Story Planning: Unlimiting Creativity

Musha Wen (u5962473)

A report submitted for the course
COMP8755 Individual Computing Project
Supervised by: Dr Patrik Haslum
The Australian National University

May 2018

Except where otherwise indicated, this report is my own original work.

Musha Wen (u5962473)
25 May 2018

Acknowledgments

I would like to appreciate all the people who helped me during this project.

First and foremost, I would like to thank my supervisor, Dr. Patrik Haslum, who offer me the chance to do research in the field that I interested in. He guided me to accomplish this project, gave me suggestions and helped me when I faced difficulties.

I would also like to thank my parents, who afford the cost of my study in Australian National University. Whenever I felt struggled or dispirited, they always encourage me.

My thanks also to the lecturers and professors who taught me lessons during the past two years. They did their best on the courses they taught. These courses brought me deeper understanding of the Artificial Intelligence field. The knowledge I acquired from the courses forms the basic understanding of this project.

Finally I would like to thank all the people who helped me do the experiment to evaluate the outcomes of this project. They generously donated their precious time no matter they were busy or not. Their efforts significantly increase the reliability of the evaluation.

Abstract

Story generation has been a very popular topic in the field of Artificial Intelligence. Researchers have made efforts to think of the methodologies for solving this problem. One of the traditional approaches, the planning-based approach has produced many convincing outcomes. However, these planning approaches have their own limitations. On one hand, they involve a large amount of human work to build a model for planners. On the other hand, those human-made models lack unbounded creativity. This report introduces an automatic approach to establish the story model with the help of Concept Net and Brown Corpus. It aims to acquire reasonable story model with unlimited creativity by using existing theories of planning-based approaches and knowledge of different datasets.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter gives an overview of the problem we addressed in this project. Description of problem, motivations and the scope of this project will be covered, in order to help the reader have a general understanding of the problem.

1.1 Problem Statement

There has been a long time since people started to consider about how to write a good story. The study on this topic can date back to the time of Aristotle [Sayers, 1936]. With the development of artificial intelligence, researchers began looking for solutions that can make machines write a piece of story automatically. Approaches to solve this problem can be roughly divided into two kinds: one is using machine learning techniques, the other is applying planning theories. In this report, we propose a methodology which can be used for addressing a problem that makes the machine automatically establish a huge story model with not only logic but unbounded creativity as well. This story model can be utilized by the planning-based approaches when solving story generation problems.

1.2 Motivations

Narrative generation has been the privilege of humans, especially writers, in a long history. People learn from the world, use their own real life experiences and together with humans' astonishing imagination, to create countless wonderful stories. Several decades ago, stories mainly came from human themselves. Some people write the stories and the others read and enjoy them. In recent years, the rapid development of computer techniques, especially the development of personal computers, leads to the growth of computer games. People still love stories, while this time it is the computer game that tells the story to people during the game. As computer games become more and more complex and realistic, a set of pre-defined stories are no longer sufficient to cover all the events (or stories) happening in the world of a game. The demand for approaches for automated story generation begins to emerge and we call this task the *narrative generation* problem. Machine generated stories are always not expected to be of the same quality as the ones made by a real person. However, some key

properties should still be satisfied: the story should be logical and realistic while a better story should also be creative and attractive. This suggests that whatever events are happening, movements of characters in the scene should have their reasons.

From a technical perspective, as previously discussed, using techniques based on machine learning to address story generation problem is also feasible. However, there also exists some limitations. According to [Goodwin and Sharp, 2016], plots generated by machine learning models are not coherent enough, which makes the story we get not meaningful enough.

On the other hand, some works already exist for automated narrative generation and can meet the basic requirement of logicality. For example, the pure planning approaches we introduced in the previous section. Nevertheless, the stories generated with these techniques usually lack creativity and therefore are not attractive enough. One of the convincing outcome generated by these approaches can be achieved in [Brenner, 2010]. We found that one possible reason which might leads to the shortage of creativity in this story plot is that it does not include failure of characters and does not provide as well as solve the conflicts among characters either. These problems can be settled with a larger story model since more events and more connections between the events can generate more creativity. Though a plot in a story is very similar to an event in planning, describing the story world is a great burden to the researchers and the project always ends with generating an uncreative story. Therefore, we aim to address the story model establishment problem so as to bring more creativity to the stories generated under our model.

1.3 Project Scope

In this project, we put forward a new approach on the basis of existing methods so as to gain a story model which can be used for settling story generation problems with planning theories. A story model can be regarded as a series of events that might happen in the story world together with some rules to connect these events together in order to keep the events happen in a reasonable order in the story world. This concept is concluded from the existing work on story models. There exists two famous story models, one is Plotto [Cook, 2011], the other is SceÌĄalextric [veale2017deja]. The events in the model we get at the end of this project are expected to be not only logical but also diverse enough to allow for creative story generation. This project aims to solve the story model generation problem with more creativity with the help of Concept Net [Liu and Singh, 2004] and Brown Corpus [Francis, 1965]. We use Concept Net to generate as many events as possible and find rules to connect these events together so as to get an event chain, which can also be regarded as a backbone of a story. Brown Corpus is used for a supplementary for the Concept Net. We extract information from that corpus by Natural Language Processing techniques. With knowledge we retrieved from these two datasets, we establish a story model with unlimited creativity that can be used for addressing story generation problems. Details about these datasets will be introduced in the next chapter.

1.4 Report Outline

There will be four chapters to introduce the approach we use in this project to solve the story model generation problem. Chapter 2 offers information which is helpful for understanding the method discussed in following chapters. Chapter 3 introduces concrete steps of what we do in this project and shows the consequence we get at the end of the project. Chapter 4 is about experiments we do to evaluate the results we got with brief analysis. Chapter 5 summarizes the outcome and discuss some future work we can do to improve our work. Chapter 6 gives a conclusion of this whole project.

Chapter 2

Background

This chapter mainly introduces the background of our project, which is helpful for understanding our work discussed in the next chapter. The first two sections offer a basic understanding of the datasets we used in this project. The third section covers the algorithms and libraries which help us achieve our project goal. The last section is a brief summary of this chapter.

