Extracting Events from Fairy Tales for Story Summarization

Bianca Trish Adolfo and Ethel Ong[1]
Center for Language Technologies - AdRIC, De La Salle University
[1]ethel.ong@dlsu.edu.ph

ABSTRACT
Automatic text summarization is mostly used to provide quick access to relevant information from a huge volume of documents and news, especially from online document search facilities. Summarizing fictional stories, however, may pose some challenges for the machine since important information can appear in unexpected places in the text. A prerequisite for generating story summaries is a computational model that captures the events needed to recreate the story while ignoring irrelevant details without losing the central idea of the story writer. In this paper, we describe our approach in using extractive summarization techniques to identify relevant events from a corpus of five (5) fairy tales. Results from comparing the events found in the computer-generated summary against a human-annotated events as the reference text showed that the event extraction algorithm has a precision of 62.33% representing the relevant events that were extracted, a recall of 42.25% representing the percentage of total relevant results that were retrieved, and an f-measure of 50.36% that specifies the accuracy of the test. Problems with the varying sentence structure led to incorrect and missing extraction instances for different event details. The extraction algorithm also encountered difficulty when dealing with clauses.

Keywords
Event extraction, story world graph, story summarization, fairy tales

1. Introduction
A summary is a text that contains information retrieved from one or more source documents [1]. Humans perform summarization by recognizing important ideas from a given document and then integrating these into a reduced and concise text that captures the meaning and central theme from the original content. This gives the reader a condensed yet concise account of the main ideas to highlight the important points while disregarding irrelevant text without losing the information content. Recognizing and selecting important information from the original text to include in the summary, however, is subjective and varies among individuals who perform the summarization. Writers and readers may also have differing preferences in deciding when a particular story element is deemed important.

Automated text summarization has employed different techniques in looking for the important parts of a given document. One of the most common approach is to calculate the word frequencies, which can include keywords and proper nouns from a particular domain. The word frequencies are used in scoring sentences. Selection of sentences is based on the number of high frequency words contained in the sentence and the sentence's position in the original text. Sentences can also be scored based on the presence of cue words, such as "in summary" and "in conclusion" [2]. The presence of numerical values may also highlight the importance of a sentence depending, for example, in scientific or medical literature [3]. Other measures include similarity of words with the title of the document, vocabulary overlap between sentences, word co-occurrences, and sentence length [2].

Using these strategies as basis for extracting important sentences have been shown to work with medical literature [3], legal documents and news articles, where the predictable location of items, abundance of surface markers, template-like structures and straightforwardness of textual content help facilitate the summarization task [4]. However, in documents such as dialogues, speech and stories, important information can appear in unexpected places. Identifying these remains one of the challenges facing research in automated summarization. Another challenge with using current summarization techniques on stories is that stories do not have blocks of text that summarize the main idea, and there is an even smaller chance of finding the important ideas in the same position across different stories [4].

Research in the automatic summarization of stories have been conducted by Kazantseva and Szpakowicz [4] and Zhang [5]. In the former, the generated summaries contain text describing the setting and the main characters, without revealing the plot, to help readers decide whether or not they are interested in further reading the story. Zhang [5], on the other hand, built a cognitive model of narrative comprehension and coherence used and a proposition-based summarization strategy in summarizing a story. Zhang's model is created by extracting propositions from each sentence in the story, to mimick how the human brain retains some extracted information while forgetting the others.

During a reading task, children must be able to focus their attention in order to decode words, maintain reading fluency, and understand what they read [6]. However, because children have short attention spans, shortened text are more suitable than longer ones especially for beginning readers. Furthermore, reading and writing classes have employed summarization activities to teach students to discern important ideas from a given text and to reduce it to main points in order to facilitate comprehension. Marzano [7] reported that summarizing is crucial to comprehension as it allows students to "restate content in a succinct manner that highlights the most crucial information", enabling them to increase their understanding of content by 19 percentile points.

Stories promote the acquisition of language, culture and values for young children. Thus, it is a common practice for publishers to come up with varying lengths of classic and popular stories to make them appropriate to different age groups. Publishers use a leveled reading system to indicate the readability of a text and tailor materials to the student's reading abilities [8]. A prime example, classic fairy tales, have survived through time by being

Philippine Computing Journal Dedicated Issue on Natural Language Processing, pages 25–33
Vol. XIV No. 1 August 2019

adapted into different versions. These are then retold to young learners to give them opportunities to learn and reflect on important values and lessons that can be gleaned from the stories.

The main contribution of our study to the field of computing is to combine techniques in automated text summarization and story understanding, particularly event detection and extraction, in order to produce summaries of select classic fairy tales. A story world graph is designed to represent story characters, and the temporal and causal relationships of story events. This is subsequently used to produce a computer-generated summary for each of the classic fairy tales.

We used Dolphin Books' Classic Tales Collection as our corpus to develop and to evaluate our summarizer. Specifically, the corpus is comprised of the following stories: *Pied Piper*, *The Ugly Duckling*, *The Little Mermaid*, *Snow White and the Seven Dwarfs*, and *Cinderella*. These stories have certain characteristics, such as the obvious presence of good and evil characters, clear plots with no subplots, and the existence of a moral value or lesson [5].

