

**The Plots of Children and Machines: The  
Statistical and Symbolic Semantic Analysis of  
Narratives**

*Harry Reeves Halpin*

Master of Science  
School of Informatics  
University of Edinburgh  
2003

# Abstract

This thesis presents a method of automatic plot analysis of narrative texts that uses both components of traditional symbolic analysis of natural language and statistical machine-learning. In particular, we are investigating the story rewriting task. In the story rewriting task, an exemplar story is read to the pupils and the pupils rewrite the story in StoryStation, which allows them to concentrate more on diction and grammar than on content creation. However, often in the process of content creation the pupil improperly recalls the story. Our method of automatic plot analysis should allow the tutoring system to automatically analyze the plot of the story and provide relevant feedback to both the pupil and teacher.

In the first phase, annotation and symbolic linguistic methods are used to provide a semantic analysis of the texts. An XML-based pipeline is used for the detection of events and entities in raw text by using multiple levels of automatic annotation and processing, including pronominal resolution. The results of the pipeline are a collection of propositions in a simplified event calculus which are then compared to the exemplar story. This comparison using an algorithm uses a number of techniques to solve the crucial problems of similarity and temporality of the events. The results of this comparison are combined with statistical similarity comparison using Latent Semantic Analysis (LSA) to form a set of features. These features are used in machine-learning to weigh the significance of the presence, absence, and ordering of the event calculus and assign ratings for plot. A collection of children's stories were graded by a teacher and used as the training corpus for the machine-learning algorithm. Machine-learning algorithms like Naive Bayes and Maximum Entropy were successful in separating the "good" stories from the "poor" stories, although fine-grained analysis into "excellent" and "fair" stories was unsuccessful. The machine-learners that used the symbolic event calculus as a feature were more successful than those that just relied on LSA similarity scores, making a case for combining symbolic and statistical models.

# Acknowledgements

I would like to thank all the wonderful people who have made this project possible and my time in Edinburgh truly a year I will never forget. First, I'd like to thank my flat-mates, Kavita and Rob, for creating a community in the flat to sustain me. I'd also like mention Mike, Micah, Ulla, Nina, Panda, and the rest of the ACE ensemble for keeping my heart strong and committed to my ideals of social change, of which this little project is a strange demonstration. I can also never forget the crazy crew at Black Medicine, whose free coffee and humor kept me sane on many a day. Last, this is for Liz, Brian, Dorian, Ken, Gwenn and the rest of my community in States for their love, even if they are so far away. I'd also like to thank all the wild and wondrous people from autonomous villages of the Alps to occupied Universities in Greece that gave me a taste of the Old Country that gives me hope yet.

Academically, first and foremost I'd like to thank Johanna Moore, who gave me the chance and the advice needed to make this project happen. I'd also like to thank Judy Robertson for StoryStation, a tutoring system that will create better writers in our next generation. I'd also like to thank Henry Thompson for the lunches and inspiration, Claire Grover for help with the XML pipeline, and both Mirella Lapata and Alex Lascarides for sage advice. I thank Colin Fraser for proof-reading the entire thesis, and being a friend. I can never forget Cem Boszahin and Denis Zayrek, good friends who made my time both in Scotland and Turkey unforgettable. The entire community of researchers and students here at Informatics, from the labs of Forrest Hill to Buccleuch Place, have created a atmosphere unparalleled in research into this mysterious thing we call "information". Lastly, I'd like to thank my parents, who provided so much care and support during the year. Thanks to my parents, the St. Andrew's Society, and the American Friends of the University of Edinburgh, for providing finances this year.

Most of all, I'd like to acknowledge you, dear reader, for somehow finding this thesis and thumbing through it. May you find something of value in it, and may it open your horizons about what these strange machines we call computers are capable of. May our investigations into computation, of which this one is but a infinitesimal part, illuminate through their workings this how we can raise ourselves up on two legs, open our mouth, and tell a story.



## **Declaration**

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

*(Harry Reeves Halpin)*

To my parents, Harry Halpin Sr. and Rebecca Halpin, for providing the love and care that made this MSc. possible here. I hope I made you proud.

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Literature Review</b>	<b>6</b>
<b>3</b>	<b>Methodology: Symbols, Statistics, and the Frame Problem</b>	<b>24</b>
<b>4</b>	<b>Pipeline Annotation</b>	<b>36</b>
<b>5</b>	<b>The Plot-Comparison Algorithm</b>	<b>47</b>
<b>6</b>	<b>Machine-Learning</b>	<b>60</b>
<b>7</b>	<b>Results and Evaluation</b>	<b>70</b>
<b>8</b>	<b>Conclusions</b>	<b>86</b>
<b>A</b>	<b>Exemplar Story</b>	<b>95</b>
<b>B</b>	<b>Sample Stories</b>	<b>103</b>
<b>C</b>	<b>Sample Output from XML Pipeline</b>	<b>106</b>
<b>D</b>	<b>Statistical Tests of Reliability</b>	<b>122</b>
	<b>Bibliography</b>	<b>125</b>

# Chapter 1

## Introduction

The power of a text is different when it is read from when it is copied out. Only the copied text thus commands the soul of him who is occupied with it, whereas the mere reader never discovers the new aspects of his inner self that are opened by the text, that road cut through the interior jungle forever closing behind it: because the reader follows the movement of his mind in the free flight of day-dreaming, whereas the copier submits it to command.

*Walter Benjamin, "The Story Teller"*

### 1.1 Goal of Project

The goal of this project is to address how one can use both techniques from symbolic natural language processing and statistical machine-learning to tackle the problem of plot analysis in a collection of stories rewritten by children. These stories were collected from classes at Methilhill and Torbain Primary Schools in Kirkcaldy, Scotland by Judy Roberston. The children were told a story by a storyteller, and were asked to rewrite the story in their own words, all of which are variants of a story called "Nils's Adventure", a story from "The Wonderful Adventures of Nils" (Lagerlof, 1907). A transcript of the story as told by the storyteller is in Appendix A. The stories were written by children ages 10-12 from a broad range of reading levels. We hope to develop a model of plot analysis based upon how teachers would rate these stories. The models of plot analysis investigated in this thesis are part of a wider project to

enhance the automated tutor StoryStation (Roberston and Wiemar-Hastings, 2002). Techniques explored here show that pursuing both formal symbolic processing and statistical machine-learning, complimentary approaches to natural language processing, can deliver more promising results than either approach by itself.

*Symbolic natural language processing* denotes the traditional approach to computational linguistics that uses formal rules and levels of abstraction to operate over symbols. Traditional artificial intelligence, planning, and theoretical linguistics (especially “deep” representations of semantics) fall under this rubric. Whether these symbols denote actual words, grammatical categories, lexical semantics, or discourse phenomena is not an issue, since formal symbolic natural language is a methodological approach to computational linguistics, not a domain of inquiry within computational linguistics. Examples of this approach are using rules to resolve pronominal anaphora, rules to turn a sentence into a series of non-recursive chunks, and lists of backward and forward-looking centers. This project will analyze a corpus using some of the above techniques to achieve a formal representation of the plot of a children’s story from raw text.

*Statistical machine-learning* denotes the approach that has gained more and more currency in the last decade by using statistical regularities in the corpora, instead of explicit rules, to solve problems in computational linguistics. In particular, this approach often involves not only finding the statistical regularities of words or other linguistic features in the data, but then building a probabilistic model of the data using machine-learning algorithms. Comparing texts according to their similarities in word distributions or building a maximum entropy model of the linguistic features of a text would fall under this methodology. Both of these techniques will be explored in this paper as ways of solving some problems that plague the symbolic approach while retaining its strengths.

## 1.2 Project Hypothesis

This project applies hybrid computational techniques to the evaluation of plot in the story rewriting task of the StoryStation software. It is hypothesized that these tech-

niques can result in an analysis of children's stories that significantly approximates the feedback given by teachers on plot analysis for the story rewriting task, and can give such analysis back either in the form of a plot grade to a teacher or online feedback to the children during the rewriting process. Teachers have suggested that automated plot analysis is the one feature of StoryStation they would most like to see working.<sup>1</sup>

### 1.3 Methodology

Our methodology is a combination of statistical and symbolic analysis of a small corpora (103 stories rewritten from the same original story) of children's stories that were produced and written by children themselves during the story rewriting task. We reason that by a mixture of both these techniques a reasonable approach can be found that bridges the gap between statistical and symbolic natural language processing, while avoiding the faults of making the algorithm either excessively data-driven (too dependent on large and accurate corpora) or overly knowledge-driven (dependent on a large amount of human work and unscalable). One corollary of this approach is that small data-sets can be made amendable to statistical analysis if a moderate amount of knowledge, such as automatic or human annotation, is included in the data prior to building a probabilistic model using machine-learning techniques. Observations from the current literature in computational narrative analysis will often be compared, favorably or unfavorably, with examples from this story.

### 1.4 StoryStation and the Story Rewriting Task

StoryStation is an automated tutor for writing that currently features a number of automated agents that do a number of tasks, such helping the pupil with spelling, and providing an age-appropriate thesaurus. These agents are represented as animated characters that float on the screen of the pupil and offer advice. Their colorful icons were created by pupils themselves (Roberston and Wiemar-Hastings, 2002).

The "Pinky the Plot Analyzer" agent is currently not working in Storystation due

---

<sup>1</sup>Personal communication with Judy Roberston.

to a lack of both a theory and implementation of automated plot analysis. Ideally, we would like this agent to be able to provide some type of score for the plot of the story, and then give advice to the student on whether or not they should work more on the plot of the story. Ideally, a more fine-grained level of analysis in which the agent reminds the pupil of particular pieces of the plot they may have left out (like missing characters and events) would be highly appreciated. The ultimate goal is to have the agent be a type of general story-understanding agent that can, in principle, inspect the plot of any story. This level of functionality is outside the goals of this thesis. For this thesis, we are going to focus on one particular type of plot feedback that we hope will be instructive for further development: rank the plot of the rewritten stories for the *story rewriting task*.

In the story rewriting task, an *exemplar story* is read to the pupils and the children rewrite the story in StoryStation, which allows them to concentrate more on diction and grammar than on inventing a plot. However, if the pupil forgets or gets wrong significant parts of the plot, or entirely forgets the story, this sabotages the writing process for the pupil, and may frustrate further efforts by the pupil to focus on building their writing ability. Having an intelligent agent to comment on the plot structure would take the burden off the pupil and increase the value of StoryStation for the pupil. Also, the teacher may want to be alerted to students that may need help with their plot. For example, the agent could tell a pupil who has gone considerably astray in their plot rewriting to re-read the story or ask the teacher for help.