2.1 Concept Net

Concept Net is a huge network, which contains millions of words as well as phrases from different languages. This knowledge is collected from a variety of resources [Liu and Singh, 2004], including the Wiktionary and the WordNet. This dataset originated from the crowdsourcing project Open Mind Common Sense, which was launched in 1999 at the MIT Media Lab [Liu and Singh, 2004]. With new sources added into the dataset, there appears several versions of Concept Net. We use Concept Net 5 in our project. For brevity, when we mention this version of the dataset in following chapters, we omit the version number.
The data structure of Concept Net can be abstracted as a huge directed graph. In this graph, each node represents a word or phrase. Each node n_1 is connected to another node n_2 if these two concepts can establish a connection via one of the relations defined by Concept Net. Therefore, the edge between any two nodes represents the relation that these two nodes satisfy. We can infer that this word net mainly offer us linguistic information about the relations between different words in different languages. With this information, we can easily find a word's reasonable successor from the linguistic perspective.
In this project, our work based on 46 relations of Concept Net. Among these relations, we focus on 10 relations. We provide the definitions [Liu and Singh, 2004] of these 10 relations so has to help the reader understand our approach introduced in the next chapter better.
HasPrerequiste In order for A to happen, B needs to happen; B is a dependency of A. *Example: dream → sleep*.
Entails If A is happening, B is also happening. *Example: run → move*.
Causes A and B are events, and it is typical for A to cause B. *Example: exercise →*

sweat.
HasSubevent A and B are events, and B happens as a subevent of A. *Example: eating → chewing*.
HasFirstSubevent A is an event that begins with subevent B. *Example: sleep → close eyes*.
HasLastSubevent A is an event that concludes with subevent B. *Example: cook → clean up kitchen*.
CreatedBy B is a process or agent that creates A. *Example: cake → bake*.
CapableOf Something that A can typically do is B. *Example: knife → cut*.
IsA A is a subtype or a specific instance of B; every A is a B. This can include specific instances; the distinction between subtypes and instances is often blurry in language. This is the *hyponym* relation is WordNet. *Example: car → vehicle; Chicago → city*.
MannerOf A is a specific way to do B. Similar to *IsA*, but for verbs. *Example: auction → sale*.

2.2 Brown Corpus

The Brown Corpus, prepared for linguistic research on modern English, was the first general corpus of texts that can be read by computer [Francis, 1965]. This corpus was firstly established in 1960s by Brown University. It contains 500 samples, each of which has over 2000 words and there are over 1 million words in total. The sentences in this corpus are roughly divided into 15 text categories, including reportage and reviews. Since the data is collected from such a wide range, we believe that using this corpus will provide information that is much closer to the daily life of humans. There are six versions of the Brown Corpus. We choose to use the original Form A. For brevity, we call this version of corpus as Brown Corpus in following chapters.

2.3 Natural Language Processing

Natural Language Processing (NLP) is an area in artificial intelligence which focuses on making computers understand human's natural language. A natural language processing system converts natural language into structured information which can be easily processed by computers. In this work, we utilize two key techniques from the NLP area: dependency parsing and part-of-speech tagging.

Dependency parsing is the process of translating the human natural language into a tree structure based on some predefined grammars. The dependency refers to the grammatical functions of words in the sentences. This is an well researched topic in the NLP area and several algorithms have been introduced by previous works. In this work, we will use the dependency parsing tool developed by Standford [Bauer, 2014] to extract the *Subject-Verb-Object* triples.

Part-of-speech (POS) tagging is another core technique in the NLP area. As the name suggests, POS-tagging algorithms aim to perform a grammatical analysis over words in natural languages based on the the word itself and the word's context.

A few models has been put forward for the POS-tagging task. For example the maximum cross-entropy POS-tagging proposed in [Yi, 2015] and some recent deep neural network based POS-tagging works can be found in [Popov, 2016; Zheng et al., 2013; Perez-Ortiz and Forcada, 2001]. In this work, we use the Maxnet POS-tagger provided in the Standford CoreNLP toolkit to validate the *Subject-Verb-Object* pairs we extracted from the real world texts.

2.4 Summary

In this chapter, we discussed the basic knowledge to understand our approach. We introduced two datasets which offer us data to generate events. Techniques we applied during our project were also briefly introduced in this chapter. All the background information will be utilized by our methodology to achieve our objective in the next chapter.

Chapter 3

Design and Implementation

This chapter introduces the approach to solve the story model generation problem with the help of Concept Net. In the first three sections, concrete steps of the approach will be introduced. The fourth section will show the results of the solution. Experiments to evaluate the consequence will be discussed in next chapter.

3.1 Overview

In order to generate reasonable events from two huge datasets, it is necessary for us to extract meaningful information from these two datasets. Because of the characteristics of these two datasets, our work can be roughly divided into two parts. In the first part, we process the original data from the dataset and extract useful information. In the second part, we generate the events with the help of given knowledge in Concept Net and an extending rule defined by ourselves. Since the outcome is not good enough merely using Concept Net, we use algorithms and techniques introduced in previous chapter to extend our events set via abstracting information from Brown Corpus.

3.2 Data Preprocessing

We have already mentioned in chapter 2 that although Concept Net has many strengths in providing information we need to generate events, this word net still brings us much inconvenience when doing the generation task. In order to extract useful information from this network, it is necessary for us to apply some preprocessing operation on the data in it. The operations we discuss in this section bring us much efficiency on the rest of work.

3.2.1 Acquire English Words

Since Concept Net is multilingual, the first step we need to do is extracting all the English words and phrases from the original dataset. Finding out what language each node belongs to is not difficult because the network use different labels to distinguish different languages. However, the official dataset for this network only provides us the information about edges rather than nodes. In this case, we decide to keep all

edges whose starting and ending nodes are all labelled as English text. After this step, we acquire all English words and phrases in the dataset. For convenience, we call the set composed of these English words as *English subset*, which is actually a subset of Concept Net.

3.2.2 Discard Meaningless Data

As we previously mentioned in chapter 2, texts in Concept Net are extracted from Wikipedia or some other open source word nets. Therefore, some texts with digits are included as English phrases in our subset. Examples including: *100 kilos* and *1000 meters*. Considering we are addressing the problem of story model generation, which mainly focuses on the backbone of a story plot, phrases with digits seem not to be helpful since they are more likely to play a descriptive role in a sentence and do not have much effect on the generation of backbone. In this case, we decide to delete all texts containing digits from the *English subset*. The new subset we get after this step is still called the *English subset* in the rest of this report.