In this paper, we present the design of our story world graph that is adapted from the story world model commonly used in automated story generation [9][10] and interactive storytelling systems [11]. Then, we detail the processes and challenges in identifying events and in extracting story elements from our input corpus to populate this graph. We followed this with a discussion of the results in validating the performance of the event extraction algorithm using precision, recall and F-measure. We also describe how the story world graph is used by an automated summarizer to produce a summary of the given fairy tales. We end our paper by presenting opportunities for future work to improve the performance of the event extraction algorithm and the automated story summarizer.

2. Modeling Fairy Tales

Stories are primarily comprised of characters who inhabit the story world to perform a sequence of actions or events that effect changes to the world while enabling them to progress toward the attainment of their goals. Automated story generation systems typically involve building a computational story world model to represent these characters and their candidate actions and events that can take place in the stories to be generated, given the target domain and genre. Summarizing stories, on the other hand, entail detecting the characters and events found in a given input story and representing them in a computational model that the summarizer can subsequently use to do its task.

Previous works on story summarization have utilized different models to represent a story and its elements. Charniak [12] translates each statement from an input story into assertions to relate the story to background real world knowledge. The extracted assertions are later used to generate responses that answer questions about the input story. Lehnert [13] used affect units or structures that overlap to encode story arcs and model the plot of the story.

Another approach is with the use of a discourse graph as proposed in [14]. Using multiple documents as input, each node in the graph is a sentence while the directed edges denote the possible ordering of the nodes. Zhang [3], on the other hand, extracts propositions, in the form of `predicate(arg1, arg2, ...)` from each sentence. The predicate can be a verb, noun or adjective, but the arguments *arg* must be nouns.

We use a story world graph to capture and to represent story events, their elements and their relations with one another. Story generation systems use the term *story world* to refer to a representation of everything that is happening in the story, including the description of the setting, the state of the characters and objects, and even the sequence of events that have taken place [9][10][15]. We adhere to this definition to describe our story world graph, where nodes represent the events and characters, while edges represent the event sequence.

2.1 Event Model

Different definitions of an event exist in literature. In our work, we define an event to be an action that a story character has performed, as well as a character's response to another character's actions or to some naturally occurring phenomena [16]. Events representing character actions and responses are usually expressed as verbs in the story text. Some events can only be executed at a specific location, may utilize certain objects as instruments (e.g., *spoon* for eating), and may involve a recipient of the action (e.g., eating *ice cream*). Referred to as the elements of an event in our research, these are similar to the concept of semantic roles used to recognize the predicate-argument structure of a given sentence [17]. Table 1 shows the structure for representing a story event in an event model, including an event's associated elements.

Table 1. Representation of a Story Event

Field	Description
Action	Verb or verb phrase
Agent	Subject or Doer of action
Purpose	Purpose of the action
Cause	Cause of an action
DirectObj	Direct object
IndirectObj	Indirect object
Complement	Subjective/Objective complement
Location	Where the event took place
Time	When the event took place
SentenceID	Reference to the original story text
Coref	Referent of the agent if it is a pronoun

Because there are numerous forms of verbs in the English language, not all verbs from the story text are treated as events. Copula or linking verbs do not describe actions; but instead, they are used to connect the subject of a sentence to the predicate. The predicate can be another noun or an adjective. *Be* verbs are prime examples. They are used in stories to describe a character with the use of adjectives (i.e., *Ariel is lovely*.) or another noun (i.e., *Ariel is a mermaid*.); the character's state (i.e., *Hansel is hungry and tired*.); and even the relationships between characters (i.e., *Eric is Ariel's husband*., *Ariel's father is King Triton*.). As these descriptions are useful to understand story characters, the elements extracted from these types of verbs are used to populate a character model, as described in the next section.

2.2 Character Model

Characters are essential to stories. They perform actions to address a certain conflict or dilemma, and to achieve a target goal. The actions they perform can effect changes to the story world; while their physical and emotional state can be affected by the occurrence of various events in the story world.

A portion of the story world graph is thus dedicated for storing details about the story characters using the character model in Table 2. Each character has a *name*, *gender*, and frequency or *number of times mentioned* in the story. During initialization, *gender* is set to *unknown* and the frequency count is set to *zero*. The values for these are updated after coreference resolution has been performed on an input story text.

Table 2. Representation of a Story Character

Field	Description
Name	Name of the character
Gender	Gender of character (default= 'unknown')
NoOfTimes Mentioned	Frequency count used to determine the importance of the character
Description[]	Adjectives or complements describing the character
Relationship[]	Connections with other characters

Characters can have zero or more *descriptions* and zero or more *relationships* with other characters. These are extracted from sentences that do not contain events but instead have the form NounPhrase1 - Verb [be] - Nounphrase2, where the linking verbs connect the subject to the complements. The complements provide relevant information that can describe the characters.

