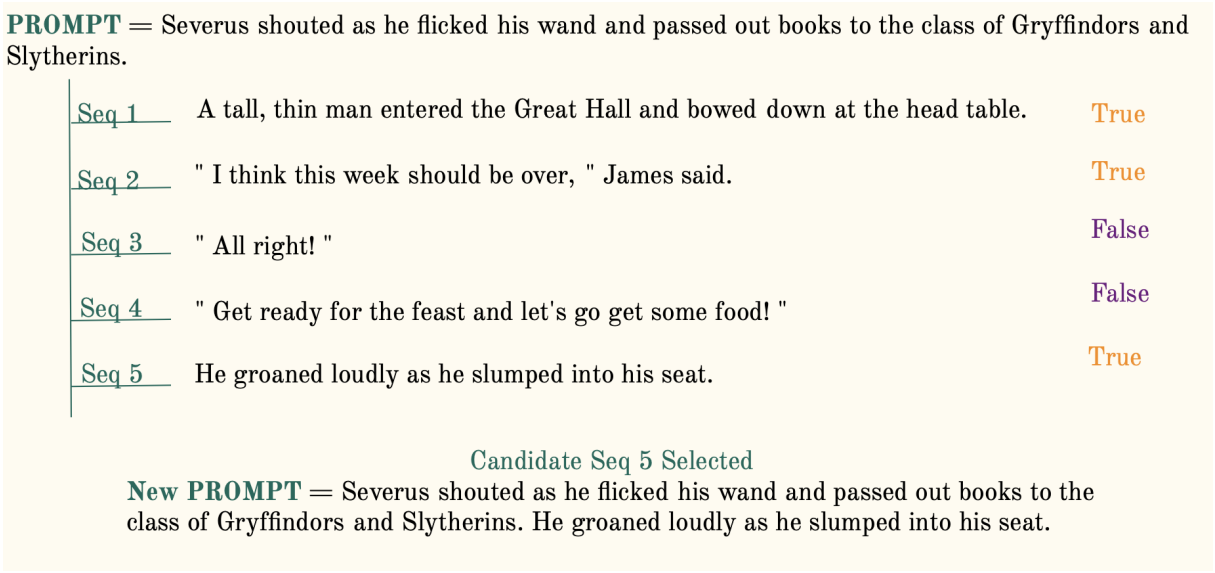

Figure 5: Generating next sentence in Closed World System

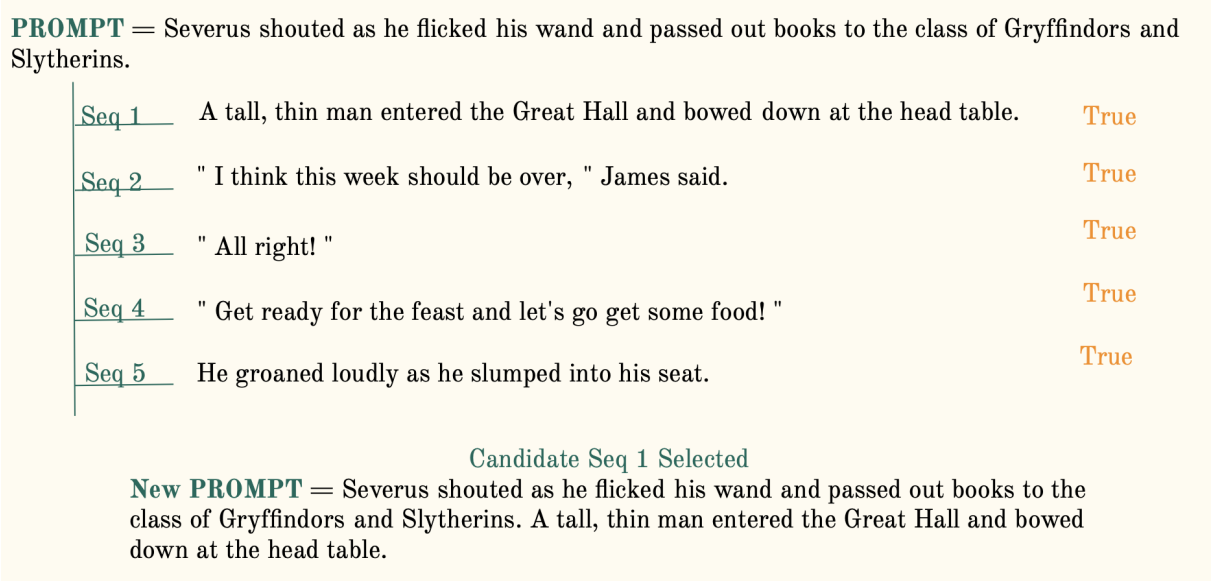

Figure 6: Generating next sentence in Open World System

A few examples of different interpretations for the same initial prompt are in figures 7 and 8.

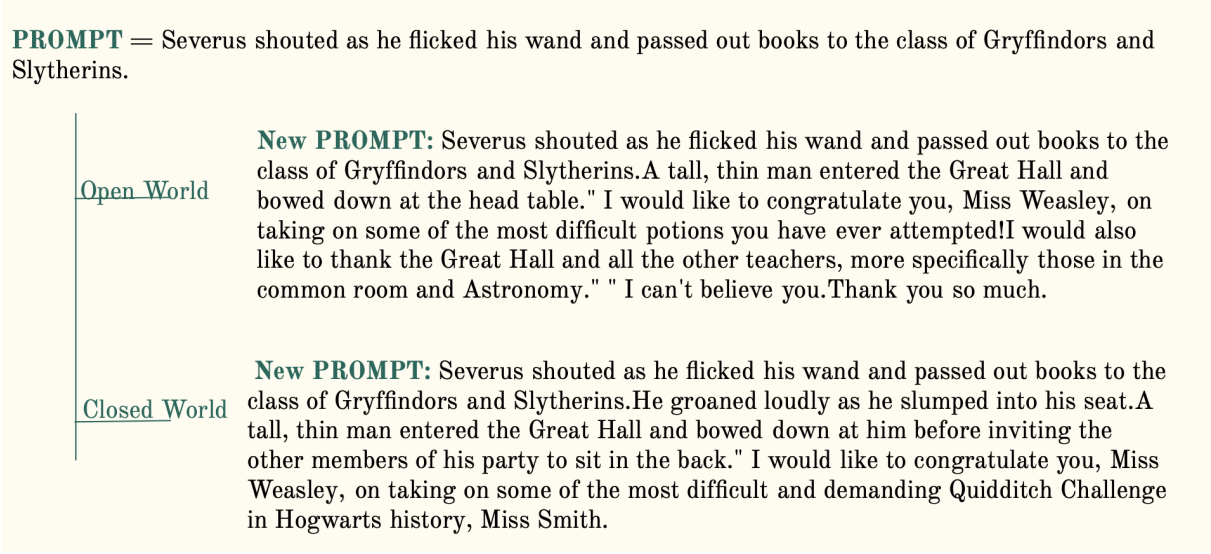

Figure 7: Example Generation with different interpretations

PROMPT = It was November and despite the cold rain outside, Albus was feeling good .

Open World — **New PROMPT:** It was November and despite the cold rain outside, Albus was feeling good .A little bit.A lot.A lot.A lot.A little bit.

Closed World — **New PROMPT:** It was November and despite the cold rain outside, Albus was feeling good .A little bit." I think I 'll leave the house for a bit. "Albus wrote to Sirius to let him know he was going out with another girl.A month later, Albus was still acting like a bloody idiot.He was excited to get married to Hermione's sister, Adelina.

Figure 8: Example Generation with different interpretations

5.3.1. Analysis

The Open world allows for more exploration while the Closed world is less flexible. The former allows exploration but at the same time validate events that have nothing in common with the preceding event. This allows for repetition in the generated stories and also the plot may not be consistent.

As seen in figures 5 and 6, the closed world does a better job of validating the generations. The open world accepts all the generations since they do not openly contradict with the initial prompt. The possible location for the prompt in the example would be a classroom. By observation we can say that Sequence 2 and Sequence 5 are valid candidates. The closed world also seems to accept Sequence 1 whose location is Great Hall. It rejects Sequence 3 and Sequence 4 because they return empty states.