3.2.3 Data Classification

Apart from the words, relations, which can also be regarded as the edges in the network, between different words provided by Concept Net is the element that we are more interested in. With these relations, we can easily find out what the connection is between two adjacent concepts. We can even generate a simple sentence based on the relations. One possible step to bring our generation task more convenience is to have a better understanding on the relation information contained in each word. Because of the reason we have already mentioned in section 3.2.1 that there are not enough information on nodes provided by the network, we choose to do this step based on the edges rather than nodes. We classify the elements in the English subset based on the relations. We traverse all the edges in the network of the English subset and put the concept in the directory of the relations that satisfied by current concept. It is apparent that one concept may satisfy more than one relation, so we may find different relation directories have some words in common.
Since verbs and nouns play different roles in a sentence, it is also necessary to distinguish them. On the basis of the former classification step, in the directory of each relation, we classify the concepts according to the part of speech for this concept. We use 'v' to label verbs, 'n' to label nouns, 'a' to label adjectives, 'r' to label adverbs and 'no_class' to label the concepts whose part of speech have not been marked by the Concept Net.
When we classify the concepts, we also keep the edge information. Therefore, in the directories we get, we can easily find out which concept satisfies what relation with which other concept. For instance, the directories tell us that *order meal HasPrequisite go to restaurant*.
After these two classification operations, we get 46 directories, each of which represents a relation. These directories are the data we are going to talk about in the the

following sections. Among all the 46 relations, there are 6 that hold special meanings, which are *HasPrerequisite*, *Entails*, *Causes*, *HasSubevent*, *HasFirstSubevent* and *HasLast-Subevent*. These specific relations describe the connection between concepts that are similar to the link between events in a planning model. We note that with these relations, we can order two related concepts in a logical order. Therefore, these specific relations can be used as the rules for connecting different events together when we apply planning strategies on our event model.
For clarity, we will call the directories we created *relation_ directory* and a specific directory, for example, the directory for the relation which is called *causes*, will be mentioned as *causes_directory*. We denote a specific file in a relation directory as *pos.relation_name*, for instance, *verb.causes*. With these files, we can start our work on next step, which is called event generation.

3.3 Event Generation

For purpose of addressing the problem of story model generation, it is necessary to have a good understanding on the structure of human-written stories. We choose one scenario from the classical fairy tale *Snow White and the Seven Dwarfs* as an example. This short paragraph describes the scenario of the snow white first come to the seven dwarfs' cottage.

Little snow-white was so hungry and thirsty that she ate some vegetables and bread from each plate and drank a drop of wine out of each mug, for she did not wish to take all from one only. Then, as she was so tired, she laid herself down on one of the little beds, but none of them suited her, one was too long, another too short, but at last she found that the seventh one was right, and so she remained in it, said a prayer and went to sleep. [Brothers, 2013]

Considering the habits of English expression, if the predicate of a simple sentence is a copula followed by an adjective, this sentence is more likely to express a status. In our project, we focus on the sentences that contain notional verbs first. According to this principle, we can extract five events from the plot above, they are: *snow-white eat vegetables and bread*, *snow-white drink wine*, *snow-white lay down*, *snow-white say a prayer* and *snow-white go to sleep*. From this example, we suggest that a story plot is consisted of several events, which means a event can be referred to the primitive unit of a story. Therefore, we consider the events generation first.
From the examples above, we also find that an event is mainly composed of a subject, a predicate and an object. Since there exists intransitive verbs in English, the object for a sentence seems to be optional while the subject and the predicate are necessary. Thus, we try to find appropriate subject, predicate and object from Concept Net so as to make them compose a sentence, which can also be regarded as the backbone of an event.
Because of the binary relation offered by Concept Net, we firstly find the subject and predicate pair, and the predicate and object pair separately. Then we match the pairs

we found. For brevity, we will call these two kinds of pairs *S-V* and *V-O* pair.

3.3.1 Extracting S-V and V-O Pairs

The Concept Net provides us many interesting relations. One of the most attractive relations is called *CapableOf*. This relation describes a pair of concepts: A and B, where A is capable of B. This kind of capability can rationally describe a connection between a subject and a predicate. From the *CapableOf_directory* we have already got, we can retrieve many *S-V* pairs. Another relation which can be similarly used as *CapableOf* is *Causes*, which describes a causal relation between two concepts: A causes B. *IsA* is the other most interesting relation. It suggests that A is a subtype or a specific instance of B. We generate a rule based on this relation to extend our *S-V* pairs.

Extending Rule If exists two subjects S_1 and S_2 satisfy a relation that S_1 *IsA* S_2, we also have an $S - V$ pair $S_2 - V_2$, then we can have a new $S - V$ pair $S_1 - V_2$.

With the three relations and the extending rule, we extract 10063 appropriate subjects and 40651 *S-V* pairs from Concept Net. Figure 3.1 offers details about number of verbs with certain number of subjects. From this figure, we can find that most of verbs have no more than 10 subjects to make up reasonable *S-V* pairs while there are also some verbs, each of which can be matched with more than 50 subjects.

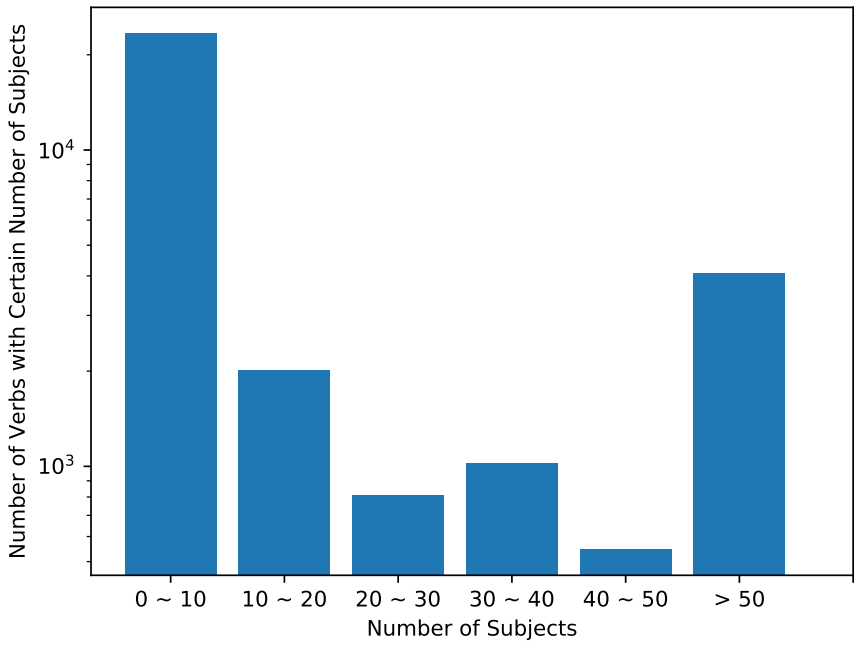

Figure 3.1: Number of Verbs with Certian Number of Subjects

V-O pairs can be similarly retrieved as the *S-V* pairs. The relation *Causes* is also

useful in this case. Method to apply this relation is the same as above. *CreatedBy* is a relation that needs to be used inversely. It describes a connection that A is created by B. Therefore, A represents an object and B is a predicate. The extending rule can be used as well on the basis of a relation called *MannerOf*. This relation is very similar to *IsA*, which suggests that A is a specific way to do B. With these three relations and the extending rule, we retrieve proper 11840 objects and 17115 *V-O* pairs. Figure 3.2 shows the distribution of verbs that are matched with different number of objects. From this graph, we can see that most of verbs have less than 10 appropriate objects to form *V-O* pairs.