Figure 1 shows the resulting character model for *Ariel* that is derived from the story text "*Ariel is a mermaid. She is a singer. Ariel is Prince Eric's wife.*"

```
Character:              Ariel
Gender:                 Female
NoOfTimesMentioned:     3
Description:            [a mermaid, a singer]
Relationship:
    Name:       Prince Eric
    Connection: wife
```

Figure 1. Character Model - Ariel

3. Extracting Story Events

Event extraction is the process of identifying and extracting events from an input story text, including information about the story character (termed the *actor* or doer of the action), *when* and *where* the event occurred, to whom or who is affected by the event, and why the event occurred. A number of challenges need to be addressed when extracting these events and their elements. Knowledge-based and data-driven approaches can be utilized. The former uses linguistic patterns while the latter uses annotated data as basis for extraction. We used a knowledge-based approach where linguistic features are identified with the use of Stanford CoreNLP. The process is outlined in Figure 2.

Figure 2. Process Flow for Event Extraction

3.1 Character Names

Names given to story characters come in various forms: (1) popular proper nouns, i.e., *John* and *Mary*; (2) classic proper nouns, i.e., *Cinderella* and *Hansel*; and (3) compound nouns, i.e., *Snow White*, *Little Red Riding Hood*. Usually, secondary characters with minor roles are referred to in the story using their common nouns, i.e., *woodcutter*, *stepmother*, *queen* and *father*. Fable stories, which have animal characters, would have animal names as the character names, i.e., *three little pigs*, *wolf*, and *ugly duckling*.

Aside from using the PERSON tag supplied by Stanford CoreNLP, special character names are determined by identifying the agents in each sentence. These agents are then considered as characters if they appear more than once in the story. The assumption is that appearing more than once means the agent has a role to play and should therefore be tracked.

3.2 Resolving Coreferences

Relying on the occurrences of a character's name in the story text alone is not sufficient to identify all events that the character is involved in. This is because characters may be referred to differently in the story, for example, *Ariel* can be referred to as *she*, *mermaid*, *her* and *princess*.

Stanford CoreNLP has a method for resolving coreferences. Given the sample sentences in Listing 1, Stanford CoreNLP produces an output as shown in Figure 3.

Listing 1. Sample sentences to coreference resolution.

```
[1] John is a prince.
[2] Mary is a princess.
[3] John likes her.
[4] She is pretty.
```

```
{
    1=CHAIN1-["John" in sentence 1,
              "a prince" in sentence 1,
              "John" in sentence 3]
    2=CHAIN2-["Mary" in sentence 2,
              "a princess" in sentence 2,
              "her" in sentence 3,
              "She" in sentence 4]
}
```

Figure 3. Output of Coreference Resolution

Each chain corresponds to one character, and lists the sentence numbers and the coreferences appearing in that sentence. The number of unique sentences that a particular character is mentioned in is used as the value for *NoOfTimesMentioned.* For example, *John* (represented by CHAIN1) is mentioned twice while *Mary* (represented by CHAIN2) is mentioned three times in the given sentences.

Another output is the representative mention, shown in Figure 4, which is the most distinguishable name out of all the coreferences in the chain. The representative mention is used to replace all other coreferences of the character in the story. For example, "*John likes her.*" is transformed to "*John likes Mary.*"

```
representative mention: "John" [1] is mentioned by:
        a prince [1]
        John [3]
representative mention: "Mary" [2] is mentioned by:
        a princess [2]
        her [3]
        She [4]
```

Figure 4. Representative Mention from Stanford CoreNLP

Aside from resolving character references, the output of coreference resolution is also used for two other purposes: (i) To determine the number of times a character is mentioned; the higher the count, the more important the character is treated to be; and (ii) To determine the character's gender. By default, Stanford CoreNLP's gender annotation function is used first. If it returns a *null* value, then the pronouns found in the coreference resolution output is used instead.

3.3 Annotating Elements of a Sentence

Before events can be extracted, the dependency relations among the different elements of a sentence - agent (noun phrase), event (verb) phrase, and purpose phrase (verb phrase complement) - must be determined, as shown in Figure 5. Three approaches are used to determine these elements, namely constituent tree traversal, word relationships, and dependency graph based on word relationships for processing compound sentences.

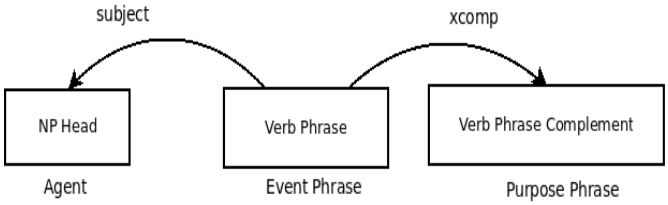

Figure 5. Dependency Relations in Sentences [18]

Constituent trees are quite long; but they are useful when extracting clauses and phrases based on their POS tags (e.g., *NP*, *VP*). The word relations are used to determine the usage of the word in a sentence, e.g., as a *root* (verb), *nsubj* (subject), *dobj* (direct object), *advcl* (adverbial clause); its lemmatized form, named entity, POS tag, as well as the index position of the word in the sentence. The dependency graph is used to split a compound sentence of the form S-CC-S, such as "*She accepted the invitation, and all of them were glad.*", into simpler sentences by utilizing the constituent tree and word relations. These are shown in Figures 6 and 7, respectively.

3.4 Classifying Sentences

Sentence patterns are used to classify the annotated sentences. There are currently nine (9) sentence patterns considered in our study. These are grouped into five (5) categories, as shown in Table 3, including sample sentences for each pattern. The first and last categories, the *Be-Verbs Description* and *Relationships*, do not contain events but are instead used to extract character descriptions and relationships, respectively.