The generations in open world are prone to repetitions. In 8, the generated text keeps on repeating "A little bit". Whereas, the closed world story makes sense and

the events in a plausible order.

The example in figure 7 has almost same generations for both the interpretations. The closed world generation is of little better quality.

Here is an example of rejecting generations in GPTCOM:

For the prompt:
Leaves littered the wet grey pavement as the wind billowed through the trees , shaking their crackling bounty of crimson , ginger and golden leaves ." Of course, Erin." Erin groaned, furrowing her brows as she stared at the bright green poster in the dimly lit corner of the corner shop.She groaned again as she read the name – " Erin " – and felt something scrape her eyes.,
the state conditions are given in table 4.

Leaves	MadeOf:	'mud', 'water', 'grass', 'flower', 'plant'
	Location:	set()
	Subevent:	set()
	IsA:	'result of rain', 'sign of rain', 'sign of wet weather', 'sign of wet season', 'scene of crime'
	Desires:	'dry', 'lawn'
	HasA:	'stain on your shirt', 'stain on it', 'stain on your pant', 'effect of make wet', 'dandruff'
	UsedFor:	'throw up', 'throw away', 'clean', 'clean up mess', 'clean up'
	CreatedBy:	'raindrop', 'puddle', 'water', 'droplet', 'rain'
Erin	MadeOf:	'metal', 'paper', 'plastic', 'diamond', 'glass', 'information', 'word', 'cell', 'pen', 'fact'
	Location:	set()
	Subevent:	set()
	IsA:	'activity', 'sign of knowledge', 'sign of intelligence', 'way of think', 'way of communication', 'skill', 'function of eye', 'sign of intelligence', 'function of brain', 'information source'
	Desires:	'play', 'love', 'explore', 'write it down', 'find someone name', 'get information', 'get name', 'understand', 'find someone', 'explore universe'
	HasA:	'effect of make list', 'effect of reduce vision', 'effect of change person identity', 'lens', 'many function', 'many side', 'effect of confuse', 'two side', 'effect of change your identity', 'effect of recognition'
	UsedFor:	'focus', 'find information', 'find out information', 'see thing', 'look at thing', 'stare', 'find someone', 'find person', 'find out person identity', 'visualize'
	CreatedBy:	'computer'

Table 4: State Conditions

The state restrictions for the generated text:

Her head spun and she glanced down, aware of the pale green trail of destruction which once ran all the way to the entrance of the abandoned store.

are given in table 5.

Erin	MadeOf:	'metal', 'diamond', 'glass', 'gold', 'silver'
	Location:	set()
	Subevent:	set()
	IsA:	'signal', 'measurement', 'way of communication', 'way to communicate'
	Desires:	'go to movie', 'play card', 'play game', 'go to film', 'play tennis'
	HasA:	'effect of see through person', 'effect of see through thing', 'effect of see'
	UsedFor:	set()
	CreatedBy:	set()
Leaves	MadeOf:	set()
	Location:	set()
	Subevent:	set()
	IsA:	set()
	Desires:	set()
	HasA:	set()
	UsedFor:	set()
	CreatedBy:	set()

Table 5: Target State Restrictions for "Her head spun and she glanced down, aware of the pale green trail of destruction which once ran all the way to the entrance of the abandoned store."

GPTCOM validates this generation as False, due to conflicting attributes in Given State Conditions and Target State Restrictions. This example will be invalidated in both Open world and Closed world versions.

5.4. Experiments

We considered Closed world experiments for all the future experiments as it performs better than the Open world interpretation. We take the value of $n = 10$ and $NS = 6$.

GPTCOM has been designed to refine the generations of GPT-2. To check whether the GPTCOM algorithm has any significant improvements over GPT-2 we compare and contrast both the generations. The GPT-2 model is finetuned for 3 epochs and with a block size of 1024.