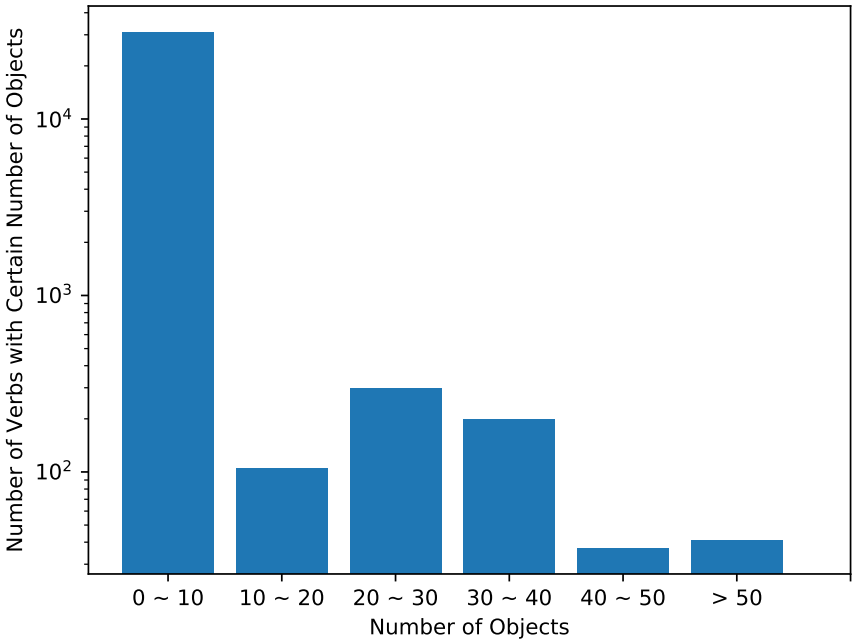

Figure 3.2: Number of Verbs with Certian Number of Objedts

3.3.2 Limitation of Concept Net in Retrieving Objects

It is apparent that if we want to match an *S-V* pair and a *V-O* pair and merge them in order so as to get an *S-V-O* chain, which can be referred to as a simple event as well, we should find two pairs that have common predicate. However, when we apply this operation on the pairs we have already got in previous steps, we acquire only 391 *S-V-O* chains. We checked these 391 concepts manually, which are used as predicates in each *S-V-O* pair and found that there are only 22 verbs in the set while 307 of the concepts are nouns or phrases. Figure 3.3 demonstrates part of the 391 concepts set.

The reason for this is that the relations we chose to extract *V-O* pairs focus on the relationship between two noun phrases rather than the connection between a verb

flood	money	hand sanitizer	censorship	lawsuit
insomnia	darkness	crash	lack of sleep	mental illness
loneliness	full bladder	damage	thirst	infidelity
car crash	run	parents	anxiety	zombie
thought	sport	work	heat	stress
exercize	pressure	music	plants	trust
murder	affair	eye contact	argument	handshake
creativity	consumerism	laughter	starvation	beer
flames	flatulence	drink	old age	hunger
intoxication	oxygen	snow	sunburn	ideology

Figure 3.3: Part of the 391 Matched Verbs

and a noun. In this case, merely depending on the knowledge of Concept Net is not enough. We use Brown Corpus for extending the pairs we have already got.

3.3.3 Extension of S-V and V-O Pairs

The key idea of this step is extracting *S-V-O* chains from data in the real world. Brown Corpus offers us abundant real world data. We use the Stanford CoreNLP toolkit [Bauer, 2014] to perform a dependency parsing on each data of Brown Corpus and extract possible *S-V-O* chains. Then we use the Maxnet POSTagger [Manning et al., 2014] to check the word class for the predicates of the potential *S-V-O* chains to ensure that it is a verb. Finally we use a lemmatizer from the NLTK library [Loper and Bird, 2002] to normalize the words in the possible chains we found. In order to make these chains correspond to the pairs we have already got, we divide each chain into two parts, one is an *S-V* pair, the other is a *V-O* pair.
This extension helps us retrieve 15182 appropriate subjects with 58163 *S-V* pairs and 19146 proper objects with 37014 *V-O* pairs in total. Figure 3.4 and Figure 3.5 show the improvement in number of appropriate subjects and objects of each verb respectively.

3.3.4 Matching S-V-O Chains

The methodology to match *S-V* and *V-O* pairs is same as the one we have already mentioned in section 3.3.2. Here are some examples of several chains we generated.
Example 1 Michael Jackson eat food.
Example 2 Animals drink cognac.
Example 3 Cook cook meal.
Example 4 Seasoner cook Sharon.
These chains ensure that we have 2638 verbs. When each of these verbs becomes the predicate of one sentence, there must exist at least one appropriate subject and object. We can generate a great number of events based on these chains and this is the unlimited creativity we want to have in our approach.