The second category is used to extract the *Complement* for the subject. Checking each sentence against a predefined list of linking verbs is not an option as this would be resource-heavy.

```
(ROOT
  (S
    (S
      (NP (PRP She))
      (VP (VBD accepted)
        (NP (DT the) (NN invitation))))
    (, ,)
    (CC and)
    (S
      (NP
        (NP (DT all))
        (PP (IN of)
          (NP (PRP them))))
      (VP (VBD were)
        (ADJP (JJ glad))))
    (. .)))
```

Figure 6. Constituent Tree Representation of a Sentence

```
WORD: She -  REL: nsubj -  IND:
WORD: accepted -  REL: root -  I
WORD: the -  REL: det -  IND: 3
WORD: invitation -  REL: dobj -
WORD: and -  REL: cc -  IND: 6 -
WORD: all -  REL: nsubj -  IND:
WORD: of -  REL: case -  IND: 8
WORD: them -  REL: nmod:of -  IN
WORD: were -  REL: cop -  IND: 1
WORD: glad -  REL: conj:and -  I
```

Figure 7. Word Relations in a Sentence

The absence of a direct object after the verb phrase is used to identify sentences that fall under the third category, *Intransitive Verbs*. Most sentences in a story fall under the fourth category, where the main marker is the presence of a *Direct Object*. The sheer number of possible sentence variations makes classification task the hardest for this category.

While stories abound in dialogues, these are currently not included in the scope of our study.

3.5 Story World Graph

The story world graph is the representation of everything that is happening in the story. In the graph, nodes are used to represent the events and the story characters, while edges are used to represent the event sequence. Given the following story excerpt from *Cinderella*, the corresponding story world graph derived from the sentences in the story text is shown in Figure 8.

> Once upon a time, there was a beautiful girl named Cinderella. She lived with her wicked stepmother and two stepsisters. They treated Cinderella very badly. One day, they were invited for a grand ball in the king's palace.

The circular numbered nodes represent the sentence pattern associated with a particular sentence in the story. This sentence pattern is used to determine the grammar rules to be applied during event extraction. The nodes are connected to each other from left to right based on the sequential order of appearance of the sentence in the story text. This also assumes a temporal ordering of event occurrences with no flashbacks. Rectangles depict event details (following the event model indicated in Table 1) that are explicitly stated in the sentences. Coreferences are represented by a curved shape; since these could span multiple events, edges are used to connect the character names to their representative mention in the story text.

Table 3. Categories of sentence patterns with example story text

Category & Pattern No.		Sentence Pattern	Example Story Text
Be-Verbs Desc	1	NP1 + V-be + Adv/TP	Snow White is here.
	2	NP1 + V-be + Adj (SC)	She was so tired.
	3	NP1 + V-be + NP1 (SC)	Ariel is a mermaid.
Complement	4	NP1 + LV + Adj (SC)	The apple looks very delicious.
	5	NP1 + LV + NP1 (SC)	Ariel became a princess.
Intransitive V	6	NP1 + V-int (no DO)	The prince will arrive.
Direct Object	7	NP1 + V-tran + NP2 (DO)	She asked her magic mirror.
	8	NP1 + V-tran + NP2 (IO) + NP3 (DO)	She told the dwarves her sad story.
Be-Verbs Rel	9	NP1 + V-be + NP2 (special case)	Eric is Ariel's husband.

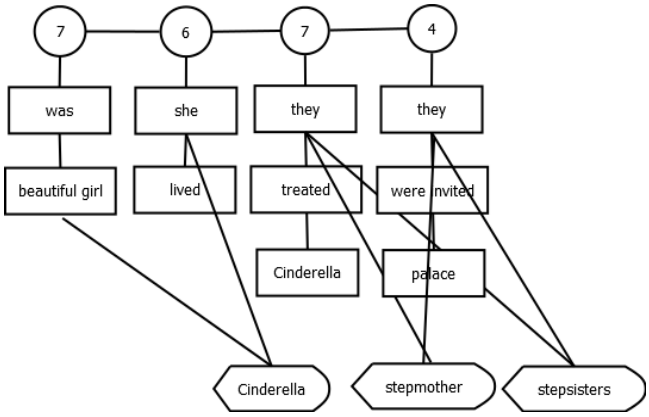

Figure 8. Story World Graph for an Excerpt of *Cinderella*

4. Issues in Event Extraction

From the nine (9) sentence patterns in Table 3, the event extractor can extract story elements for 52 sentence varieties. Several issues were encountered due to the variances in the structure of sentences found in the corpus of fairy tales.