## 1.5 Overview of Thesis

The remainder of this thesis will present the argument for using a hybrid symbolic and statistical approach to modelling stories in the context of the wider literature, and then present our particular technique for solving the problem for the story rewriting task. In Chapter 2 (Literature Review), we present an interdisciplinary literature review of children's narratives and computational models of stories that starts with early influential research and ends with some current approaches. Due to the diversity of disciplines and approaches to the problem, at the end of this chapter we will synthe-

size a vocabulary and approach to stories especially suited to the problems. In Chapter 3 (Methodology: Symbols, Statistics, and the Frame Problem) we argue for the hybrid approach in a more philosophical and model-theoretic way, and then investigate some recent developments that we take as inspiration for our own work. We also detail a practical example using our corpus showing the limitations of using only statistics to analyze our problem. Then we progress from background and methodological issues to our experimental design. In Chapter 4 (Pipeline Annotation) we describe the process we use to automatically add a large amount of traditional symbolic knowledge to our corpus in the form of XML mark-up. In Chapter 5 (The Plot-Comparison Algorithm) we present the difficult problem of comparing plots, and present a number of variants of a basic algorithm for solving this problem. In Chapter 6 (Machine-Learning) we phase into the machine-learning portion of our experimental design and further detail how this approach solves a number of potential problems. In Chapter 7 (Results and Evaluation) we present our final results for a number of algorithms. In Chapter 8 (Conclusions) we draw a number of insights from our work on this project and make tentative plans for future research.

## Chapter 2

# Literature Review

Culture is made up of stories: stories of words, of creation, of destruction, of science, of numbers, of concepts, of musical conventions, of aesthetic expectations, of solutions for problems, of answers to questions. Stories write our lives: they set the expectations whenever we make a choice, however free we may be in selection. At the same time, it is our re-telling of them that reproduces them, keeps them alive, makes them what they are, as they make us. Terms in language and elements of stories work only because we hold them in common; writing these stories and creating the languages to describe them is the ultimate collective activity, is the common denominator of all social activity. To be free of stories and the languages they are told in is impossible in human relations - the universe we know it is not only described by stories, but exists as stories.

*Anonymous, "Hunter/Gatherer"*

Stories have been studied in a formal framework since the dawn of cognitive science and artificial intelligence. Researchers from a wide variety of backgrounds, including anthropologists interested in folk-lore and computer scientists interested in natural language understanding and generation, have applied their wits to the problem.<sup>1</sup> The goal of this section is to synthesize a terminology and a number of differing levels of looking at stories with the goal of creating a conceptual and symbolic framework in which to tackle the problem of automated analysis of plot. A number of approaches to

---

<sup>1</sup>Let's not forget literary critics and theorists, although for purposes of length we will constrain our study here to look primarily approaches which at least consider themselves "scientific" or "formal", although upon closer inspection more insight may be perhaps gained from the experiences of teachers and writers than the often ad-hoc formalisms of the intentional sciences.

the symbolic analysis of stories will be divided by discipline and presented in roughly chronological order, with their claims evaluated in turn. Whatever concepts do justice to stories should be kept for our project if possible, but whatever failings they have should also be analyzed so the current project does not repeat the mistakes of the past.

## 2.1 Towards a Science of Stories

Although the work in symbolic analysis of stories goes back far, most view the structuralism of Claude Levi-Strauss as pivotal(1963). Building on the foundations of linguistics of Saussure, he notes that stories, and in particular the domain of myths, can be thought of as their own “language”, since there are an astounding number of similarities in stories across cultures. He argued that such similarities were not at the level of content (such as particular characters and events) but at the level of structure. While the content exists in *synchronic* or “non-reversible time”, because it must be read through in a particular order, the structure exists in *diachronic* or “reversible time” that exists separately from content since it is not dependent on specific events that unfold in one direction in time. Stories can then be thought of as having a particular macro-structure akin to the structure of a sentence. As evidence of this structure, he notes that stories can be paraphrased and translated, and their structure should remain the same. He hypothesized that myths consist of essential elements or “mythemes” that are put together according to certain rules. For example, in his famous study of myths he forsakes the actual plots and characters to find abstract mythemes of “blood relations which are overemphasized”, “monsters being slain”, and so on. He claimed that to tell a myth one has to focus on the temporal order of synchronic time, but that to understand a myth the structure, the diachronic dimension, should be analyzed. However, for our purposes and the purposes of history his analysis of mythemes seems more akin to literary interpretation than a conceptual framework that may be exported outside of the limited domain of myths. As shown by research into narrative development, and as we explore later, the division between structure, content, and time is not so easily made(Hickmann, 2003). From his imaginative mythemes, one can conclude that studying form is harder than studying content, and that our study of the plot, should

strive not to find universal structure across all children's stories but focus more on the content of each particular story.

The next major advance was the work of Propp<sup>2</sup> in his "Morphology of the Folk Tale", which was in turn popularized by Claude Levi-Strauss(1968). By "morphology", Propp meant the "description of forms" of semiotics. Propp focused his work on finding out the common components of fairy tales, and he defined the notion of a stock character (such as the "Hero", the "Villain", the "Helper") and the idea that actions can be categorized as orderly functions of the narrative (such as "Preparation", "Complication", "Recognition", and "Return"). His research set off a flurry of activity in similar fields, such as the work of Colby that used a similar framework to analyze Eskimo folktales(1973). While his specific categories are far too numerous for anyone to know or use, his division into actions and characters, and his insight that each action performs a vital part in the unfolding of the story, are useful. However, from his work one learns that the division of story elements into a finite set of categories seems to be futile, or at least hard to do in a complete and systematic way that is portable between different story domains.

Bartlett completed the first comprehensive study of one story, the famed "War of the Ghosts"(1932). He was interested in how people memorized various parts of the story in recall tasks. Bartlett noted that generally unfamiliar things were not recalled while story elements that were familiar from other stories were sometimes added to the story during recall. He proposed that these systematic distortions reflected the use of *schemas* in storing and retrieving information. A schema for a story consists of a setting, a theme, main character or characters, causal and temporal chains of events, and a resolution. As proof of the difference between causality and temporal chaining of events, he also noted that there was significant difference in recall between causally-linked events and temporally-linked events, with causally-linked events being more easy to recall. Apparently, parts of the story that made use of already familiar causal relations existed in a "schema" for the story. His focus on causal relations has led to a focus on causation and planning in narrative analysis ever since. From his study, we gain some fundamental and fairly portable categories for analysis: char-

---

<sup>2</sup>Which while published in 1928 in Russian was not translated into English until 1958.

acters, events, setting, theme. However, in children's stories the difference between causality and temporal order, especially in stories of an imaginary nature, may not be so straightforward. In general, the question of how to properly formalize a schema into a computational system has never been solved, leading to what is called traditionally the "Frame Problem" in artificial intelligence.

## 2.2 The Rise and Fall of Story Grammars

The next breakthrough in story analysis was the seminal "Notes for a schema of stories", in which Rumelhart used a *story grammar*, a series of production rules, to describe the structure of stories as a context-free grammar(1975). He claims that "just as simple sentences can be said to have an internal structure, so too can stories be said to have an internal structure. This is in spite of the fact that no one has ever been able to specify a general structure for stories that distinguishes the strings of sentences which form stories from strings which do not"(1975). Rumelhart attempted to define these rules for a story structure, citing Propp as a formative influence yet clearly proceeding in the tradition of Levi-Strauss, although using less imaginative categories and rules. His production rules attempt to show how the idea of "well-formedness" carries into stories, separating a story from "little more than a string of sentences", and that just as phrases serve as arguments for sentences, "sentences themselves can serve as arguments for higher predicates"(Rumelhart, 1975). While the spirit of Rumelhart's analysis closely follows production rules of context-free grammars for sentences, each story actually has a sentence as a terminal symbol. Non-terminals are groups of sentences. Here's a sample of his story rules:

Story  $\Rightarrow$  Setting + Episode

Setting  $\Rightarrow$  (State)\*

Episode  $\Rightarrow$  Event + Reaction

Event  $\Rightarrow$  (Episode OR Change-of-state OR Action OR Event + Event)

Reaction  $\Rightarrow$  Internal Response + Overt Response

Internal Response  $\Rightarrow$  (Emotion + Desire)

Overt Response  $\Rightarrow$  (Action OR Attempt\*)

In addition, Rumelhart defined a parallel discourse-level “semantics” to every rule, consisting of a set of propositions that connect the events. For example, for the production rule of “Story  $\Rightarrow$  Setting + Episode” there is the parallel semantic rule of “ALLOW(Setting, Episode)”, which means that “the setting forms a structure into which the remainder of the story can be linked”(Rumelhart, 1975). However, not only do the semantic rules seem to be only restatements of the production rules in propositional form, they are very vague in application, such as when Rumelhart mentions that “ALLOW(Setting, Episode)...plays not a integral part in the body of the story and under certain conditions can be eliminated”(1975). Almost all later research into story grammars did away with having a parallel propositional semantics, and concentrated on the “syntax” of the story rules.

There are several advantages to the formal analysis proposed by Rumelhart. It is much more grounded and less interpretative than the anthropological analysis of Levi-Strauss, and consists of a series of a few rules as opposed to the huge lists of categories that Propp adopts. They also provide complete coverage for all the sentences in a story, showing how higher-level structures tie a stories together and distinguishes a story from a set of arbitrary sentences, and how they supposedly generalize to all “well-formed” stories instead of a particular domain of stories.

Yet story grammars have a number of significant problems. First, both Rumelhart and his followers admit that their grammars only work in stories with a single protagonist. While Rumelhart claims his categories are “syntactic”, behind his syntax lies intentional terms such as “Desire” and “Plan”. They do not identify or use as arguments to their structure content-related features such as particular characters and events in the story, and so neglect the actual content of the plot. This approach is incapable of detecting the presence or absence of a particular character in an event. They also limit the component sentences of their story to those which can be reduced to a single proposition. Given that many sentences contain multiple propositions through devices such as connectives, this is unsatisfactory. Also, story grammars have no way to situate their propositions in characters, such as when one character believes a propo-

sition and communicates this proposition to another character. Lastly, the number of syntactic categories are upon closer inspection fairly arbitrary and so hard for either a computer algorithm or a human annotator to discriminate. Who can clearly tell the difference between an “Event” and a “Change-of-state”, or an “Action” and a “Pre-action”? This is not helped by the paucity of Rumelhart’s own definitions of his terms. For example, Rumelhart defines “event” as “the change of state or action or the causing of a change of state or action”(1975). If a human has difficulty reliably distinguishing these categories, it would be difficult to design an algorithm capable of extracting these categories.

After Rumelhart’s initial publication there was a brief upsurge in story grammar research, most notably Mandler and Johnson’s “Remembrance of Things Parsed”, where they dramatically reconfigured Rumelhart’s original grammar to parse Barlett’s “War of the Ghosts”(1977). Mandler and Johnson continue to apologize that their “grammar is far from complete” and only works for stories with a single protagonist.<sup>3</sup> While Mandler and Johnson’s ultimate goal was to discover a theoretical explanation for differences in recall of narrative, our primary interest will be in their modifications to Rumelhart’s syntax.