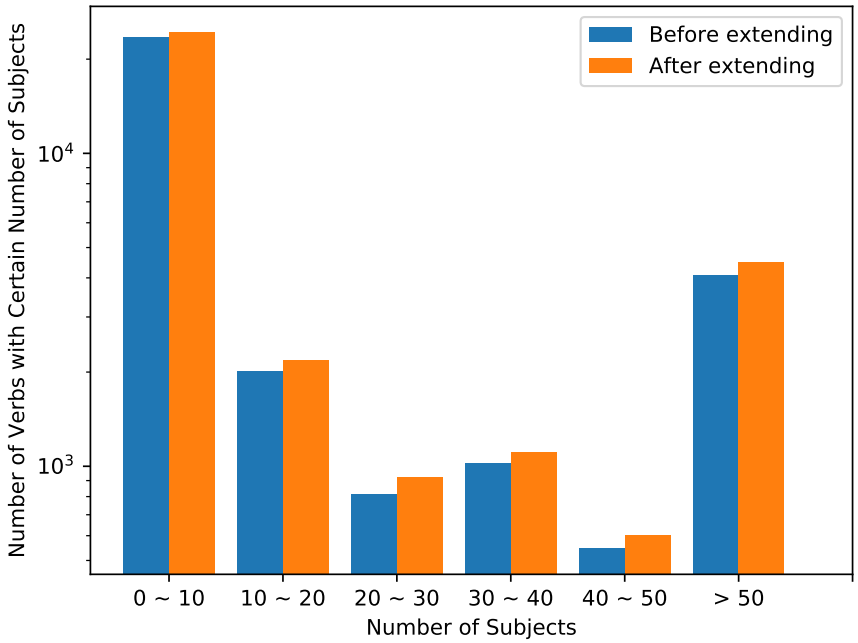

Figure 3.4: Comparison of Distribution Between Original S-V and Updated S-V

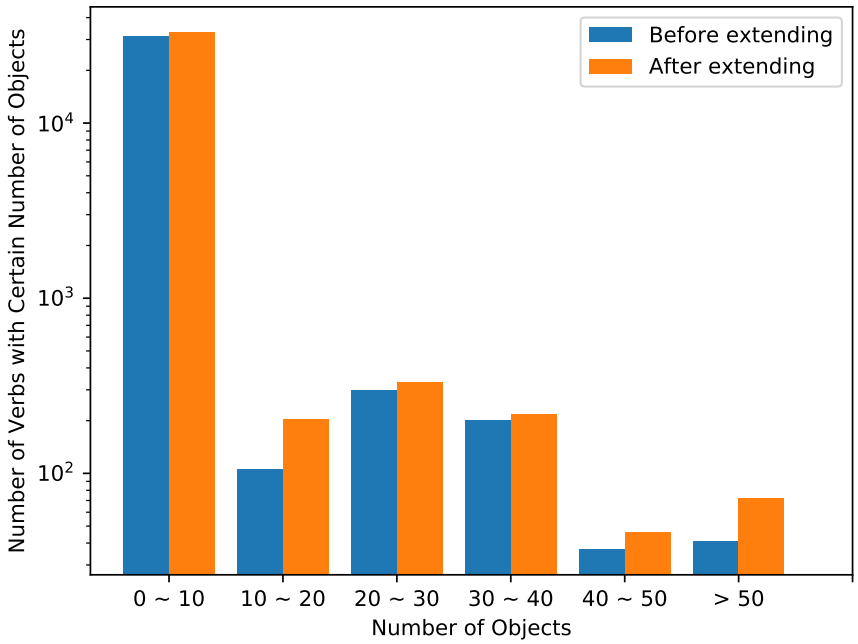

Figure 3.5: Comparison of Distribution Between Original V-O and Updated V-O

3.4 Results

With previous operations, we finally generate 106,801,767 events with 2638 verbs in total. If we ignore the events that use copula as predicates, there are 21,689,535 notional verb events left. From the examples we listed in section 3.3.4, some of the events we generated are reasonable while others may not be logical enough in the understanding of human cognition and need more constraints to make the sentence become reasonable. We will evaluate the quality of our model in next chapter by doing related experiments. Details about the analysis of our outcome will also be discussed in that chapter.

3.5 Summary

In this chapter, we introduced our approach to establish story model with the knowledge of Concept Net and Brown Corpus. The model we established with the approach of this chapter roughly achieves our goal of this project, which is to build a story model with logic and unlimited creativity for planning-based story generation problems.

Chapter 4

Experimental Methodology

This chapter presents the experiments done to evaluate our approach introduced in last chapter. The experiment is related to evaluating the basis of our inference, which is a key step in our approaches, and the outcomes of our approach as well. It also provides us an evaluation at the potential rules which can be used for connecting different events so as to address the story generation problem.

4.1 Overview

This experiment can be divided into three parts, which are corresponding to three different aims though the approaches to deliver the three parts are similar. The first part aims to check the reliability of the relations between two verbs. If we can get a good outcome in this part, we can draw a conclusion that these relations can be used as rules when we do further operation of planning-based story generation methodologies. We expect to prove that the basis of our inference, which is used for generating different pairs, is responsible as well by the second part of the experiment. The last part is much more important since it offers us a general evaluation at the events we generated from the two datasets.
Each participant will be asked to answer equal number of questions from a question sheet. The volunteers are expected to judge from their subjective point of view so that we can retrieve more convincing feedback generated by humans. It is meaningful since we plan to use our result for addressing story generation problems while it is necessary for each event to be closer to human thinking styles so that the story can be interesting and logical enough in human consciousness.

4.2 Participants

Before we introduce the concrete process of our experiment, we discuss who participant in this study first.
There are 50 participants in this experiment. Some of them are students who are studying in Australian National University (ANU). Rest of the students are from different universities, for example, University of Michigan and Duke University. Among

the students from ANU, some of them are undergraduate students, others are post-graduate students. These students have different background, some of them studying in Australia before they came to ANU, others having study experience in Singapore or China. Apart from the difference in background, these students are from different colleges as well. The majority of the participants study computer science or information technology while there are still some volunteers who study accounting or arts. We attempt to have people from completely different background to give us feedback of the outcome since we believe that living and educational background may influence the thinking style of people. The feedback given by people from different background can be more reliable.
Considering the volunteers may be worried about the privacy issues, we only collect the data that related to our project. No other personal information is collected during this experiment.

4.3 Methodology

The methodology to do this experiment is straightforward. We collect the data via questionnaire survey. Since we have mentioned that there are 10 relations that are helpful for our project in previous chapter and we have already collected large amounts of word pairs that satisfy different relations, for each question paper, we pick 5 pairs from the pair set of each relation. Therefore, there are 50 pairs on each question paper for the participants to answer. Among the 10 relations, six of them are the relations which can be used as rules to connect different events together. These relations are: *Causes*, *HasSubevent*, *HasFirstSubevent*, *HasLastSubevent*, *HasPrerequisite* and *Entails*. We call these relations as *Event Connection Relations* in following chapters. The other four are the relations we used for generating *S-V* and *V-O* pairs, they are: *IsA*, *CapableOf*, *MannerOf* and *CreatedBy*. These four relations are called *Event Generation Relations* for brief in following paragraph. The evaluation result of these four relations can give us a general feedback about whether the operations we applied on the datasets are reasonable or not. We covered all the relations we used in our approach introduced in last chapter. Besides, we also add 5 *S-V-O* pairs on each paper so that the participants can evaluate the quality of the events generated by our model. Thus, for each participant, he needs to answer 55 questions in total.
All the data involved in the question papers are selected randomly. We note that all the questionnaires are completely different from each other. The participants are required to judge whether each pair is reasonable or not from their own point of view. If they think the pair is relevant in logic, fill in the blank with 1, otherwise fill in it with 0. We ask the participants to provide binary answers rather than offer them a set of levels, such as 1-5, in which 5 represents for totally relevant in logic, 1 represents for totally irrelevant in logic and 3 means the medium level of relevance since we are worried that this kind of grade level may bring ambiguous results. This kind of ambiguous results cannot offer us enough information for judging the quality of our model. No extra information is provided to them when they make their judgements

in case the extra information might interfere participants. Considering if we offer the information of relation to the participants, which means let them know the original relation set of each pair, it may affect their judgements, we do not offer this part of information to the participants when doing the experiment.
Here is a sample questionnaire. We only list five word pair questions and last *S-V-O* pair questions in Figure 4.1 because of the limitation of space.