4.1 Extracting Time

The constituent tree sometimes treats time as part of the prepositional phrase for location, thus doing away with the NP-TMP temporal tag needed to distinguish time elements, as shown in Figure 9. In other occasions, such as in Figure 10, the problem was reversed, with NP-TMP tagging the time but the location tag is now missing.

```
CONSTITUENT TREE (PENN):
(ROOT
  (S
    (NP (PRP She))
    (VP (VBZ is)
      (PP (IN at)
        (NP (DT the) (NN park) (NN today))))
    (. .)))
```

Figure 9. Time Element as part of Location

4.2 Identifying Agents

A sentence in the story can reference one or more characters. In sentences with multiple characters, one event instance is generated for each identified character. This, however, is not the case for the sentence "*Snow White, her prince, and the seven dwarves are in the woods.*", since "*her prince*" is treated as a possessive noun rather than a separate subject as shown in the word relations in Figure 11, leading to the corresponding event models in Figure 12. This scenario highlights the importance of word relations to correctly identify the different agents in a sentence for event extraction to work properly.

```
CONSTITUENT TREE (PENN):
(ROOT
  (S
    (NP (PRP She))
    (VP (VBZ is)
      (NP-TMP (JJ outside) (NN today)))
    (. .)))
```

Figure 10. Location Element as part of Time

```
WORD: Snow -  REL: compound -  IND: 1 -  LEMM
WORD: White -  REL: nsubj -  IND: 2 -  LEMMA:
WORD: her -  REL: nmod:poss -  IND: 4 -  LEMM
WORD: prince -  REL: conj:and -  IND: 5 -  LE
WORD: and -  REL: cc -  IND: 7 -  LEMMA: and
WORD: the -  REL: det -  IND: 8 -  LEMMA: the
WORD: seven -  REL: nummod -  IND: 9 -  LEMMA
WORD: dwarves -  REL: conj:and -  IND: 10 -
WORD: are -  REL: cop -  IND: 11 -  LEMMA: be
WORD: in -  REL: case -  IND: 12 -  LEMMA: in
WORD: the -  REL: det -  IND: 13 -  LEMMA: th
WORD: woods -  REL: root -  IND: 14 -  LEMMA:
```

Figure 11. Incorrect Word Relation Tag

```
**********
SentenceID: 13
SentencePattern: 1
Action: are
Agent: Snow White her prince
Purpose:
Cause:
Dobj:
Iobj:
Complement:
Location: in the woods
Time:
**********
```

```
**********
SentenceID: 13
SentencePattern: 1
Action: are
Agent: the seven dwarves
Purpose:
Cause:
Dobj:
Iobj:
Complement:
Location: in the woods
Time:
**********
```

Figure 12. Incomplete Event Models

However, there are also instances when relying on word relation tags alone is not sufficient to properly identify the agents. This is the case for the story text "*The mirror said Snow White is the prettiest in the land.*" and its corresponding word relations shown

in Figure 13 that indicates two (2) *nsubj* tags. Simply assuming that all *nsubj*-tagged instances are *Agents* would mean that the event model will contain *mirror* and *White.*

```
WORD: The - REL: det
WORD: mirror - REL: nsubj
WORD: said - REL: root
WORD: Snow - REL: compound
WORD: White - REL: nsubj
WORD: is - REL: cop
WORD: the - REL: det
WORD: prettiest - REL: ccomp
WORD: in - REL: case
WORD: the - REL: det
WORD: land - REL: nmod:in
```

Figure 13. Multiple Instances of *nsubj* Tags

Sentence pattern #4 assumes the presence of an adjective phrase immediately after the linking verb to serve as a subjective complement, i.e., *very delicious* is the subjective complement for *apple*. The event extraction algorithm initially failed to detect the agent when the input sentence became "*The apple in the basket looks very delicious.*". The extractor cannot properly determine which of the two entities tagged as *nsubj* - "*apple*" and "*basket*", should be the agent. To resolve this, the *nsubj* that is closest to the root, in this case, "*apple*", is treated as the agent of the sentence.

4.3 Identifying the Purpose of an Event

Sentence pattern #1 is used to identify the adverb or prepositional phrase for time/place after the *be*-verb. It works well for straightforward sentences like the one shown in Table 3. However, in the sentence "*He was here for dinner.*", the rule has to be revised such that any prepositional phrase appearing after another adverb or prepositional phrase would be extracted as the value for the *Purpose* field of the event, as shown in Table 4.

Table 4. Sample Event Representation with Purpose

Sentence Pattern	1
Action	was
Agent	He
Purpose	for dinner
Location	here

5. Results and Discussion

The performance of the event extraction algorithm was evaluated by comparing the generated story world graph with those derived manually by a linguist. Five stories were used, with a total of 201 events containing 201 instances for action, 201 for agent, 18 for purpose, 22 for cause, 98 for direct object, 14 for indirect object, 66 for complement, 45 for location and 47 for item that were extracted by the linguist. Precision, recall and F-measure were calculated using the formula in (1), (2) and (3), respectively. The variables are described in Table 5.

$$Precision = \frac{P + L + E}{P + L + E + W} \quad (1)$$

$$Recall = \frac{P + L + E}{P + L + E + M} \quad (2)$$

$$FMeasure = \frac{2\,x\,Precision\,x\,Recall}{Precision + Recall} \quad (3)$$

Table 5. Variables used for counting extraction instances

Meaning	Description
Perfect (P)	Number of extracted instances that are exactly the same as the manual annotation of linguists
Lacking (L)	The extraction is missing some details, e.g., if the extraction got only "*him*" as the direct object, but the linguist annotated "*him, lost children*" as the direct object
Excess (E)	The extraction contains extra details not found in the linguist's annotation, e.g., if the system extracted "*had grown tired of*" vs the manual extraction "*had grown*"
Missing (M)	When the manual annotation extracted an item for a given field in the Event Model, but the system did not, e.g., manual extraction had a value "*now*" for time, but the system has "*null*"
Wrong (W)	The values extracted by the system do not match the manual annotation, including any fields that the linguist may have left blank

The extraction algorithm has a *precision* of 62.33%, representing the percentage of retrieved instances that are relevant; a *recall* of 42.25%, representing the percentage of total relevant instances that are retrieved; and an *F-measure* of 50.36%. While *precision* considers incorrect instances, *recall* considers missing instances of event extraction. Table 6 shows a detailed breakdown of the precision, recall and F-measure values for the individual elements in the event model.