All sentences (terminal nodes) are categorized as either “States” or “Events”, and then connected by three types of relationships: “And”, “Then”, and “Cause”. The combinations of these basic structures leads to higher categories, such as “Beginning”, “Development”, and “Ending”. Some of the rewrite rules for Mandler and Johnson are:

Story  $\Rightarrow$  Setting and Event Structure

Setting  $\Rightarrow$  State\* (And Event\*) OR Event\*

State\*  $\Rightarrow$  State ((And State)<sup>N</sup>)

Event\*  $\Rightarrow$  Event (And OR Then OR Cause) Event)<sup>N</sup> ((And State)<sup>N</sup>)

Event Structure  $\Rightarrow$  Episode ((Then Episode)<sup>N</sup>)

Episode  $\Rightarrow$  Beginning Cause Development Cause Ending

For the sake of brevity, only about one-fourth of the rules have been listed. The

---

<sup>3</sup>Rumelhart attempted to develop multiple protagonist grammars, but met with little success(Rumelhart, 1980).

number of rules has increased greatly when adopting to a more complex story, and criticisms of Rumelhart's grammar still apply to Mandler and Johnson. However, the insight of this approach is that they apply only two types of categories to surface realizations, "Event" and "State". These rules also manage to capture some of the temporal and causal relations in a story by their use of "And", "Then", and "Cause", and form these into higher-order events, such as "Beginning", "Development", and "Ending". The baroque and numerous categories of Propp becomes a problem again. Great emphasis was put on this type of higher narrative structure by Mandler and Johnson, as "even the simplest well-formed story will have an ending" and they even attempt to categorize dramatic flourish as "Emphasis"(1977). Ultimately, the most telling observation of this approach is that while they simplified the syntax on the sentence level, serious problems still plague them as they attempt to define a hierarchical macro-structure. If a story grammar's rules need to be rewritten and modified for each story, and the number of rules grow in proportion to the length of the story, then clearly the entire approach to analyzing stories through using rules is deficient.

Story grammars also came under critique in their own time, but for different reasons than those detailed. First, a detailed critique of story grammars was delivered by Black and Wilensky, who noted their inherent inability to contain "embedded stories" or distinguish sub-plots(Black and Wilensky, 1979). While there was a vigorous defense, researchers moved into more informal paradigms such as scripts(Mandler and Johnson, 1980). Wilensky also noted that what was being measured by story grammars was not whether or not a story was "well-formed" but whether it was coherent(1982). He argued that the actual "storiness" of a story was more related to whether a story is interesting than coherent, and the interest of a story cannot be deduced from its structure alone. Instead of "story grammars", he develops a theory of "story points" that defines a story by its having "a certain content" instead of a "certain form". The crucial element of content for a story is its "poignancy". Wilensky formalizes poignancy by showing how the goal-related planning of multiple characters leads to situations that are "poignant", mainly through the conflict and subsumption of the goals of the various characters(1982). He also notes that what he defines as "points" should be extended to cover "enigma, novelty, tragedy, and, humor". In spite of these ambitious goals, his

theory fared little better than story grammars since it relied on formalizing very fuzzy and vague “points”. Yet, elements needed to uncover these relations are often implicit or not present in the surface text at all. Wilensky’s approach of focusing on crucial “points” directly addresses the deficiencies of analyzing a story purely on the basis of structural coherence, but at the same time provides no practical criteria to identify these crucial events.

## 2.3 Schemas, Scripts, Frames, and Stories

Daunted by the failure of story grammars, story researchers began to work on a number of models that were popular in other fields such as artificial intelligence and cognitive psychology. Almost all of these models can trace their heritage back to the schema theory first articulated by Bartlett(1932).

A *schema* is a hypothetical mental structure that stores generic concepts. These generic concepts are abstract or prototypical knowledge of regularities that have been discovered through repeated exposure to many particular instances. Schemas further organize new information and retrieve information. Given schema theory as a basis, researchers tried to discover what schemas are employed by children to recall narratives(Trabasso and Sperry, 1985). Narrative schemas can include the temporal ordering of events and causal relations between events. During story recall, missing events are inferred to fill in omitted information, and events may be reordered to correspond to a different time sequence than that presented by the actual narrative, in order to match the internal schema(Stein and Trabasso, 1982).

*Scripts* can be considered schemas for common-place situations. A schema is a general method of constructing, storing, and retrieving “common-sense” information. A script is the information stored in the schema for a situation that is invoked when another situation of that type is encountered. Scripts are primarily used in psychological literature and artificial intelligence to explain human expectations and ability to predict events, and have also been used to explain the coherence of narratives(Schank and Abelson, 1977). Their most important feature is that they make explicit the unmentioned details and assumptions of the story. Through various planning and inference

techniques, inferences can be made from a script to “make sense” of a given situation. An example is “Tom goes inside. He sits and makes an order of food”. A script is activated by the reader to make the logical deduction that “Tom is in a restaurant” since the script would contain the knowledge that “Restaurants allow one to sit down and order food”.

Scripts can be formalized as *frames*, which represent knowledge as a structure of named slots that are instantiated to represent a particular story. Slots accept a range of values and often have default values (Minsky, 1975). In practice, frames tend to be computationally implemented as a knowledge-base of world knowledge which may occupy the variable slots in the frame. These variables can be determined by the operation of an inference engine. A narrative can then be considered a frame or series of frames, with the content and explanation of a story determined by a particular narrative in conjunction with an inference engine. Narratives have also been computationally modeled using *semantic networks*, which are a graph whose relations between nodes (entities in the narrative) are labeled with a semantic relationship. Trabasso did work where he attempted to “weigh” certain relations of the network to determine importance of an event in the story (1985). All of these models, from the informal idea of schema to the more formal frames and semantic networks, attempt to capture the content of the story. This is in distinct contrast to the structuralist approach of story grammars which attempt to capture the form of the story.

However, there are a number of problems with each of these formal representations. From a theoretical point of view, even though they focus on characters and events in the story, these implementations are usually static and have trouble representing the dynamic flow of time in stories. Second, the actual components of the script, frame, or semantic network have to be hand-coded, consuming a large amount of human time. This approach is also “brittle” and unscalable, since a semantic network for one story cannot be used by another, and the scripts and frames are very particular to a certain situation in the narrative. For the purposes of our project, since scripts and frames are concerned mostly with the formalizing of large numbers of facts in order to build correct inferences, one should ask how important common-sense inferences are in children’s stories? Indeed, in “Nils’s Adventure” events such as boys jumping on the backs

of storks and becoming elves are common-place, which is far from common-sense inference. The revelation of a causal link between Nils's finding and throwing away the coin and the disappearance of the city is crucial, yet far from common-sense. Indeed, research in children's narratives shows that unusual events are crucial to most children's stories (Hickmann, 2003). While common-sense inference is useful in mundane situations such as restaurants, its ability to generalize such reasoning to interesting narrative situations is questionable. However, some form of frames and reasoning that are appropriate to stories would be useful.

To further complicate frame-based approaches, there are several fundamental problems in artificial intelligence that become apparent when using frames, and these are often referred to as "The Frame Problem" (Morgenstern, 1996). The classical *Frame Problem* asks how can any set of propositions or frames fully represent a dynamic world, such as the temporal flow of events in a children's story? This is closely related to the *Quantification Problem*, which is concerned with the question of how anyone can choose the right number of things to represent without formalizing everything? Lastly, the *Ramification Problem* notes that things that are formalized often have unexpected and unpredictable interactions which are left out of the model. These problems are deep, and their solutions not obvious, if such solutions exist. We will delve into these issues at much greater length in Chapter 3. Yet, from this entire history of story analysis, it is clear that we need to formalize the events, characters, setting, and the temporal and casual flow of a story, yet avoiding burdening a human annotator with deciding exactly what to formalize.

## 2.4 Computational Narration

While many of the models previously mentioned have their historical roots in cognitive psychology, research into story understanding by computers began with the doctoral dissertation of Eugene Charniak, "Towards a Model of Children's Story Comprehension", under the supervision of Marvin Minsky (1972). Charniak was especially interested in children's stories since he felt they would be easier than other types of text for an artificial intelligence system to understand. Many still agree with his early ob-

servations, making story understanding a fertile research area for semantic or “deep understanding” AI systems. First, Charniak separates the task of story-understanding from that of natural language understanding from raw text, assuming that the problem of extracting semantic propositions from raw text would be solved independently at some point. As input, his system took a “deep representation” of the story, essentially semantic propositions, that had to be hand-coded. Second, he viewed the task of story-understanding as one of logical reasoning by applying the story to a background of “common-sense” facts, like scripts. Ever since then, these two assumptions have governed almost all research in computational narration. This includes the field of story generation, as started by the program TALE-SPIN that took a database of propositional facts about a story and tried to order them into a coherent story (Meehan, 1976).

Recently, there has been much research on how narratives can integrate AI planning techniques into both story understanding and generation (Christian and Young, 2001). Most of this research echoes the work of de Beaugrande, who states that a story can be considered the progression from an initial state to a final state through a process of planning and problem-solving jointly committed by the narrator and the reader (1980). Recent research in story generation shows that various parts of a story can develop through generating uncertainty in the reader. The “climax” can be explained as the moment of maximum uncertainty in the reader, caused through creating a large number of inferences. The story ends as the narrative resolves the inferences (Bailey, 1999). Yet, at this stage in development the research into planning and stories seems mostly irrelevant to the story rewriting task, due to two assumptions. First, it would be infeasible for us to handcode each student’s story into a “deep semantic representation” suitable for processing by these systems. Second, we do not wish to employ at this stage a database of common-sense knowledge or inference for reasons already mentioned. Lastly, we believe the focus on planning is often at the expense of the actual events in the plots, and it is the extraction and labeling of these that seem more directly relevant to the story rewriting task. At higher-levels of interpretation the child’s understanding of the plans of the characters may be useful, but in most of our stories the plans are either transparent (such as “Let’s have an adventure”) or simple (“If you want to fly, get on the back of the flying stork”). For these reasons, against the grain of

current research, a focus on planning may only overly complicate issues at this point.

## 2.5 Discourse Psychology

Despite these difficulties in the field of artificial intelligence, recent work in the psychology of human discourse has provided a solid analysis of the levels of processing a narrative. While there are a number of competing psychological models of discourse, one of the most widely theories is the “construction-integration” model of Kintsch and van Dijk (1978), which is summarized by Graesser (2003). In this theory, there are six levels of cognitive representation. Also noted will be computational analogies to the levels:

1. *Surface Code*: This is “most, if not all, of the exact wording and syntax of the explicit text”(Graesser, 2003). For our purposes, this will be the exact text produced by the pupil, which in future research will be automatically recorded by StoryStation.
2. *Textbase*: This is “the meaning of the explicit propositions in the text, but in a stripped down form that glosses over the details of the surface form”. In essence, textbases may be formalized as “a set of propositions...that contains a predicate (main verb, adjective) that inter-relates noun-like arguments”(Graesser, 2003). Clearly, this level should be formalized into some type of logical calculus built upon propositions, as most of traditional story-understanding programs have done.
3. *Situation Model*: A “mental micro-world of what the story is about”, including “the spatial setting” and the “chronological sequence of episodes in the plot”(Graesser, 2003). On this level things are less clear than propositions or even frames. Elements like characters, their plans, causation, and temporal ordering of events make their appearance on this level. For our purposes, we will venture into this level only when appropriate, attempting to minimize our assumptions.

4. *Thematic Point*: “The moral, adage, or main message” of the story (Graesser, 2003). While clear in some stories, such as Aesop’s “Tortoise and Hen” fable, in our particular story of “Nils’s Adventure”, the moral is far from clear. Is it not to throw away things like strange green coins which may be useful later? Or to always have adventures, but be home at a reasonable hour? While this level may be the most interesting from a pedagogical viewpoint, it is a matter that is currently beyond understanding with computation, if it should be dealt with computationally at all. However, a grasp of the overall “point” of the story would be crucial in correctly grading the story, and there is probably a relation between this “point” and the presence of various events and characters.
5. *Agent Perspective*: This level deals with what character the reader of the story views the story from. Often readers can give an recollection of the story through a particular agent, such as one of the characters. The reader may make differing judgments about whether to re-tell the story using either first or third person. Surprisingly agent perspective is not a problem for our current corpus, since the all the pupils happen to write from the third person omniscient narrator perspective, with very rare switches into first-person. While automatic identification of agent perspective would be a useful advance, for our purposes we will assume the child takes on a third-person omniscient narrator perspective.
6. *Genre and Pragmatic Context*: While the genre of a story and its pragmatic use are interesting, for our purposes these levels are set: the genre is a children’s story and the pragmatic context is the pedagogical tutor StoryStation.