4.4 Results & Analysis

After the collection of the results we acquire from questionnaires, we calculate the accuracy of each relation separately. The equation of accuracy is defined as follows:

$$Accuracy_{Relation} = \frac{n_{positive}}{n_{Relation}}$$

In this equation, $Accuracy_{Relation}$ represents for the accuracy of logical relevance for each relation. For each relation, total number of the pairs that are marked as relevant in logic by the participants is denoted by $n_{positive}$. $n_{Relation}$ represents for the number of pairs for each relation that selected for this experiment. In our experiment, for all the relations, $n_{Relation} = 250$.
With the equation above, we compute the accuracy. The results are shown in the following tables.

Table 4.1: Accuracy of Event Connection Relations

Relation Name	Causes	HasSubevent	HasFirstSubevent
Accuracy	0.776	0.732	0.796
Relation Name	HasLastSubevent	HasPrerequisite	Entails
Accuracy	0.732	0.756	0.772

Table 4.2: Accuracy of Event Generation Relations

Relation Name	IsA	CapableOf	MannerOf	CreatedBy	S-V-O
Accuracy	0.760	0.768	0.816	0.800	0.596

Table 4.1 shows the accuracy for the event connection relations, which can be used as potential rules in further operation of planning-based story generation approaches. Data in Table 4.2 is mainly related to the relations we used for generating events. The events we generated in this project are also evaluated in this table. From the two tables we can find that all the relations perform high reliability in logic since the accuracy for each relation is approximately between 0.7 and 0.8. The accuracy for *S-V-O* pairs is not as high as the it of the relations, it is only around 0.6. We analyze the reason for occurrence of this phenomenon, the possible reason is as follows:

```
1  Are the following words relevant in logic? Put 1 in the bracket if they are
       relevant, otherwise put 0: [  ]
2          do_exercises, active
3
4  Are the following words relevant in logic? Put 1 in the bracket if they are
       relevant, otherwise put 0: [  ]
5          stay_in_bed, bed_sores
6
7  Are the following words relevant in logic? Put 1 in the bracket if they are
       relevant, otherwise put 0: [  ]
8          flirting, getting_caught_by_wife
9
10 Are the following words relevant in logic? Put 1 in the bracket if they are
       relevant, otherwise put 0: [  ]
11         taking_midterm, will_fail
12
13 Are the following words relevant in logic? Put 1 in the bracket if they are
       relevant, otherwise put 0: [  ]
14         watch_film, eat_popcorn
15
16                                         ⋮
17
18 Can the words listed below form a reasonable sentence or event? Put 1 in the
       bracket if they can, otherwise put 0: [  ]
19         relief_worker, take, pace
20
21 Can the words listed below form a reasonable sentence or event? Put 1 in the
       bracket if they can, otherwise put 0: [  ]
22         maxillo_facial_surgeon, do, Cardinal
23
24 Can the words listed below form a reasonable sentence or event? Put 1 in the
       bracket if they can, otherwise put 0: [  ]
25         space_age, give, instruction
26
27 Can the words listed below form a reasonable sentence or event? Put 1 in the
       bracket if they can, otherwise put 0: [  ]
28         change_of_location, begin, tone
29
30 Can the words listed below form a reasonable sentence or event? Put 1 in the
       bracket if they can, otherwise put 0: [  ]
31         cruiser, have, clearly
```

Figure 4.1: Sample of Questionnaire

On one hand, though our datasets provide us adequate constraints and heuristics to extract *S-V* and *V-O* pairs, they do not provide us enough information for matching them appropriately, which may cause the error in *S-V-O* pairs we achieved. On the other hand, we can find from the Table 4.2 that the accuracy for event generation relations are around 0.8, which means we already had errors occurred when we extracting *S-V* and *V-O* pairs. When we do match operation, the error may accumulate, which leads to the error of *S-V-O* pairs higher than those of the relations. Besides, though dependency parsing is a widely used technique to extract the relations between words in a sentence, it will still be challenging to extract accurate relations under every situation. Specifically, dependency parsing techniques often make mistakes when dealing with passive verbs. As analyzed in Bourdon et al. [1998], a typical mistake will be the problem of pseudo-passive. For example, when parsing the sentence "He was yelled at.", it will be quite challenging for the algorithm to group the words "yelled" and "at" as a atomic phrase. Similar examples can be found in our outcome, such as the fourth example we mentioned in section 3.3.4. This shortcoming can introduce a significant level of inaccuracy to the *S-V* and *V-O* pairs we extracted.

4.5 Summary

This chapter introduced the concrete process of our experiment on the outcome of our project. We collected the data by questionnaire survey. We also discussed and analyzed the result of experiment in this chapter. We got a conclusion that our model performs a high reliability in generating events as well as provides convincing rules used for connecting events together.

Chapter 5

Discussion

In this chapter, we follow the content of last chapter, summarize our approach for establishing a story model for addressing story generation problems and discuss strengths as well as weaknesses of our model compared to two other models. Some future work will also be covered in this chapter.

5.1 Current Result

With the approach we introduced in chapter 3, we extract 2638 verbs which are appropriate to be an action of an event. These verbs generate 106,801,767 events in total. Among these events, there are 21,689,535 events whose actions are notional verbs. If we consider the accuracy we get from the experiment that we discussed in last chapter, there are still more than ten million events that can be used for further operation of story planning approaches.
Former studies on story model already gain some outstanding results. Two of them are the model established by Cook in [Cook, 2011], which contains 1,852 story fragments and the model mentioned by Veale in [Veale, 2017], which offers us more than 3,000 story plots with 800 action verbs. The results provided by these two models have around ten thousand events and fifteen thousand events separately. Compared to these two models, our outcome has a significant superiority in number. However, it also has limitations. The quality of the results depends on the dataset we choose. For instance, if our data mainly comes from academic literatures, we may get events which contain a number of terminologies. On the other hand, the outcome may contain many simple words if the data is from fairy tales.
Apart from the large amount of events, the outcome also provides us some feasible methods to connect different events together. These methods can be regarded as potential heuristics when we apply the planning strategies so as to generate story plots in further steps. The reliability of these rules are evaluated by the experiment in last chapter as well, which shows that these rules are reasonable.