The elements *action* and *agent* achieved the highest performance measures, followed by the *direct object*, with an F-measure of 96.76%, 96.50% and 73.49%, respectively. The elements *purpose* and *complement* achieved the lowest performance, with 0% and 11.27%, respectively. The *time* element, with a recall value of 19.15%, is not always extracted, but when it is, there is an 81.82% chance of correct extraction. The complement is usually misclassified as either a direct object or an action. The purpose and the cause fields are usually interchanged.

Table 6. Performance measures for each element of an Event

Field	Precision	Recall	F-Measure
Action	97.00%	96.52%	96.76%
Agent	95.07%	97.97%	96.50%
Purpose	0.00%	0.00%	0.00%
Cause	35.00%	31.82%	33.33%
Direct Obj	67.52%	80.61%	73.49%
Indirect Object	50.00%	21.43%	30.00%
Complement	80.00%	6.06%	11.27%
Location	54.55%	26.67%	35.82%
Time	81.82%	19.15%	31.03%
Average	**62.33%**	**42.25%**	**50.36%**

Further breakdown of the extraction instances for each event element is shown in Table 7. Because the current algorithm cannot correctly detect all verbs in compound and complex sentences, there is a high case of *lacking* instances (36.23%).

Table 7. Extraction instances for each element in the Event Model

	Perfect		Lacking		Excess		Missing		Wrong	
	Count	%	Count	%	Count	%	Count	%	Count	%
Action	101	48.79%	75	36.23%	18	8.70%	7	3.38%	6	2.90%
Agent	165	79.71%	28	13.53%	0	0.00%	4	1.93%	10	4.83%
Purpose	0	0.00%	0	0.00%	0	0.00%	18	85.71%	3	14.29%
Cause	3	8.57%	4	11.43%	0	0.00%	15	42.86%	13	37.14%
Direct Object	55	40.44%	22	16.18%	2	1.47%	19	13.97%	38	27.94%
Indirect Object	3	17.65%	0	0.00%	0	0.00%	11	64.71%	3	17.65%
Complement	3	4.48%	0	0.00%	1	1.49%	62	92.54%	1	1.49%
Location	9	16.36%	2	3.64%	1	1.82%	33	60.00%	10	18.18%
Time	8	16.33%	1	2.04%	0	0.00%	38	77.55%	2	4.08%

Consider the sentence "*She accepted the invitation and all of them were glad*.", the verb *were* was not extracted.

The *excess* instances (8.70%) is correlated with the *missing* instances of the complement element (100%). Consider the sentence "*The people had grown tired*." The complement, *tired*, is extracted with the verb, i.e., action="*had grown tired*", instead of as a separate element, i.e., action="*had grown*" and complement="*tired*".

The same problem on compound and complex sentences occurred with the element *agent*, wherein the system could not extract the second agent, "*all of them*", in the sentence "*She accepted an invitation and all of them were glad.*", contributing to the 13.53% of *lacking* instances.

Though only few events in the corpus had a *purpose* clause and a *cause* clause, the algorithm still failed to correctly extract from these instances. They are also usually interchanged. Consider the sentence "*They decided to make a crystal coffin for her so that everyone could admire her beauty*." The event extraction algorithm treated the underlined clause as a *cause* element, whereas the linguist considered this as a purpose. This accounted for the 37.14% of *wrong* instances for *cause* and 85.71% of *missing* instances for *purpose*.

In the case of direct objects, they are considered *missing*, such as in "*would not let her go*", when the algorithm failed to tag "*her*" as the direct object because it is followed by a verb and does not conform to the existing patterns for determining a direct object.

The *wrong* instances are attributed to the *missing* complement instances as well. Some variations in a sentence's POS tag make it possible for the complement to be placed under the direct object. In the sentence "*Before leaving, the fairy godmother said she should be home by midnight*.", the extracted fields shown in Table 8. On the other hand, in the manual annotation, the linguist further decomposed the direct object into another event as indicated in Table 9.

6. Generating Story Summaries

Given the story world graph representing the characters and events of an input story, four different ways of generating the summary can be explored depending on the target content. The summarizer can (1) focus on the causal relations that exist between two events; (2) choose events where the main character is involved in, either as an agent or recipient of the action; (3) focus on events where two or more characters interact, or events that highlight the relationships between characters; and (4) focus on describing the setting and the characters in the story.

Table 8. Event with Complement Placed under Direct Object

Agent	fairy godmother
Action	said
Direct object	she should be home by midnight

Table 9. Complement Modelled as Another Event

Action	should be
Agent	she
Location	home
Time	by midnight

If the summarizer adopts the second approach, events where the main character is the actor are selected from the story world graph. The summary starts with the main character's descriptions and relationships with other characters as indicated in the character model. This is followed by the narration of events that the main character did. A constraint on the length of the resulting summary is also specified – it should not exceed half the length of the original story text.