Clearly, decisions have to be made about what levels our plot analysis and comparison system needs. For sheer statistical analysis, models generally stay on surface code. However, if we wish to do anything more complicated we need to at least attempt to acquire textbase from text, automatically if possible. Given the constraints and qualities of our propositional forms, we need to build a situation model with minimal assumptions from the exemplar story, and then attempt to extract the situation model of the pupil’s rewritten story. Finally, the two situation models need to be compared by some type of matching process. The higher levels of discourse, such as the “moral” of the

story, are currently beyond the scope of this project, although they will factor into the ratings given to the stories by the human raters.<sup>4</sup>

## 2.6 Children and Narration

It is interesting to note that very little of the work into understanding children's narratives has taken any of the relevant psychological and discourse literature of children's narratives into account. In the tradition of Charniak(1972), children's stories are either viewed as the same as adult stories or as simple adult stories. This is surprising, given the varied and complex linguistic development a child is expected to go through that differentiates their story writing and story understanding ability from that of adults. While most story-understanding systems claim to avoid this by dealing only with "deep representations", since we need to deal with surface text and the characteristics of children's linguistic development around the age of 10-12 (the age of most users of StoryStation), these differences need to be taken into account. Unfortunately, most artificial intelligence researchers have lacked clear exposure to the discipline of first-language acquisition. The recent publication of Hickmann's "Children's Discourse: Person, Space and Time across Language"(2003) makes much of this current research accessible. Instead of engaging in a full-scale review of children's narrative development, the claims and research of this field will be mentioned as issues relevant to the design of a story-comparison system are noted.

Especially interesting is the fact that children *develop* the ability to use narrative over time. This development process could help us discover the essential elements of a discourse. There are generally agreed upon levels of narrative development, whose extremes we present here(Applebee, 1978; Hickmann, 2003):

1. *heaps*: A series of unrelated referents and events. This shows the basic structure of stories to be "bare" referents and events, even if they are unconnected.
2. *sequences*: A series of events linked by a single referent, usually with some type of similarity relation between events. This could be considered computationally modeled by centering theory.

---

<sup>4</sup>See Chapter 7 for details on ratings.

3. *focused chains*: The focus now follows a single character, the main character.
4. *narratives*: Expansion of the focus to other elements of the story in an orderly fashion, as well as elaboration on themes.

Although by the age of ten the child should be fully able to use narratives, these levels help us decide what the “primitives” for our symbolic plot analysis system should be (referents and events). Our corpus shows this variance in story rewriting, with stories ranging from unrelated events and characters to an understanding of the causal flow of events and point of a story.

The conflict between focusing on form or on content in children’s stories has haunted the entire field. To ground these terms in language acquisition, *coherence* is defined as the hierarchical and structural “properties of the content in the discourse” (Hickmann, 2003), while *cohesion* is defined as the use of linguistic markers to bind sentences together into a linear discourse unit instead of a series of unrelated sentences. Coherence is usually thought of as structure that works on a higher level of organization, while cohesion works on the level of adjacent sentences and clauses. Work in language development show that these two are not easily separable and develop together in children (Hickmann, 2003; Applebee, 1978). Indeed, the use of linguistic indicators and centering are good predictors of coherence, and both cohesion and coherence are influential in creating a “well-formed” story. Given this, it is no wonder that efforts like story grammars that focus on only coherence at the expense of cohesion fail to adequately represent stories. What is needed is a balance between cohesion and coherence (content and form) in a model of stories. However, for the time being this is still an area in which not much work has been done, so we focus on the more computationally tractable and reliable area of cohesion (since extraction of entities from raw text is possible, if far from fool-proof). We leave for later research the more difficult, but ultimately needed, coherence relations. It is our belief that these higher-level coherence relations, such as the presence of a clearly delineated beginning and ending of a story, are not understood well enough to be symbolically modeled.

## 2.7 Synthesis

It is clear that the study of narratives is by its nature interdisciplinary and so to some degree fractured. A number of guidelines should be gathered at this point for the story rewriting task. First, what level of formality can we achieve that does the most justice to the data while making the least assumptions? The analyses of structuralism and story grammars purposely avoid the content of the story that the pupil is recalling and rewriting. Evidence from children's narratives shows that the form and content of the story cross-cut each other, unable to be fully separated (Hickmann, 2003). While there have been many theories about hierarchical structuring of narrative, current theories are either unreliable or not generalizable to all stories. A formalism that focuses on characters, events, and temporal flow of the plot would be better suited for our project, especially given the simplicity and small size of our corpus of stories. In essence, we are returning to the famous minimalist definition of narrative by Labov (1972):

“A narrative consists of at least two events presented in a temporal and causal sequence.”

A symbolic language is still needed to capture this minimalist definition. The question is in what tradition should we work: one that emphasizes planning or one that focuses more on the temporality of events (Beaugrande, 1980). Again, we choose the latter option, since a more complex formalism based on reasoning and planning would be difficult to extract from raw text. Research in children's language shows that such “common-sense” reasoning traditionally used by AI planning does not do justice to children's stories that are by nature unusual and defy “common-sense” reasoning (Hickmann, 2003).

We need to choose the level on which to categorize the elements of the narrative. In the tradition of Propp and Wilensky we could begin by functionally describing the different elements of the story (“The Main Character”, “The Climax”, and so on), yet these fall into the classic problem of creating too many fairly arbitrary categories of story elements (Wilensky, 1982; Propp, 1968). In order to keep the kinds of categories minimal, we consider the plots of stories to have only two different categories, events and entities. Entities would include animate characters such as “boy” and “geese”, as well as inanimate objects such as “coins” and “cities”. Entities can also in a wider

sense be used for any referent, although in children’s stories one would expect physical entities. Events would usually be composed of the interactions of entities, such as the “boy throwing the coin”. Clearly, it is composed of two entities, a “boy” and a “coin” connected by an action, the “throwing”. As this show, events can be thought of as predicates, and entities as arguments to these predicates. The predicate will usually be described by a verb, and its entities as the subject and objects, which are usually nouns. Together these can form events such as *throws(boy,coin)*. This symbolic formalism is convenient for shallow text-processing (an issue to be taken up in Chapter 4) and maps partially onto the formal event calculus as defined by Shanahan and used in other story-understanding models (Shanahan, 1997; Mueller, 2003).<sup>5</sup> Shanahan’s event calculus is “a many-sorted first-order predicate calculus with explicit time and can be used to express diverse representations” (Mueller, 2003).

So each story consists of a group of *entities* that are present in the story, the *paradigm P*. *P* consists of a set of entities  $n_1 \dots n_x$ . These entities are then arranged into *events*  $e_1 \dots e_y$ . An event must contain an *indicator* of the event, which is an entity (usually a verb) that distinguishes the event from other events. Null events that do not explicitly give an indicator the are not allowed.<sup>6</sup> The indicators are usually verbs, while the rest of the entities of an event tend to be, but are not exclusively, nouns. From here on, an event *e* designates a concrete indicator (such as “throw”), while *n* indicates entities that are part of an event, such as “boy” and “coin” in *throws(boy,coin)*. Time is made explicit through the variable *t*. Normally, the Shanahan event calculus has a series of predicates to deal with Vendler’s relations of achievements, accomplishments, and other types of temporal relations (Shanahan, 1997; Steedman, 2000).

---

<sup>5</sup>This binary division into events and entities is far from perfect; for example, how does one deal with states like “the boy is hungry”. There would two ways one could think of: *boy(hungry)* or *is(hungry,boy)*. In our scheme, are states like “hungry” an entity or an event? The automatic extraction technique we use in Chapter 4 would result in *is(hungry,boy)*. The second objection to this scheme is that many important facets, like discourse markers that may mark transitions, are lost in this scheme. This is currently the price we must pay for abstraction into the event calculus, and with every case of abstraction information must be lost. However, at later stages in research the analysis or research discourse markers, tense, aspect, and other things not captured by event entity predicates should not be ignored.

<sup>6</sup>For example, a child could write “Randy here” which could be translated into a “null” indicator, such as could translate into *null(Randy)*. While this is an interesting extension, we will not deal with it here due to the details of the computational implementation in Chapter 4 that do not detect it.

These are defined for event  $e$ , fluent  $f$  (an event that continues in time) and time  $t$ . Self-explanatory predicates such as  $Happens(e,t)$ ,  $HoldsAt(f,t)$ ,  $Initiates(e,f,t)$ , and  $Terminates(e,f,t)$  can be used to specify the temporal relation (Mueller, 2003). However, given evidence from children's stories that the vast majority of the narrative for 10-12 year olds takes place in either simple past or simple present, we collapse the distinction between fluent  $f$  and event  $e$ , as to have all events categorized as  $e$  (Hickmann, 2003). We will also use only the default predicate  $Happens(e,t)$ , which will be denoted implicitly from now on. So, any story's *temporal order* is a partial ordering of entities and events. It will be denoted using  $e$  to represent events,  $n$  for entities, and  $t$  for time, whereas values assigned to  $a$ ,  $b$ , and  $c$  represent series of entities in distinct events  $e_1$ ,  $e_2$ , and  $e_3$  and entities  $n$  denoted as  $n_l^k$ , with  $l$  representing the event the entity is associated with and  $k$  the particular entity in the event, so  $n_3^1$  is event 3 and entity 1. So, a story can be considered an *event structure* of the following form:

$$e_1(t_1, (n_1^1, n_1^2, \dots, n_1^a)), e_2(t_2, (n_2^2, n_2^4, \dots, n_2^b)), \dots, e_3(t^3, (n_3^2, n_3^4, \dots, n_3^c))$$

Where time  $t_1 \leq t_2 \leq \dots, t_3$ , and where  $n_t^a \neq n_t^b$  for any  $t$ . In other words, there must be a partial order of time and distinct entities in every event.

A sentence may map onto one, multiple, or no events. Two stories are said to match if they are composed of the same ordering of events and entities. For the purposes of our implementation, a total ordering will be produced from the sentences due to the limitations of our event extraction from raw text (see Chapter 4). However, in theory a partial ordering would suffice.

For the time being, to avoid the quantification problem, entities and events will be limited only to those mentioned explicitly in the text. This approach also regulates us to the second lowest level of Graesser's levels of discourse, the text-base level, with only the temporal ordering being the explicit situation model (2003).

For the time being, minimalism and clarity will be principles of our formal model to explore children's narratives for both practical and theoretical reasons mentioned above. As we develop our algorithms for plot comparison we will return to the works mentioned here for insight and lessons.

## Chapter 3

# Methodology: Symbols, Statistics, and the Frame Problem

Establishing a conceptual framework may be closer to the heart of intelligence than working within one.