5.2 Future Work

In spite of the results we have already achieved in this project, there still remains many further work for improvements. On one hand, when we tried to find the objects for the verbs we get from Concept Net, we do not distinguish transitive verbs and intransitive verbs, which leads to our loss of the information of generating events that are based on intransitive verbs. Nevertheless, with the knowledge provided by Concept Net, it is hard for us to distinguish them. We may use the word net like Verb Net [Schuler, 2005] to provide us information on this aspect so that we can get more interesting *V-O* pairs. On the other hand, Concept Net not only provides us millions of words but also millions of phrases. We may be able to divide the phrases into two classes roughly, which are noun phrases and verb phrases, with the help of some natural language processing techniques so that we can extract more interesting information from the phrases and make use of them for event generation.

Chapter 6

Conclusion

In this project, we introduced a methodology to make the machine automatically generate a story model with not only logic but also unbounded creativity. We firstly processed the data in Concept Net and retrieved many *S-V* and *V-O* pairs. Since these pairs are not satisfying enough, we extracted some other pairs from Brown Corpus by applying traditional NLP techniques on it. We matched these two parts of data. With the aid of the knowledge extracted from Concept Net and Brown Corpus, we achieved a huge story model with more than ten million events in it. Experiments have proved the high reliability of this model and there exists potential solution to address story generation problems with this model.

Appendix A

Study Contract

Australian National University

INDEPENDENT STUDY CONTRACT

Note: Enrolment is subject to approval by the course convenor

SECTION A (Students and Supervisors)

UniID: u5962473 ____

SURNAME: Wen FIRST NAMES: Musha

PROJECT SUPERVISOR (*may be external*): Patrik Haslum

COURSE SUPERVISOR (*a RSCS academic*): Peter Strazdins

COURSE CODE, TITLE AND UNITS: COMP8755, Individual Computing Project, 12 units

SEMESTER ☐ S1 ☐ S2 YEAR: _____

PROJECT TITLE:

Story Planning: Unlimiting Creativity

LEARNING OBJECTIVES:

On the completion of the project, the following learning objectives are expected to achieve:

- Have a good knowledge of narrative planning and supporting techniques
- Be able to conduct research in existing automatic narrative generation approaches
- Develop solid skills in implementing planners
- Be able to deliver a project report of research outcomes.

PROJECT DESCRIPTION:

Approaches to generate a story automatically has been an attractive topic in the field of Artificial Intelligence. Machine learning approaches have been developed and can produce generative models from data in a mostly unsupervised manner. However, these models can only replicate the surface appearance of fiction but are unable to generate a coherent plot so far. Since the essence of a story is sequences of events that change the state of the (story) world, which is similar to the essence of a plan, a narrative of a story is thought to have parallels with a plan. This has led researchers to study planning-based approaches to automatic narrative generation.

According to the basic idea of planning, a formal description of the story world, including characters, places, objects, their relations, and the things they can do, is required to contribute a model. Nevertheless, writing this world model may not only be a great burden on systems designers but also limit the creativity of it. The ultimate goal of this project is to build a planner which can automatically generating a story with both unbounded creativity and significance.

In this project, we will first study and evaluate some existing approaches of narrative generation. In this stage,

Australian National University

We will summarize the advantages and disadvantages of different planners. After that, we will focus on combining and improving the existing planners. As different approaches are examined and combined, we will also be utilizing the resources we have, a semantic database for example, to evaluate these approaches. Finally, we may explore the existing approaches and try to introduce a new feasible approach for automatic narrative generation.

ASSESSMENT (as per the project course's rules web page, with the differences noted below. Supervisors, please nominate an examiner):

Assessed project components:	**% of mark**	**Due date**	**Evaluated by:**
Report: name style: research report (e.g. research report, software description..., no less than 45% weight assigned)	60%		(examiner)
Artefact: name kind: software (e.g. software, user interface, robot..., no more than 45% weight assigned)	30%		(supervisor)
Presentation:	10%		(course convenor)

MEETING DATES (IF KNOWN):

STUDENT DECLARATION: I agree to fulfil the above defined contract:

Musha Wen ..... 27/7/2017 .....

Signature Date

SECTION B (Supervisor):

I am willing to supervise and support this project. I have checked the student's academic record and believe this student can complete the project.

..... 27/7/2017 .....

Signature Date

REQUIRED DEPARTMENT RESOURCES:

Australian National University

SECTION C (Course convenor approval)

....

Signature Date

Appendix B

Readme

Directory structure

```
.
|___ conceptnet_relation_classification
| |___ relations_all:  the directory contains data of relations after classficatio
|  \__ relation_classify.py:  Python script which read the raw relation data
|                             from concept and classify the relations
|___ tests: the directory contains all question papers for evaluation
|___ tests-results-code:  the directory contains answers to all question papers for evaluation
|  \__ test-accuracy.py:  the script which calculates the accuracy from question papers
|___ util.py:       utility functions for reading the classified relation data
|___ FragmentGeneration.py:     Generate fragments from specified relations
|                               and write the results to file
|___ fragments_v.txt:   the fragments generated from specified relations.
                        each row represents a fragment
|___ extend_by_manner_of.py:  extend the fragments generated using the relation MannerOf
|___ fragments_v_extended_manner_of.txt:  the fragments after extending with relation MannerOf
|___ AssignSubjects.py:   Assign subjects to verbs based on the specified relations
|___ EventGenerator.py:   Python script to generate events based on the relations
|___ EventDatasetGenerator.py:  Generate events and store them in a specific format.
|                               See details below.
|___ GenerateTests.py:  Python script to generate questions papers for evaluation
 \__ BrownData
  |___ Server.java:     Setup a socket server, receive a query of a sentence,
  |                     use StandfordNLP toolkit to extract required relations
   \__ query.ipynb:     read the Brown corpus, for each sentance,
                        send a query to the backend server and record the relations returned
```

Running Instruction

To run all the code and reproduce the result, use the following steps:

1. Prepare dataset and setup environments

   The dataset and Standford CoreNLP library is not included. First, download the `assertions.csv` from ConceptNet website and place it in directory `conceptnet_relation_classfication`. This can be setup by