The examples of generations are given in figures 9

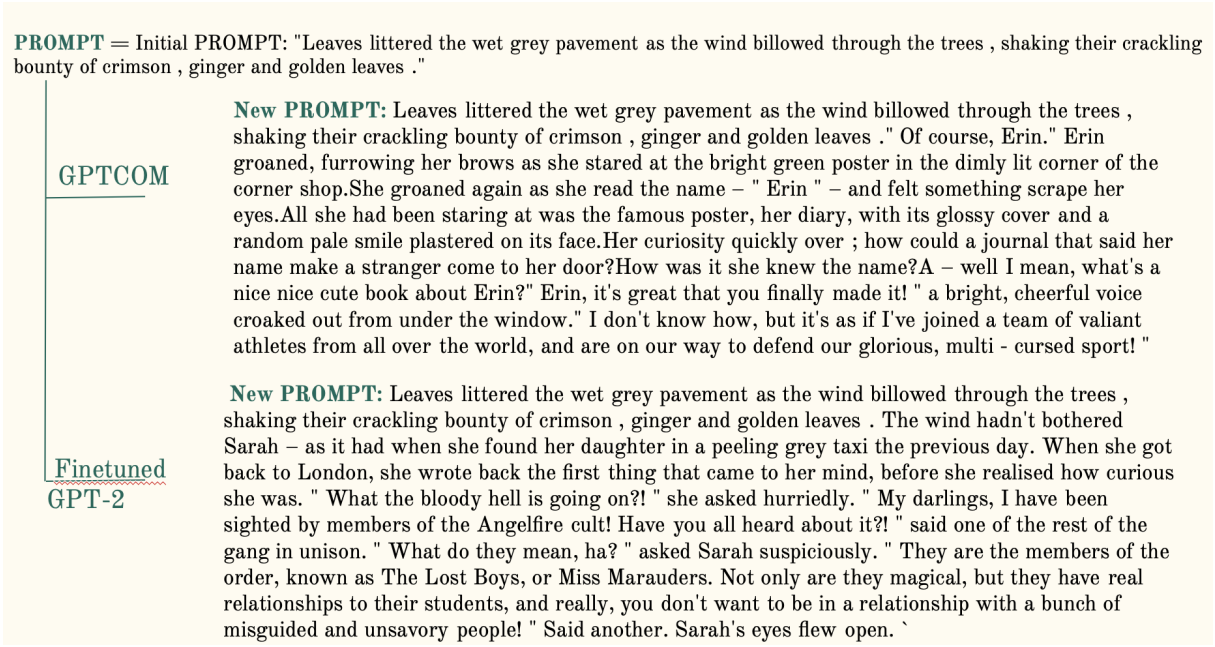

Figure 9: Example Generation with GPT-2 and proposed GPTCOM

5.4.1. Human Evaluation

We perform Human evaluation on the generations from GPTCOM and GPT-2. Similar to the evaluation in (Purdy et al., 2018) and (Tambwekar et al., 2019), we ask

human annotators to grade the generations on a 5 point scale for the following metrics.

1. This story exhibits CORRECT GRAMMAR.
2. This story's events occur in a PLAUSIBLE ORDER.
3. This story's sentences MAKE SENSE given sentences before and after them
4. This story AVOIDS REPETITION
5. This story uses INTERESTING LANGUAGE
6. This story is of HIGH QUALITY
7. This story is ENJOYABLE
8. This story REMINDS ME OF HARRY POTTER FAN FICTION
9. This story FOLLOWS A SINGLE PLOT.

We asked 12 human annotators who are part of the Computer and Information Sciences department, University of Pennsylvania to do the evaluation. They rated 6 story generation pairs(Finetunned GPT-2 and GPTCOM(closed world version)). The metrics were rated as Strongly Disagree, Disagree, Neutral, Agree and Strongly Agree. The metrics were then converted to numerical scale with Strongly Disagree being 0 and Strongly Agree being 4.

The results are in figure 10.