*Brian Cantwell Smith, "God, Approximately"*

### 3.1 The Frame Problem and Relatives

Currently, the proposal for a symbolic model is a stripped-down event calculus that can be used to represent both the exemplar story and rewritten stories. Although the methodology that led to this model has been purposefully minimalist, the event-calculus still suffers from the Frame Problem: how can a formal system represent a dynamic and changing world (McCarthy and Hayes, 1969)?

Shanahan claims to have defeated the Frame Problem on some level by adding in an explicit representation of time and predicates for defining time in a Vendlerian temporal ontology (1997). While on the surface simply encapsulating time as a variable in the symbolic system may seem to defeat the Frame Problem, this solution barely scratches the surface of the problem, much less solves it.

The Frame Problem is much more complex than just how one can formally represent time. The Frame Problem cuts to the heart of symbolic representation and modeling itself. Dennett provides this story to illustrate the Frame Problem in full (1984):

A robot must retrieve its battery from a closet, but the battery is on a trolley with a bomb. The robot knows (through its internal frame for the situation) that its action *pullout(trolley, closet)* will get its battery out. The robot pulls the trolley out, and the bomb explodes and the robot has destroyed itself. The robot didn't have a bomb in its frame, nor did it suspect that pulling the trolley out would put it in physical danger from the bomb.

The problem is not just quantifying time, but not knowing the unintended consequences of leaving *something* out of your model, such as the leaving the existence of the bomb out of the robot's model. If we develop a model of the exemplar story in event calculus, and create a model of the pupil's rewritten story in event calculus, and then try to compare them event-by-event, our algorithm will probably fail to match many perfectly good rewritten stories unless the pupil exactly copies the story word for word. The pupil may add in a detail like "When the stork flew away, Nils left Sweden", which is completely true but not explicit in the exemplar story. The pupil may leave out unimportant details due to their irrelevance to the flow of the story. In the introduction of "Nils's Adventure", Nils is found riding geese. Yet Nils riding on geese has little to do with the rest of the story. The model must clearly be *flexible*.

The Frame Problem's relatives point some additional barriers to this flexibility. First, the *Quantification Problem* that states "actions may contain indefinitely many preconditions that our model does not anticipate"(Steedman, 2000). How many and what things do we formalize into the model? Obviously, it would be impossible for us to formalize everything. As Dennett points out, if the robot took as a prerequisite to any action a perfect formal model of the world, the robot would never stop quantifying things and so would simply never act(1984). In the model we proposed earlier, the event calculus, there are two choices: Either let a human formalize and quantify all possible events she deems relevant for the task, or let the computer automatically quantify events in the model. Both options clearly are ridden with problems. "Nils's Adventure" introduces Nils as a Swedish boy that flies on geese. However, much more could have been said in the introduction to the story that may help make the events of the story make sense. Was it even really relevant for the geese to be mentioned, since for the rest of the story Nils interacts with a stork? These are hard judgments for a human to make, and no obvious algorithms to help computers make these choices

seems feasible.

This leads to the second relative of the Frame Problem: the *Ramification Problem* that states “actions may interact with other actions in complicated and unforeseen ways with consequences that our default model does not predict”(Steedman, 2000). From the perspective of the reader, the manifestation of this problem in the story is what makes a story interesting, although it may be hard for a “common-sense” inference engine to discover. Do we demand a “fantastical-sense” inference engine, and what could that possibly entail? Must we demand not only the ability to predict the common-place, but the uncommon from artificial intelligence? In “Nils’s Adventure”, should not the plot comparison algorithm notice if the coin is left out by the pupil, and the city still disappears? Is that more or less correct than the city not disappearing? A coin that causes a city to disappear will just not exist in common-sense databases of causal relations.

This leads to the most vicious relative of the Frame Problem, which I term the *Significance Problem*: How can a model formalize only the important things? This problem is a close relative of the Quantification Problem. The problem is not how to formalize everything, but how to formalize only things that make a difference in the operation of the model. As Dennett points out, if his robot had to sort its knowledge by relevance, assuming such distinctions could be made, it would never stop processing(1984). More realistically, this means standards of significance must be made explicit, yet this is often a fairly subjective judgment that is open to interpretation. While teachers definitely have training and rationale in grading plots that would form an important basis of study, one would suspect that a lot is unconscious skill and fine interpretation rather than a list of rules that are easily formalized. Practically, StoryStation could have multiple judges read possible interpretations of the story, rank the events by the importance, and try to find some measure of reliability for this. Realistically, we would find it doubtful that such a method would actually be reliable and it would require considerable human effort. A much more natural task for teachers it to give a story a single holistic ranking for plot.

It appears that in light of the Frame Problem, the plot comparison problem for the story rewriting task is *AI-complete*. This means that it needs to have a fully artificial

intelligent tutor, capable of subtle interpretation, fine judgment, and complete world-knowledge.

### 3.2 The Limits of Models

To improve our simple event calculus model to deal with the Frame Problem and its relatives, an examination of the entire process of plot comparing must be done. Although the problem seems to be one of just comparing two models of stories in an event calculus, it is also a problem that concerns models and their limits. In the words of B.C. Smith(1985):

To build a model is to conceive of the world in a certain delimited way...every model deals with its level at some particular level of abstraction, paying attention to certain details, throwing away others, grouping together similar aspects into common categories and so forth.

In the style of Smith's metaphysics as developed in "The Origin of Objects", we shall analyze the differing levels of the semantics of story representation(1996). In essence, each of these differing levels of representation share the fact that they are about the story. Yet, each level abstracts and interprets the story in a different model, either formal or informal. First, there is the story itself, which is read or spoken to the pupil and so forms some type of representation in the pupil's mind. Let us call this the *pupil-exemplar story* relation. StoryStation cannot pry open the pupil's mind and view the representation directly; our plot analysis judges the ability of the pupil to commit their mental representation to words in the *pupil-rewritten story* relation. It is this ability to write a coherent narrative that teachers want StoryStation's assistance with. However, to do this a model of the rewritten story has to be made, the *rewritten story-rewritten story model* relationship. It is at this level of modeling that the event calculus of the story exists. StoryStation must then make another model from the exemplar story with which to compare the rewritten story, and so there is the *exemplar story-exemplar story model* relation. A human could also feasibly create an *exemplar story model* by hand. Finally, the *exemplar story model* is compared to the *rewritten story model*. Failure for the child to succeed in rewriting the plot of the story can happen at any number of levels. The child may not have even read the story (*the pupil-exemplar* relation),

or the child may have a perfect knowledge of the story but be unable to communicate it (*the pupil rewritten story relation*). However, these are outside of the scope of this project. What we can control is the creation of the model (*rewritten story-rewritten story model relation*). Later, we need to develop our plot-comparison algorithm, and we assume that unless it is hand-crafted, the creation of the model of the exemplar story (*exemplar story-exemplar story model relation*) will use the exact same event calculus extraction technique. However, our formal model of the events for both the *exemplar story-exemplar model* and the *rewritten story-rewritten story model* relations suffers from a serious case of the Frame Problem. Comparing plots in the story rewriting task is not as concise as it appears, for there are nested levels of abstraction where things can go wrong.

### 3.3 A Statistical Approach to Story Comparison

Since models operate primarily through abstraction, and if the abstraction is suffering from the Frame Problem, there must be a way we *reverse the process of abstraction*. B.C. Smith calls this *reconciliation*, “filling out the details” left out by abstraction (1996). Using the levels of abstractions and relationships between them just detailed, it is clear that any model of a story, to overcome the Frame Problem, must reconcile with the original data, informal as it is, of the story. Yet there is a paradox: a model by nature is limited to representation with formal symbols.

Then clearly the question is one of methodology: How directly are these representations made from the data? Instead of using a formal representation that abstracts from the underlying data, such as our proposed “event calculus”, a model that keeps closer to the data can be made from the probabilities of the words in the original data itself. Word frequency counts are one example of this. While admittedly the probabilities underlying the data are really *not* the essence of the story itself, these probabilities are “closer” to the data than the event calculus. This type of modeling definitely can help us overcome, though not defeat, the Frame Problem.<sup>1</sup> While probabilities are on

---

<sup>1</sup>Even these models make mathematical assumptions, usually about what type of statistical distributions lie beneath the data and the relations between the features of the data. See Chapter 6 for more information.

some level definitely both formal and symbolic, they are created directly from data collected in the real world, unlike an event calculus that is either hand-crafted or automatically created through many levels of abstraction. The caveat of this is that the reliability of a statistical model is dependent on the amount of the data that is used to form it, which for the corpus of stories is quite small. Also, the corpus may or may not be representative of the task. Still, statistical modeling skirts by the Frame Problem since the model changes directly with any changes in the data. It also skirts by the Quantification Problem by quantifying only what can be gained from the data, along with whatever mathematical assumptions the statistical model makes. It attempts to escape from the Ramification Problem by making it clear the mathematical assumptions about what parts of the data change other parts of the data. How it escapes the Significance Problem is more subtle: if the training data of a statistical model includes human judges of significance, then these can be extracted statistically from the data itself through machine-learning.

Korb (1998) delivers what he considers his “solution” to Dennett’s(1984) classic version of the Frame Problem through statistical modeling: A Bayesian network learns the causal structure of the trolley, the bomb, and so makes a probabilistic model. A search strategy is applied to this model, and if the bomb explodes, try again but remember what happened. If the bomb does not explode, a solution has been reached.

This is a classic example of machine-learning, with the past experience being the training data and the future experience being the test data. With enough correct training data, a better model should be deduced. The question becomes one of resources and quality: How much data do we have and how well does it represent the task we want it to do?

### 3.4 Limits of Statistical Modeling

However, assumptions can be brought in via the mathematical model. For part of the experiment we will use a mathematical model with minimal assumptions, a model that uses the variance of word distributions in combination with nearest-neighbors for plot classification. *Latent Semantic Analysis* (LSA) provides an approximation of “seman-

tic” similarity based on the hypothesis that the semantics of a word can be deduced from its context in an entire document (Foltz, 1996). Details of how Latent Semantic Analysis and Nearest-Neighbors works can be found in Chapter 6 and the experimental results can be found in Chapter 7. These results show the method can distinguish non-poor stories from poor stories with LSA and Nearest Neighbors, but a fine-grained analysis of plot is not feasible using these statistical techniques.

The rewritten stories and their rankings for plot coherence for the “Nils’s Adventure” corpus can be thought of as a distribution of plot coherence rankings (For details on the ranking and reliability of plot coherency, the exact numbers of the ranks in the corpus, and the results of using only LSA to classify stories; see Chapter 7), with a large group of poor rewritten stories, a large group of good or fair rewritten stories (featuring minor omissions and errors), and a small group of excellent rewritten stories. It is not surprising a statistical model based primarily on word distributions could correctly discriminate the non-poor from the poor rewritten stories, since these two groups are on opposite ends of the word distributions. The excellent and good rewritten stories closely resemble the exemplar story, sometimes almost word for word, and so share the same word-distribution with the exemplar story. The poor rewritten stories usually have little resemblance to the exemplar story, and so have a drastically different word distribution. Both these types of stories can easily be detected by using statistical methods like LSA that quantify the similarity between word-distributions. However, minor omissions and errors that characterize the problems of the rewritten stories can not be so easily quantified by looking at the word distributions.