Natural language generation techniques are then used to transform these selected events and event chains from the story world graph into sentences and paragraphs that would comprise the story summary, where the event becomes the verb of the sentence while the agent becomes the subject of the sentence. The remaining elements of the selected event form the direct or indirect object, and prepositional phrase of a story text in the resulting summary. Referring expression generation uses the pronouns stored as part of the representative mention of a character name. Sentence aggregation combines phrases and sentences, that is, instead of generating "*Cinderella ate. Cinderella drank.*", aggregation leads to the sentence "*Cinderella ate and drank.*". Lexicalization is not performed, and instead, the actual words from the original story text that stored in the story world graph are used in generating the summary text.

Given that there is no unified view on what a best summary should contain, we do not aim to produce the best summarization algorithm. Instead, the purpose of our evaluation is to determine how a story summarizer that does not prioritize the causal relations in choosing events would fare against an ideal summary

produced by linguists, while taking into account the use of the story world graph in the summarization process. Precision, recall and F-measures were again used, using equations (4), (5) and (3), respectively, where *Gen* is the number of generated events, *and Ideal* is the number of events found in the human-authored summaries. The values for the *ideal* summaries were calculated by taking the average of the number of events found in the three (3) linguists' individual summaries.

$$Precision = \frac{Instances\ in\ Gen\ \&\ Ideal}{Instances\ in\ Gen} \quad (4)$$

$$Recall = \frac{Instances\ in\ Gen\ and\ Ideal}{Instances\ in\ Ideal} \quad (5)$$

The performance measures for the five generated summaries are shown in Table 10. A precision of 75.43% shows that the extracted events are part of the ideal summary, and of this value, the system has a recall of 31.84%. The story world graph achieved an overall F-measure of 44.78%. While story events involving the protagonist have a high chance of being included in the ideal summary, there are still a number of other events that a human author may consider as important even if these do not involve the main story character.

Table 10. Performance measures for the generated summaries

Story	Precision	Recall	F-measure
Pied Piper	88.89%	14.43%	24.83%
Ugly Duckling	78.79%	59.39%	67.73%
Cinderella	53.13%	20.50%	29.58%
Snow White	78.57%	30.50%	43.94%
Little Mermaid	77.78%	34.40%	47.70%
Average	**75.43%**	**31.84%**	**44.78%**

The generated summaries of *Pied Piper* and *Cinderella* contained very few events, numbering 3 and 6, respectively, thus the low performance measures as seen in Table 10. In other stories where the secondary characters were frequently mentioned, the precision of the resulting summaries went down. This is because there were more events in the ideal summary that focused on describing the secondary characters, their relationship with the protagonist, and events that the secondary characters did to influence the turnout of the story. This also led to low recall, highlighting that the events where the main character is the actor constitutes only a small portion of one linguist's ideal summary.

Another linguist prioritized the selection of events that moved the plot forward, regardless of the role played by the story characters. The ideal summary in this case included events involving other characters that played a role in changing the state of the story world. This also affected the values for recall.

7. Discussion

Using human-annotated events as the reference text for comparison, test results showed that the event extraction algorithm has a precision of 62.33%, a recall of 42.25%, and an F-measure of 50.36%. The extraction of actions and agents achieved the highest precision and recall values, while the cause and purpose achieved the lowest. Problems with the varying sentence structure led to incorrect and missing extraction instances for different event details. The extraction algorithm also encountered difficulty when dealing with clauses.

Similarly, using human-generated summaries as the reference text, our story summarizer achieved a precision of 75.43%, a recall of 31.84% and an F-measure of 44.78%. The absence of a unified view on what should constitute an ideal summary is the primary reason for these results. As seen in our results, different experts have their own criteria in determining what constitutes a good summary. One expert focused on events that she perceived to have greatly contributed to the story's outcome. For an automated summarizer to do the same, the story world graph and extraction rules should include weights that represent the importance of the events to the overall flow of the story. On the other hand, another expert's summary retained the structure of a children's story depicting the introduction, climax and resolution. This requires events to be annotated with an event type, and the causal chain of events to be modelled in the story world graph [19].

8. Conclusion and Future Work

Automatic text summarization has gained popularity in recent years as a means of providing readers a mechanism to reduce the enormous amount of textual materials from online sources, allowing for shorter reading time and more efficient selection process during a research task. For children's stories, summarization facilitates the adaptation of classic stories to make them appropriate for different age groups.

Most automatic text summarizers use word frequencies, sentence length and sentence location as factors in calculating the importance of a sentence in a given document. Such approach, however, may not work for literary forms where important information do not appear in predictable location within the text, and the lack of surface markers and template-like structures cannot help facilitate the summarization task.

In this paper, we described the use of an event extraction algorithm that identifies events and their associated details in the story. These details include the agent or the doer of the action, direct object, indirect object, cause, purpose, time and location). The extracted events are then represented in the story world graph that is subsequently used by an automated summarizer to produce a summary of a given story. Because what constitutes a good summary is subjective, we decided to identify those events where the protagonist is the actor to generate our summaries.