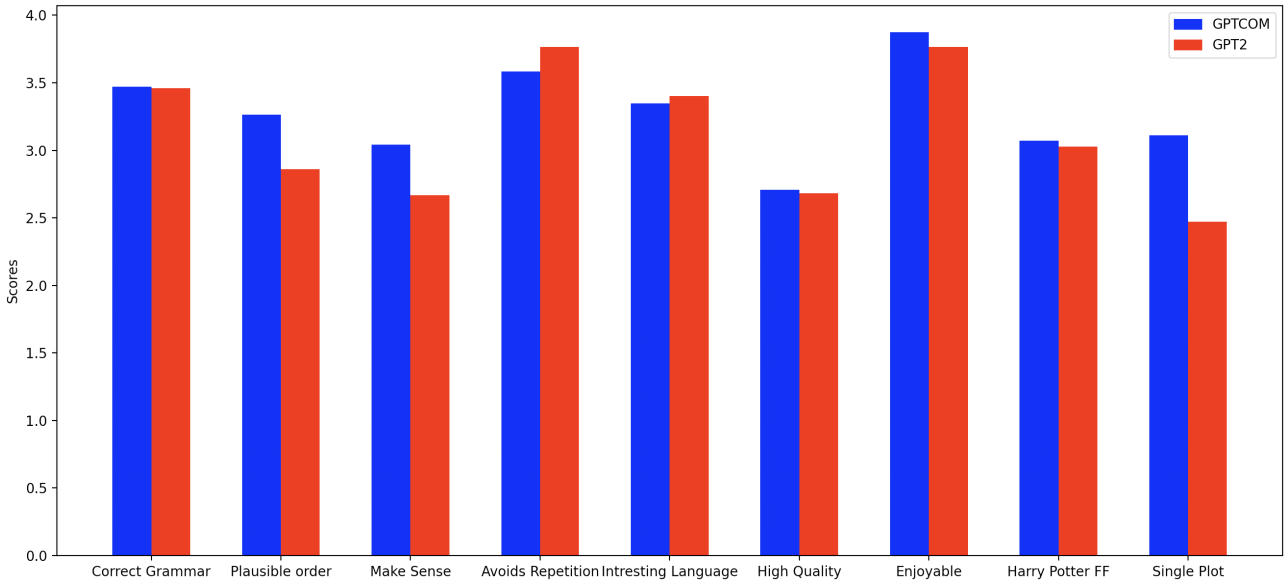

Figure 10: Evaluation

5.4.2. Analysis

GPT-2 and COMET perform similarly in most of the aspects. They both have the same generation model, that explains why they perform similar on metrics like grammar and interesting language.

GPTCOM outperforms GPT-2 on the metric Single Plot. This means that the GPT-COM is able to weed out some of the generations that do not make sense.

GPTCOM also performs slightly better on the metrics plausible order and makes sense. This could again be due to better pruning of generations by GPTCOM.

Chapter 6

Conclusion and Future Work

In this work we explored different ways of predicting the narrative in the Harry Potter world. We started with constructing the narrative event chains and using the event co-occurrences in a chain to predict the future events. We then use pre-trained large scale language models for event generation and story generation. The generated events and stories were not coherent and were prone to repetitions.

We then introduce a original system GPTCOM for integrating pre-trained language models and Commonsense Inferences. We use State tracking to prune the generations of the finetuned language model to create causally consistent narratives. The proposed system performs better than the language model in terms of plot consistency.

This work can be extended to include the physical aspects of the world as in (Martin, 2021) using VerbNet (Schuler, 2005). We believe that it would track the physical aspects and improve generations. It would also be interesting to analyse the desires and other attributes of the characters in the fanfiction stories through COMET. It would give us insight into what goals the fanfiction writers prioritize and how they perceive the characters in contrast to the original work.

BIBLIOGRAPHY

H. P. Abbott. *The Cambridge introduction to narrative*. Cambridge University Press, 2008.

D. Bamman, B. O'Connor, and N. A. Smith. Learning latent personas of film characters. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 352–361, 2013.

A. Belyy and B. Van Durme. Script induction as association rule mining. In *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events*, pages 55–62, 2020.

A. Bosselut, H. Rashkin, M. Sap, C. Malaviya, A. Celikyilmaz, and Y. Choi. Comet: Commonsense transformers for automatic knowledge graph construction. *arXiv preprint arXiv:1906.05317*, 2019.