This leads to the primary weakness of statistical models: they do not take into account any higher-order regularities in the data that are not directly derivable from the data. Mediocre rewritten stories can not be identified precisely because they have minor omissions and errors, and these errors and omissions are usually identified by knowledge of higher-order regularities of the plot that come from the exemplar story. For example, one regularity in the plot of “Nils’s Adventure” is that after Nils jumps on the stork’s back, he becomes an elf. A pupil may get this confused and write that Nils transforms into an elf, then jumps on the stork’s back, essentially forgetting the causal relation. Yet, unless the order of events is taken into account, the word distributions

for this error remain the same, and so it undetectable by LSA and other models that ignore higher-order regularities like events. Furthermore, the presence or absence of these higher-order regularities are precisely what teachers are most interested in. While one can conjecture about the means of distributions, the teacher and the pupil need to know *exactly* what was left out of their plot. As Hickmann suggests, the very flow of narration in children's discourse is based on interplays between regularities such as grammar and semantic roles (2003). These and temporal order are not captured adequately using only word frequency distributions over the entire document, regardless of how complex the mathematical processing (such as sub-space reduction in LSA) upon these documents is. Defenders of LSA would be quick to point out that these grammatical and semantic roles are captured by the word frequencies. This is clearly not true. Reversing the order of the words in "Nils's Adventure" would produce a completely garbled story, but then comparing it with the exemplar "Nils's Adventure" story produces a perfect similarity score of 1.0 using LSA! On the level of individual sentences as opposed to the whole document, imagine reversing subject and object roles for a sentence. While this is clearly in error, "Nils's jumped on the stork's back" produces the exact same LSA similarity score as "The stork jumped on Nils's back". Problems like these are endemic to statistical models. It would be more accurate to say that the similarity of a rewritten story to an exemplar story partially depends on word distributions and partially depends on higher-order regularities, a statement that could be generalizeable to many modeling tasks (Smith, 1996). The question then becomes one of integrating higher-order regularities and partial dependencies into statistical models.

### 3.5 Hybrid Models

The model must somehow capture these regularities. As shown earlier, at least some of these regularities can be captured by using formal symbolic models that make these regularities explicit. In a *hybrid model*, a formal symbolic model enters a mutually beneficial relationship with a statistical model of the data. As the Frame Problem shows, there are many things formal systems have difficulty with, such as formalizing all the significant aspects of the data. However, symbolic modeling can capture higher-

order regularities in the data that would otherwise be very difficult to find using purely statistical methods.<sup>2</sup> When symbolic modeling is used in tandem with statistical modeling, one gets the best of both worlds. The symbolic model can formalize as much of the regularities of the data as can be easily engineered or extracted, and the statistical model can make up for what it misses by reconciling the model with the data. More importantly, by using supervised training, human knowledge that would otherwise be hard to encode explicitly in a symbolic model can be encoded into the statistical model.

To return to the plot comparison problem, we have already developed a simplified event calculus as the symbolic component of our hybrid model. The use of LSA to compare the word distributions of the rewritten story with the exemplar story serves as a statistical analysis of the text. This is against the grain of research in story-understanding systems, which focus on purely formal symbol manipulation (Mueller, 2003). Only relatively recently in story analysis systems have researchers considered extracting formal knowledge representations from raw text (Riloff, 1999). These systems are problematic because they extract knowledge without being able to judge how significant it is, as well as relying on human correction to sort out irrelevant frames. We need a system that works on an explicit event calculus with absolutely minimal human supervision once the training regimen is complete, and one that discriminates on a fine-grained level not only the grade of a the story, but what factors led to their grade, so that eventually the system can be expanded to provide online feedback to students.

The closest hybrid model to ours is the online discourse analysis software embedded in the essay-grading software “Criterion” by ETS technologies, which is designed to annotate parts of an essay according to pre-defined macrostructural categories such as “Thesis”, “Main Points”, “Support” and “Conclusion” (Burstein et al., 2003). Although the domains of essays and children’s stories are vastly different, a number of innovations of this hybrid model can be analyzed and contrasted with the goals of our

---

<sup>2</sup>In fact, using a symbolic model to place higher-order regularities upon the data can be viewed as an experiment in of itself, as if the model works well one can feel safer in saying that these higher-order regularities actually are in the data. For example, it is not completely self-evident that the event calculus is a good formalization of the plot of stories, and in particular the plot of stories in the corpus used for the story rewriting task. However, if the event calculus works, that is evidence for the event calculus being a good symbolic model.

system. Its training set is manually annotated with the macrostructural categories by human judges and checked for reliability. Due to the interplay between coherence and cohesion, and considering the problems of macrostructural approaches such as story grammars, our system will instead seek to automatically extract events and entities. Burstein et al. (2003) uses Rhetorical Structure Theory to parse the text into discourse relations based on satellites and nuclei connected by rhetorical relations. Moore and Pollack (1992) note that Rhetorical Structure Theory confuses the informational (the information being conveyed) and intentional (attempting to get the speaker to believe the information conveyed) levels of discourse, with often priority being given to the intentional level. Since in a children's narrative we are attempting to convey information about the narrative, unlike the primarily persuasive task of an essay, our system focuses more on the informational level as embodied by the simplified event calculus. Burstein et al. (2003)'s system automatically classifies the text by using as features for the training set the discourse relations, word positions, key word, punctuations, and a finite-state transducer model of the typical essay. They also use the manually-annotated text, letting the machine-learning algorithm sort out which of these many features can be used as significant predictors of the location of the macrostructural annotation. While we use fewer features than Burstein et al. (2003), we follow in their footsteps by letting machine-learning discover the relative significance of features. However, our entire process is automated.

It is this synthesis of the formal<sup>3</sup> and statistical<sup>4</sup> models, tied together with human-annotated data, that provides a solid example of the kind of hybrid model our task needs. First, it allows the model to use formal regularities of the data and statistical aspects of the data as possible features for the machine-learner to use. This approach attempts to overcome the Quantification Problem by giving as many both formal and statistical features to the machine-learner as one thinks is relevant. Burstein et al. (2003) definitely has a "throw the kitchen sink" approach to selecting features. One might suspect this is because the Frame Problem can definitely be rephrased as the *Feature Problem* for machine-learning: how can one select the best features? Obvi-

---

<sup>3</sup>As in the use of finite-state transducers and macrostructural annotation by Burstein et al. (2003).

<sup>4</sup>Burstein et al. (2003) also model probabilities of discourse relations in relation to probabilities of word positions and punctuation.

ously, using as many as one can think of and letting the machine-learner sort them out is one approach. Human knowledge is encoded into the model via the supervised training of human-rated training data, and the fully trained machine-learner models the general properties of the human judgment. Given a test-set, the properly trained machine-learner should be able to properly label the unrated test data with the correct grade. For more information, see Chapter 6.

This combination of statistical modeling<sup>5</sup> and symbolic modeling<sup>6</sup> manages to give a model a good chance at overcoming the Frame Problem and its relatives. This process works because the subtle interpretation, fine judgment, and world-knowledge of a human are encoded in the unrated training data, and the relations between these and the features are modeled by the machine-learning algorithm. The Frame Problem is overcome by building directly a statistical model of the data, or at least some of its features. The Quantification Problem can be overcome at some level by giving the machine-learner as many features as possible. The Ramification and Significance Problems are solved by letting the machine-learner build a statistical model of how the features relate to human judgment on the supervised training data. Essentially, human knowledge that resists explicit symbolic representation is introduced into the model.

### 3.6 The Story Comparison Model

Our model adapts this hybrid machine-learning paradigm to the story comparison-task. The regularities of plot that underlie the stories can be formalized by an event calculus, which will be automatically extracted from the text via an XML-pipeline. The event calculus of the exemplar story will also be extracted automatically, and a number of variations of plot comparison algorithms will be developed that attempt to match the event structure or text of each rewritten story with the event structure of the exemplar story, and the presence or absence of events and other properties from the rewritten story will be output as a feature vector. The statistical word-distribution of

---

<sup>5</sup>As built either as features for the machine-learner or the statistical model the machine-learner builds itself.

<sup>6</sup>As embodied by whatever features given to the machine-learner that depend on higher-order regularities in the data.

the text will be also extracted from the text via a comparison with the exemplar story using LSA. Both the results of the plot comparison algorithm and the LSA similarity scores will be used as features for a machine-learner. A training corpus of stories marked using a reliable plot rating scale developed by a teacher will be used to train a supervised machine-learner. Once trained, the machine-learner should be able to grade ungraded texts after the features of the ungraded texts have been automatically extracted. Through a creative use of statistics and symbolic frameworks, embodied in using both LSA and the event calculus, this model skirts direct confrontation with the Frame Problem.

# Chapter 4

## Pipeline Annotation

That ambitious Comp. Sci. grad student, eager to get his Ph.D. and begin making real money, will create The Two Final Infosets: MatterML and SpiritML. Then, late one night, as rain falls in torrents and lightning flashes outside his laboratory windows, he'll run XSLT to transform the material world to the spiritual world. We'll be gone. The last material object on earth will be that graduate student's open copy of "XML in a Nutshell".

*Frank Willison*

### 4.1 XML Pipeline in a Nutshell

Extracting an event calculus from raw text involves extracting the "semantics" of each story, which can be done by layering one modular level of processing over another. A XML pipeline renders this process simple: at each level of processing, a XML-enabled natural language processing component can add mark-up to the text, and use any mark-up that the previous components left. The text can be thought of as being sent through a pipeline of XML-enabled tools whose final result is a richly annotated version of the original story. During this process, enough annotation will be provided to allow extraction of the event structure of the text. All layers in the pipeline are fully automatic. For this particular XML pipeline we used the LT TTT (Language Technology Text Tokenization Toolkit) as the underlying toolkit, although non-LT TTT

software is integrated into the pipeline when needed (Grover et al., 2000). If off-the-shelf components can not be found, software is created. This had to be done in the case of pronominal resolution and the transformation of text chunks into an event calculus.

The pipeline processes the text in this order:

1. Pretokenization
2. Words and other Tokens
3. Sentence Boundaries
4. Penn Treebank tags
5. GlenCova Pronominal Resolution
6. Lemmatization and Chunking
7. Extraction of Event Calculus

The results of the major steps in the XML pipeline for a sample story are presented in Appendix D.

## 4.2 Words, Tags, and Sentences

LT TTT is based on LT XML, a high-speed XML processing library built to deal with marked-up XML (Thompson et al., 1996). LT TTT provides a general purpose transducer called *fsgmatch* that takes as input an arbitrary stream of characters, which can then be processed at either the character level or the XML element level through a series of rules that are encoded in a separate XML file. Much of our pipeline and many of the rules were taken from the examples provided by LT TTT and similar work done using LT TTT to explore automatic deep semantic processing of medical abstracts (Grover and Lascarides, 2001). This same system led to the success of the University of Edinburgh in Named-Entity Recognition at MUC-7. For details as regards LT TTT's high reliability and accuracy at tokenization see Mikheev et al. (1998).

The entire pipeline is run as a UNIX-pipeline in the style normally used by LT

TTT, although various perl scripts are used to help the process of communication when needed. For full details of the process, see the website [www.semanticstories.org](http://www.semanticstories.org). The rest of this chapter describes the operations of the XML-based pipeline in sequential detail.