The separation of the event extraction algorithm that produces the story world graph from the story summarizer that generates the actual summary is a major contribution of our work. Future researchers can explore using the story world graph representation with their own algorithms for other NLP tasks, including story generation, and shared storytelling between a conversational agent and a human user. The story world graph could also be used to represent stories in a corpus. This eliminates the need to manually build event models for story generation systems, and instead, utilize knowledge from story world graphs that are built from a corpus of existing stories.

Future work should also look into the extraction of story details from dialogues, which can contain valuable information such as a fact or opinion, the speaker's intention, and a command from one character to another. Issues surrounding the processing of dialogues include missing actor, inquiry questions, short and contextual responses that span multiple dialogue threads.

Because writing styles of authors vary, defining sentence patterns to recognize these writing styles in order to increase the performance of the event extraction algorithm would require considerable time and effort. Future work should consider the use of machine learning techniques to identify the sentence pattern of a given story text to support downstream extraction and summarization tasks.

9. REFERENCES

[1] Radev, D., Hovy, E. and McKeown, K. 2002. Introduction to the special issue on summarization. *Computational Linguistics - Summarization*, 28, 4, 399-408. MIT Press Cambridge, MA.

[2] Meena, Y.K. and Gopalani, D. 2015. Evolutionary algorithms for extractive automatic text summarization. *Procedia Computer Science*, 48, 244-249. Elsevier.

[3] Estrella, J.A., Gelera, C.P., Quinzon, C., Ong, E., Villaruel, M.J. and Sanchez, E.M. 2018. Automated text summarization of research papers regarding the effectiveness of various treatment plans for leukemia. *Philippine Computing Journal*, 13, 1, 21-28, August 2018.

[4] Kazantseva, A. and Szpakowicz. S. 2010. Summarizing short stories. *Computational Linguistics*, 36, 1, 71-109.

[5] Zhang, R. 2013. Coherence-targeted text summarization. *Unpublished doctoral dissertation*, Hong Kong Polytechnic University.

[6] Rockets. n.d. *Helping struggling readers - Vocabulary*. Reading Rockets, WETA Public Broadcasting. www.readingrockets.org/helping/target/vocabulary

[7] Marzano, R. 2010. The art and science of teaching / summarizing to comprehend. *Reading to Learn*, 67, 6, 83-84, March 2010. ASCD.

[8] Manna, R. 2019. *Leveled reading systems, explained*, Scholastic. https://www.scholastic.com/teachers/articles/teaching-content/leveled-reading-systems-explained/

[9] Ang, K. and Ong, E. 2012. Planning children's stories using agent models. In D. Richards and B.H. Kang (eds), *Knowledge Management and Acquisition for Intelligent Systems*, *PKAW 2012*, Lecture Notes in Computer Science 7457. Springer, 195-208.

[10] Adolfo, B.T., Lao, J., Rivera, J.P., Talens, J.Z. and Ong, E. 2017. Generating children's stories from character and event models. In Phon-Amnuaisuk S., Ang SP., Lee SY. (Eds) *Multi-disciplinary Trends in Artificial Intelligence* (MIWAI 2017), LNCS 10607, 266-280. Springer International Publishing AG.

[11] Yu Galan, S., Ramos, M.J., Dy, A., Kim, Y. and Ong, E. 2018. Driving the narrative flow of an interactive storytelling system for case studies. In Geng X., Kang BH. (Eds) *PRICAI 2018: Trends in Artificial Intelligence*, LNCS 11013, 73-81. Springer, Cham.

[12] Charniak, E. 1972. Toward a model of children's story comprehension. *Technical Report*, Massachusetts Institute of Technology, Cambridge.

[13] Lehnert, W. 1982. Plot units: A narrative summarization strategy. *Strategies for Natural Language Processing*, 375-412. Psychology Press.

[14] Christensen, J., Mausam, Soderland, S. and Etzioni, O. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the NAACL: Human Language Technologies*, 1163-1173. Association for Computational Linguistics.

[15] Samson, B.P. and Ong, E. 2014. Extracting conceptual relations from children's stories. In Y.S. Kim, B.H. Kang and D. Richards (eds), *Knowledge Management and Acquisition for Smart Systems and Services, PKAW 2014*, Lecture Notes in Computer Science 8863. Springer, 195-208.

[16] Ang, K., Yu, S. and Ong, E. 2011. Theme-based cause-effect planning for multiple-scene story generation. In *Proceedings of the International Conference on Computational Creativity*, 48-53, April 27-29 2011, Mexico City, Mexico.

[17] He, S., Li, Z., Zhao, H., Bai, H. and Liu, G. 2018. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2061-2071, Melbourne, Australia. Association for Computational Linguistics.

[18] Huang, R. and Riloff, E. 2013. Multi-faceted event recognition with bootstrapped dictionaries. In *Proceedings of the 2013 Conference of the North American Chapter of the ACL Human Language Technologies*.

[19] Mostafazadeh, N., Grealish, A., Chambers, N., Allen J. and Vanderwende, L. 2016. CaTeRS: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the NAACL Human Language Technology 2016 4th Workshop on Events*, San Diego, CA. Association for Computational Linguistics.