N. Chambers and D. Jurafsky. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, 2008.

N. Chambers and D. Jurafsky. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 602–610, 2009.

T. Dozat and C. D. Manning. Deep biaffine attention for neural dependency parsing. *ArXiv*, abs/1611.01734, 2017.

A. Fan, M. Lewis, and Y. Dauphin. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 889–898, 2018. URL `https://arxiv.org/pdf/1805.04833.pdf`.

L. Fang, T. Zeng, C. Liu, L. Bo, W. Dong, and C. Chen. Outline to Story: Fine-grained Controllable Story Generation from Cascaded Events. 2021. URL `http://arxiv.org/abs/2101.00822`.

J. Frens, R. Davis, J. Lee, D. Zhang, and C. Aragon. Reviews matter: How distributed mentoring predicts lexical diversity on fanfiction. net. *arXiv preprint arXiv:1809.10268*, 2018.

M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. F. Liu, M. Peters, M. Schmitz, and L. S. Zettlemoyer. Allennlp: A deep semantic natural language processing platform. 2017.

K. Lee, L. He, and L. Zettlemoyer. Higher-order coreference resolution with coarse-to-fine inference. In *NAACL-HLT*, 2018.

L. J. Martin. *Neurosymbolic Automated Story Generation*. PhD thesis, Georgia Institute of Technology, 2021.

L. J. Martin, P. Ammanabrolu, X. Wang, W. Hancock, S. Singh, B. Harrison, and M. O. Riedl. Event Representations for Automated Story Generation with Deep Neural Nets. In *Thirty-Second AAAI Conference on Artificial Intelligence*, pages 868–875, 2018.

S. Milli and D. Bamman. Beyond canonical texts: A computational analysis of fanfiction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2048–2053, 2016.

N. Montfort. Curveship's automatic narrative style. In *Proceedings of the 6th International Conference on Foundations of Digital Games*, pages 211–218, 2011.

N. Mostafazadeh, A. Kalyanpur, L. Moon, D. Buchanan, L. Berkowitz, O. Biran, and J. Chu-Carroll. GLUCOSE: GeneraLized and COntextualized Story Explanations. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, volume 4569–4586, 2020. URL `http://arxiv.org/abs/2009.07758`.

C. Purdy, X. Wang, L. He, and M. Riedl. Predicting Generated Story Quality with Quantitative Metrics. In *14th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2018)*, pages 95–101, 2018. ISBN 9781577358046. URL `https://aaai.org/ocs/index.php/AIIDE/AIIDE18/paper/view/18106/17228`.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

H. Rashkin, A. Celikyilmaz, Y. Choi, and J. Gao. PlotMachines: Outline-Conditioned Generation with Dynamic Plot State Tracking. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, 2020. URL `https://www.aclweb.org/anthology/2020.emnlp-main.349/`.

R. Rudinger, V. Demberg, A. Modi, B. Van Durme, and M. Pinkal. Learning to predict script events from domain-specific text. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 205–210, 2015.

M. Sap, R. Le Bras, E. Allaway, C. Bhagavatula, N. Lourie, H. Rashkin, B. Roof, N. A. Smith, and Y. Choi. Atomic: An atlas of machine commonsense for if-

then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035, 2019.

K. K. Schuler. Verbnet: A broad-coverage, comprehensive verb lexicon. 2005.

R. Speer, J. Chin, and C. Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14972.

P. Tambwekar, M. Dhuliawala, L. J. Martin, A. Mehta, B. Harrison, and M. O. Riedl. Controllable neural story plot generation via reward shaping. In *IJCAI*, pages 5982–5988, 2019.

B. Thomas. What is fanfiction and why are people saying such nice things about it?? *Storyworlds: A Journal of Narrative Studies*, 3:1–24, 2011.

L. Yao, N. Peng, R. Weischedel, K. Knight, D. Zhao, and R. Yan. Plan-And-Write: Towards Better Automatic Storytelling. In *AAAI Conference on Artificial Intelligence*, pages 7378–7385, 2019. URL https://aaai.org/ojs/index.php/AAAI/article/view/4726.