The program *fsgmatch* does considerable pre-processing of the text to ensure words are tokenized properly. First, it converts all text from the character level to the XML level through cascades of regular expression rules. After dividing the text into paragraphs, it marks-up punctuation. Unnecessary white-space is deleted from the text.

Now that the story has been transformed from a stream of text characters to a stream of XML elements, it can be tokenized through another cascade of *fsgmatch* rules. Using the mark-up of the previous level, these rules extract all complete elements from the text. Since there are over thirty rules, only a few of the many traditional tokenization problems these rules correct will be mentioned: large numbers with commas (“10,000”), contractions (“can’t”), titles (“Mr.”), abbreviations (“I.B.M.”) decimal numbers (“6.7”), and hyphenated words (“self-organizing”). While these types of tokens will be very rare in the story rewriting task, marking up tokens correctly is good computational linguistic hygiene since one can never predict exactly what will be in the rewritten story nor how errors at the level of tokenization might propagate in higher levels of the pipeline. Once this tokenization is complete, words are marked-up. A maximum entropy sentence boundary detector *ltpos* is then used to disambiguate sentence boundaries. This distinguishes between a full-stop being used as an end-of-sentence marked (“Nils walked down the beach.”), part of an abbreviation (“Mr. Stork told Nils a story”), or both (“Nils went home to his home in San Fran, Cal. yesterday.”).

A part-of-speech tagger, based on Hidden Markov Models that uses Maximum Entropy for parameter estimation, is called from *ltpos* is to tag the words using the Penn Treebank tag-set (Marcus et al., 1994). The tagger was trained on the Brown Corpus and requires no full parse of the sentence in order to tag a word, which is to be avoided since a large portion of the pupil’s writings are ungrammatical, given their varied levels of grammatical development and writing fluency. We constrain ourselves to using this tagger, for while a full parse could doubtless be generated by a statistical parser

such as the Charniak parser, such parses would be incorrect for ungrammatical sentences (Charniak, 1999). Since one of the guidelines of this project is to not propagate up errors in our model through non-minimal assumptions, this is unacceptable.<sup>1</sup> From a purely intuitive view, errors based on an assumption of grammaticality could lead to further errors at higher levels of processing in the pipeline. If an incorrect full parse is given for a sentence and during the later pronominal resolution phase our pipeline uses the Hobbs Naive algorithm, which depends on a correct full parse, the pronoun resolution would likely be incorrect (1977).

### 4.3 GlenCova: Rule-based Shallow Pronominal Resolution

One of the main goals of children as they learn to write narratives is to learn the proper use of anaphora reference. When children first begin writing stories they tend to explicitly mention every entity, but as they age<sup>2</sup> they begin using anaphora references more (Hickmann, 2003). The use of pronominal anaphors signals a higher stage of narrative development, and often children first use pronouns to denote the main characters of their narratives. Yet, it makes the process of automatically parsing their text into an event calculus harder, since often crucial referents will be signified by an often ambiguous pronoun. If the child writes “He threw the coin away” which reduces in our calculus to *throw(he, coin)* and some later plot-checking algorithm is attempting to match *throw(he, coin)* to *throw(Nils, coin)*, one would want *he* to match to *Nils*. However, we cannot assume for all stories that pronouns resolve to the correct referents, for the *he* mentioned in our example could refer to a town guard mentioned earlier by the pupil. It is by this phenomena that leads the XML pipeline into the world of anaphora resolution.

Very few domains in computational linguistics have had as many researchers devote as much time and energy to them as anaphora resolution. An anaphora reference

---

<sup>1</sup>See the footnotes of Chapter 6 for a discussion of Minimum Description Length, which is a more formal argument for this approach.

<sup>2</sup>Including the target age, 10-12, of the users of StoryStation.

is a reference to an entity, called the *antecedent* of the anaphor, in the discourse. Pronoun resolution is the process of finding the correct antecedent given a pronoun. There has been a large amount of work in this field. Unfortunately, currently there are no general purpose shallow anaphoric resolution programs available. In response to this lack, we created the GlenCova rule-based pronoun resolver that operates on the “shallow” surface text. It requires the text to be divided into sentences and each word have a part-of-speech tag. We were careful to limit what types of anaphors we wanted to resolve. The most frequent incident of anaphoric resolution that needs to be resolved is when either the subject or object noun of a sentence is referred to by a pronoun. This is usually an inter-sentence anaphora as opposed to an intra-sentence anaphora. Since children’s stories are fairly straightforward as regards pronoun use (using pronouns to follow a small set of main characters in a story), a clear algorithm should account for much of the pronoun resolution needed. Our algorithm should resolve only anaphoric pronouns, not non-pronoun resolution, such as “Nils found the coin. He didn’t see any use in *this*.” We will also not cover *cataphora*, anaphora that are introduced before their referent, such as “Nils found *it*. The coin was green.”

There are several well-known algorithms for pronoun resolution. The classical approach is to use Hobb’s Naive algorithm, which requires a full parse of the sentence (1977). The algorithm finds an antecedent for the pronoun by going up the parse tree to the first *NP* node dominating the pronoun and then transversing the branches below that node in a left-to-right, breadth-first fashion, taking as antecedents any *NP* which has an *NP* or *S* node between it and the pronoun(Hobbs, 1977). We will not use this method since we do not want to restrain the children to forming sentences that are grammatical and so have a correct full parse. A popular current approach is Mitkov’s heuristic approach, that unlike the Naive algorithm does not seek a full-parse, but instead relies upon a number of anaphoric indicators in a partial parse to pick antecedents(Mitkov, 1998). These indicators include such obvious criteria as number and gender agreement but also more subtle criteria such as “given-ness” and “term preference”. Through this approach, the antecedents are all assigned points, and the one with the highest aggregate score is decided to be the antecedent of the pronoun.

A minimalist approach may solve most of the problems without resorting to sub-

tle indicator criteria or full-parsing. We base the GlenCova pronoun resolution algorithm on a cascading rule-based approach directly inspired by the CogNIAC algorithm (Baldwin, 1997). The algorithm needs the corpus to be divided into sentences and words marked-up with their POS tags. It uses three history lists to keep track of all possible noun antecedents (candidates) of the pronoun: the discourse list, the current sentence list, and previous sentence list. Note that pronouns resolved by GlenCova will be added be antecedents, but unresolved pronouns will not be.<sup>3</sup>

While far from exhaustive in covering all cases of pronominal resolution, this algorithm will likely be sufficient for the story rewriting task. In children's stories, pronouns are most often used to follow the main character, and thus the use of the pronouns will be straight-forward compared to adult anaphor usage (Hickmann, 2003). This algorithm does not guarantee to resolve all pronouns; it is very possible that a pronoun may have all its rules applied to it and still be unresolved. However, when those rules apply, they have a high-precision, usually around 90% collectively (Baldwin, 1997). The major modification to the CogNIAC algorithm was to apply the rule once per type of pronoun, instead of once per all pronouns as CogNIAC originally did. This was found to improve performance by up to 50% on a sample set of sentences chosen randomly from the corpus. Since there are relatively few proper pronouns in the story and children use proper pronouns to follow named characters, the algorithm that deals with proper pronouns before other types of pronouns would have a naturally higher accuracy. In our modified version of CogNIAC, proper pronouns are dealt with first and separately, and only after they have been resolved are other non-proper pronouns dealt with in order.

## 4.4 Lemmatization and Time

The next step is for as many words as possible to be reduced to their lemmatized form and for chunks to be made from the sentences. For this, the finite-state lemmatizer MORPHA was integrated into the pipeline (Minnen and Pearce, 2001). Since MORPHA is known to have its limitations, the lemmatizer used by Abney in the SCOL

---

<sup>3</sup>This algorithm was implemented in Java and is available for use from [www.semanticstories.org](http://www.semanticstories.org).

For  $t$  being proper, non-proper, singular and plural pronouns

For every sentence  $i$  in discourse:

1. Add all non-pronominal nouns of type  $t$  in sentence  $i$  to the history list of the current sentence.
2. Find all pronouns in the current sentence  $i$ , if any.
3. Find all candidates in the discourse, previous sentence, and current sentence history lists that match (if applicable) the pronoun.
4. Attempt to find the antecedent using the following rules in-order, exiting as soon as a rule is satisfied: <sup>a</sup>
  - (a) *Unique in Discourse*: If there is a single possible candidate in the discourse, it is the antecedent.
  - (b) *Unique in Current and Prior Sentence*: If there is a single possible antecedent in the history list of both the current sentence and prior sentence, it is the antecedent.
  - (c) *Unique in Sentence*: If there is a single possible candidate in history list of the prior sentence, pick it as the antecedent.
  - (d) *Two Proper Pronouns in Prior Sentence*: If  $t$  is of type proper pronoun and if there are two proper pronouns in the previous sentence, resolve the pronoun to be the first of the two pronouns.
5. If none of the rules apply, the pronoun is left unresolved.
6. The history list of sentence  $i$  is added to the history list of the discourse. A copy is made so that there is a history list for the prior sentence  $i - 1$ .

---

<sup>a</sup>These are a slight modification of the rules used in CogNIAC, especially the last one. They were modified as to work with text unlabeled with predicate semantics or lexical information, such as whether a noun is the “subject” of a sentence, or whether a pronoun is “possessive”.

Toolset was also used, augmented by a list of lemmas created by Mirella Lapata and added to by myself (Abney, 1995). Lemmas are needed for later processing using WordNet, which will be shown in the next chapter to be essential for comparing plots (Fellbaum, 1998).

One level of processing left out of our current pipeline is the use of time identification rules to attempt to find out what tense, aspect, voice, and modality a sentence is in. The question of how to adequately represent time in discourse is still an open area of research (Steedman, 2000). There has been even less research into automatic identification of tense, aspect, voice, and modality. Hachey and Korycinski (2003) have developed (using *fsgmatch*) a series of rules from which the voice, tense, aspect, and modality of the sentence can be gathered, and these were tested in our experiment but were found to not have enough coverage to adequately represent time information well enough to be used by the XML pipeline. This does not present a large problem for our analysis of the text, since unlike adults, children have a less complex usage of tense to demarcate the flow of time in their narratives (Hickmann, 2003). Adults typically employ a number of differing tenses: 50% of the time adults use the simple present, but they also use the progressive about 30% of the time and the past 14% of the time. In children's speech, simple past occurs 57% of time, and simple present occurs 31% of the time, with the progressive occurring 11% of the time, and all other types of tense and aspect occurring about 1% of the time (Hickmann, 2003). While adults frequently mix their tense and aspect to elaborate on complex time sequencing in narratives, for children about two-thirds of the time the story stays in the simple perfect or simple past for the entire duration of the narrative. This means that for the purposes of any algorithm, we will assume that the order of sentences reflect the temporal order of events in the plot, with the first event happening in the first sentence and the last event in the last sentence. It would be useful to use temporal information to order these further, but at the current time the technology has not yet been developed. See Appendix C for an example of a story that has had its pronouns resolved and is fully marked-up.

## 4.5 Retrieving the Event Calculus

An event calculus is composed of a series of events ordered explicitly in time. In our simplified implementation, each event is denoted and anchored by an indicator word, usually the main verb of the sentence. With this indicator, the rest of the event is composed of a series of entities, usually with each denoted by a noun. These nouns almost always include the subject and object of the indicator. However, prepositional phrases may also denote a crucial entity and its relationship to an event, and entities marked by prepositions are also in listed as entities in the event calculus.