   ```
   $ cd conceptnet_relation_classfication`
   $ wget https://s3.amazonaws.com/conceptnet/precomputed-data/2016/assertions/conceptnet-assertions-5.5.0.csv.gz
   $ gunzip -cd conceptnet-assertions-5.5.0.csv.gz > assertions.csv
   ```

   Then, download Standford CoreNLP toolkits from Standford NLP Toolkit website. In the root directory of code repo, execute the following code in a bash shell

   ```
   $ mkdir -p BrownData/lib
   $ cd BrownData/lib
   $ wget https://nlp.stanford.edu/software/stanford-parser-full-2018-02-27.zip
   $ wget http://nlp.stanford.edu/software/stanford-corenlp-full-2018-02-27.zip
   $ wget http://nlp.stanford.edu/software/stanford-english-corenlp-2018-02-27-models.jar
   ```

   After the downloading finished, unzip the zip files and place the `.jar` files in the directory `lib`.

   Finally, download `JSON` and `slf4j` package for `Java`, these packages will be used in building a socket server backend:

   ```
   $ wget http://central.maven.org/maven2/org/json/json/20180130/json-20180130.jar
   $ wget http://central.maven.org/maven2/org/slf4j/slf4j-simple/1.7.25/slf4j-simple-1.7.25.jar
   $ wget http://central.maven.org/maven2/org/slf4j/slf4j-api/1.7.25/slf4j-api-1.7.25.jar
   ```

2. Relation classfication

   To classfy the relation from ConceptNet's raw data file `assertions.csv`, first change directory to `conceptnet_relation_classification`. Then run with Python 3: `$ python3 relation_classify.py`
3. Generate relations from Brown corpus To generate the relations of subjects and objects, we use the Brown corpus. First compile the `Java` source file `Server.java`, and run the main class `Server` with `$ java -cp lib/* Server`. Then, run the `IPython` script `query.ipynb`. This requires the `nltk` library installed and the Brown corpus downloaded. The script can be run with `$ jupyter notebook` The relations created will be write to text files. Create corresponding directories in `conceptnet_relation_classification` and move the relation files to correct directories.
4. Generate and extend fragments To create fragments with specified realtions, run with `$ python3 FragmentGeneration.py`. This will create a text file `fragments_v.txt`. Each line of this file represents a fragment generated based on the relations We will also extend these fragments. Run with `$ python3 extend_by_manner_of.py` to create the text file `fragments_v_extended_manner_of.txt`. The format of this file is the same as original fragments file, but this extended version also include the relations which are extended from the relation MannerOf.
5. Generate events

   The `EventGenerator.py` provides the class `EventGenerator` which can generate all possible events given a verb. The method `EventGenerator.sample_event` can randomly sample a possible event.
6. Create evaluation question papers To re-create question papers, run `$ python3 GenerateTests.py`. To calculate the accuracy, change directory to `test_results` and run `$ python3 test-accuracy.py`

Output format

This project builds two data sets: fragments and events.

The fragments generated are stored in `fragments_v_extended_manner_of.txt`. Each row in this file contains three words: (starting, ending, relation). The starting and endding words are the cores of events, usually verbs. The relation of these two events is suggested by relation term.

The events data set is stored in directory `events` and contains two text files: `words_id` and `events.data`. For `words_id`, each row in this file represents a tuple (id, word), which defines the mapping between words appearing in events and their ids. For `events.data`, each row contains three ids, which are defined in `words_id` and the corresponding words form an event, i.e. a `S-V-O` chain.

We have sampled some events and stored in the format we described above. To generate a different set of samples, run in shell `$ python3 EventDatasetGenerator.py`. The parameters in `EventDatasetGenerator.py` can be modified to control the size of dataset.

Bibliography

Bauer, J., 2014. Shift-reduce constituency parser. https://nlp.stanford.edu/software/srparser.html. (cited on pages 6 and 14)

Bourdon, M.; Da Sylva, L.; Gagnon, M.; Kharrat, A.; Knoll, S.; and Maclachlan, A., 1998. A case study in implementing dependency-based grammars. *Processing of Dependency-Based Grammars*, (1998). (cited on page 21)

Brenner, M., 2010. Creating dynamic story plots with continual multiagent planning. In *AAAI*. (cited on page 2)

Brothers, G., 2013. *Snow White and the Seven Dwarfs*, vol. 501. Trajectory, Inc. (cited on page 11)

Cook, W., 2011. *PLOTTO: the master book of all plots*. Tin House Books. (cited on pages 2 and 23)

Francis, W. N., 1965. A standard corpus of edited present-day american english. *College English*, 26, 4 (1965), 267–273. (cited on pages 2 and 6)

Goodwin, R. and Sharp, O., 2016. Machines making movies sunspring and project benjamin. https://githubuniverse.com/2016/program/sessions/#machines-making-movies. (cited on page 2)

Liu, H. and Singh, P., 2004. ConceptnetâĂŤa practical commonsense reasoning tool-kit. *BT technology journal*, 22, 4 (2004), 211–226. (cited on pages 2 and 5)

Loper, E. and Bird, S., 2002. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, 63–70. Association for Computational Linguistics. (cited on page 14)

Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S. J.; and McClosky, D., 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, 55–60. http://www.aclweb.org/anthology/P/P14/P14-5010. (cited on page 14)

Perez-Ortiz, J. A. and Forcada, M. L., 2001. Part-of-speech tagging with recurrent neural networks. In *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*, vol. 3, 1588–1592. IEEE. (cited on page 7)

Popov, A., 2016. Deep learning architecture for part-of-speech tagging with word and suffix embeddings. In *International Conference on Artificial Intelligence: Methodology, Systems, and Applications*, 68–77. Springer. (cited on page 7)

Sayers, D. L., 1936. Aristotle on detective fiction. *English*, 1, 1 (1936), 23–35. (cited on page 1)

Schuler, K. K., 2005. *Verbnet: A Broad-coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, Philadelphia, PA, USA. AAI3179808. (cited on page 24)

Veale, T., 2017. Déja vu all over again. In *Proceedings of the 8th International Conference on Computational Creativity*. (cited on page 23)

Yi, C., 2015. An english pos tagging approach based on maximum entropy. In *Intelligent Transportation, Big Data and Smart City (ICITBS), 2015 International Conference on*, 81–84. IEEE. (cited on page 7)

Zheng, X.; Chen, H.; and Xu, T., 2013. Deep learning for chinese word segmentation and pos tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 647–657. (cited on page 7)