There are multiple ways one could automatically extract these from our text. One choice would be to build a dependency grammar using the publicly available dependency parser MiniPAR, and extract the subject, object, and prepositional entities from the dependency chains(Lin and Pantel, 2001). However, MiniPAR uses a full parsed constituent tree to assign its dependency relations and we do not wish to force any type of full grammatical parse of the sentences in this corpus, since many of the children's stories are not strictly grammatical.

However, Abney's Cass Chunker from the SCOL toolset operates on unparsed sentences, needing only words tagged with part-of-speech tags, and can deliver subject and object relations rooted at a verb, as well as prepositional entities(1995). The Cass Chunker relies on non-recursive finite-state methods such as the rules used earlier in the pipeline with *fsgmatch*. The principle behind the Cass Chunker is that a sentence can be thought of as composed of phrases which non-recursively build upon phrases at the previous level. These phrases can be built into chunks, a series of constituents that end in a head. Using regular expressions for a large-scale grammar of English, Cass can recognize these and use tools in SCOL to extract the verb, subject, and object, as well as the prepositional relations. These chunks are identified in the sentence, and from these chunks various relations can be extracted, including subject-object predicates, as well as entities involved in prepositional relations. This allows the XML pipeline to extract subject-object and other relations over any coherent chunk in a text using the SCOL *tuples* program, whether or not the sentence is grammatical. However, there is a trade-off in between this approach and one that requires a full parse such as those using dependency grammars. First, the Cass Chunker is highly precise, but often

inaccurate and misses relations and entities that are not in a chunk that can be identified by its finite-state methods. To its merit those tuples that it does identify are usually correct.

Once the tuples of chunks have been identified, they are mapped into an event calculus, adapting a few simple rules for identifying indicators, entities, and using Allen’s classic “now-point” algorithm to explicitly identify time-sequences(1987).<sup>4</sup> A *now-point* is a marker used to identify the current temporal state of the event calculus while in the process of building it. Although as pointed out earlier, it would be useful to use tense and modality to determine the now-point of a sentence, for this experiment the now-points are ordered as the tuples are delivered by the Cass Chunker. Each tuple identified by the Cass Chunker is scanned for a verb. If the verb is found, it is made the indicator. Then the chunk is scanned for subject and object relations explicitly identified by the Cass Chunker, and these are added as entities. Finally, all other relations are scanned. The objects of all other relations (such as prepositional relations) are added as entities, although the preposition itself is not added to the event calculus. Lastly, children often tend to have an over-use of connectives, especially “and”. The Cass Chunker deals with this by identifying chunks on both sides of the connective, and creating tuples from both. For example, this sentence is in the exemplar story: “Nils’s geese fly him everywhere over Sweden and when he goes on the back of a bird he becomes an elf”. It is split into three chunks by the Cass Chunker, such as “Nils’s geese fly him everywhere over Sweden”, “when Nils goes on the back”, and “Nils becomes an elf”, in which the references to “he” are resolved by the GlenCova anaphora resolution program. Note that unfortunately “bird” is not detected by the Cass Chunker. The chunks are reduced to the following by the Cass Chunker:

```
34 fly :obj Nil :rb everywhere :cdqlx Sweden :subj geese
43     go      :on back          :subj Nil
52     become :obj elf           :subj Nil
```

The “34” is the number of the chunk, which is taken to be the time. “Fly” is identified as the indicator verb, and the entities “Nils”, “everywhere”, “Sweden” and “geese” are

---

<sup>4</sup>The files for transforming the results of the Cass Chunker in tuple form to an XML-formatted event calculus are found on the web-site [www.semanticstories.org](http://www.semanticstories.org).

extracted, so that the final event calculus can be considered: *fly(t=34, Nil, everywhere, Sweden, geese)*. A similar algorithm operates on the other two chunks. They are then stored in the following XML format:

```
<Event indicator="fly" time="5">
  <Entity name="nil" />
  <Entity name="sweden" />
  <Entity name="geese" />
</Event>
<Event indicator="go" time="6">
  <Entity name="back" />
  <Entity name="nil" />
</Event>
<Event indicator="become" time="7">
  <Entity name="elf" />
  <Entity name="nil" />
</Event>
```

See Appendix C for a detailed example of how a story can be transformed from the marked-up XML to an event calculus. Ultimately, this is a primitive yet working solution to extracting an usable symbolic knowledge representation, the event calculus, from raw text. This prevents our entire system from being a “toy” system, instead allowing it to interact with the actual written stories of the pupils. It also prevents our symbolic system from becoming excessively baroque and complex, since our information extraction techniques define the limits of our symbolic model.

# Chapter 5

## The Plot-Comparison Algorithm

”One thing I don’t quite understand, Knipe. Where do the plots come from? The machine can’t possibly invent plots.”

“We feed those in, sir. That’s no problem at all. Everyone has plots...”

*Ronald Dahl, “The Great Automatic Grammatizator”*<sup>1</sup>

### 5.1 The Problem of Similarity

One of the most daunting problems facing the story rewriting task is that of creating an algorithm to identify if the student has rewritten the correct events: a Plot-Comparison Algorithm. Since the story rewriting task involves imperfect recall, story events and entities will likely be changed or left out. The story rewriting task involves the student choosing their own diction and expressing their own unique mastery of language, so variation in how the fundamental elements of the story are rewritten is to be expected. To deal with these issues, an algorithm has to be devised that takes the event structure of the rewritten story and compares it to the event structure of the exemplar story while disregarding the particularities of diction and grammar. It should be noted there is a fundamental asymmetry in the story rewriting task: There is only one exemplar story that serves as the gold standard by which all the rewritten stories are judged.<sup>2</sup>

---

<sup>1</sup>As quoted by Bailey (1999).

<sup>2</sup>This asymmetry could reflect itself in the algorithm’s input data, since it is also feasible to manually create from the exemplar story some event structure to judge the rewritten stories by. The automatic extraction of an event structure from raw text by an XML-pipeline is detailed in Chapter 4.

It is unlikely that looking at the similarity of the whole discourse (or even somehow comparing whole event structures) can solve this problem in a way that provides useful feedback to the pupil or analysis of the rewritten story to the teacher. The pupils do not need for the algorithm to tell them whether they have recalled the plot of the exemplar story correctly, they need to know what events they have forgotten or otherwise gotten wrong. Our algorithms will focus on comparing the event structure on an event-by-event basis, attempting to compare specific events in the exemplar story to specific events in the rewritten story.

The problem can then be rephrased as one of credit allocation for the similarity of rewritten events to the exemplar event. First, the words used in the events of the two story models may differ. The exemplar story model might use the event *see(Nils, stork)*, but a rewritten story may use the word “boy” where in the exemplar story the character’s proper name “Nils” was used. However, since “Nils” is a “boy” in the story, then at least partial credit should be assigned to the rewritten story for identifying the character correctly. This is an story specific similarity that can only be identified by identifying relevant co-reference chains in the story. Another more general type of similarity involves using words commonly associated in English, such as “bird” for “stork”. In the rewritten story the event *see(Nils, bird)* might be used, and this type of general similarity cannot be usually resolved by a co-reference chain unless the similarity was mentioned as part of a co-reference chain earlier in the story. So, more general similarity resolution algorithms should be employed. The online lexical database WordNet, which contains the synonyms of the sense of a word in a “synset” could be used to identify the relation, since “stork” and “bird” are in the same synset. However, as WordNet is incomplete and suffers from the biases of its creators, this solution is imperfect.

Another question to vex any comparison of similarity is one of temporal ordering. Assuming the event structures have been created from raw text and ordered correctly, there is a chance that the rewritten story model may have an event out of temporal order. For example, a pupil could rewrite the “Nils’s Adventure” story having the city disappearing at the very end, after Nils and the stork fly away. Clearly, errors in temporal ordering of events would have an effect on the final grade of the story and

should be recognized by any plot comparison algorithm. Also, there may very well be events in the story in which the temporal ordering is not important.

How much partial credit to assign a particular synonym, co-reference use, or temporal mistake is a difficult question, one that is closely related to what we in Chapter 3 identified as the Significance Problem and plan to solve later using techniques from supervised machine-learning. The problem of plot similarity will be approached in a number of ways. First, a symbolic algorithm, called the “Plot Comparison Algorithm”, will be presented that operates over the event calculus representation of both the exemplar story and the rewritten story. Afterwards, two decompositions of the Plot-Comparison Algorithm will be shown, one that operates without any temporal ordering constraints or indicator constraints (“Unconstrained Plot Comparison Algorithm”) and another that operates by comparing an event calculus representation of the exemplar story to the raw text of the pupil’s story (“Plot to Text Comparison Algorithm”). As the formal constraints are removed one by one, the model becomes simpler. In fact, the most simple model is, as mentioned in Chapter 3, just comparing the exemplar story to the pupil’s story using LSA. The operation of LSA is explained in the next chapter. The results of all of these algorithms for detecting plots are compared in Chapter 7. All of these algorithms were implemented in Java, along with various utilities for manipulating event calculi, and are available for download from [www.semanticstories.org](http://www.semanticstories.org). Note that since we are running three plot comparison algorithms, we will use capitalized letters to distinguish when we are talking about one specific formal plot comparison algorithm rather than plot comparison algorithms in general. When we mentioned the “Plot Comparison Algorithm”, we are talking about the formal algorithm presented in the chapter below, but when we are talking about “plot comparison algorithms” we are collectively referring to not only the Plot Comparison Algorithm, but its variants the Unconstrained Plot Comparison Algorithm and the Plot to Text Algorithm.

## 5.2 The Plot Comparison Algorithm

The Plot Comparison Algorithm approaches these questions in a discrete manner, taking as input the event structures from both a rewritten story and an exemplar story

and comparing their events for use of co-reference, synonyms, and temporal ordering one at a time. Informally, the ideal way to check a plot for conformity is to cycle through the events in the exemplar story’s event structure, checking them one at a time against the rewritten story’s event structure. To keep track of temporal order a now-point is created for the rewritten story model. Each event from the exemplar story is iterated through, and a match is attempted using the indicator of the event.<sup>3</sup> In the event *see(Nils, bird)*, *see* is the event’s indicator while *Nils* and *bird* are entities. If there is not an exact match of indicators for events, the algorithm attempts to find a match using a co-reference list gathered from the exemplar story event structure and a synset automatically derived from WordNet. If either of these succeeds, this is noted. Then the algorithm iterates through the entities of the exemplar event and attempts to match them to the entities of the event structure in the same fashion as it matched event indicators, first checking for an exact match between words and then checking for a match using the co-reference list and WordNet. Once an indicator match is made, any matches between indicators and entities are output to the binary feature vector, and then a “temporal ordering” match is made. If the match is correct and on the same point in the rewritten story model, the match is correct. The “now-point” progresses to the next event, and this match between an indicator or an event is recorded as a 11. A “similar” match of the synset or coreference chain is marked as a 10. When there is no match, the algorithm records a 00. If the events were found in the right temporal order, then a 1 is output at the end of the event-matching, otherwise a 0 is output.

Formally, given the event structure of the exemplar story model  $e_a$  is composed of events that are ordered  $e_a^1, e_a^2 \dots e_a^n$ , and the event structure of the rewritten story model  $e_b$  is composed of events are ordered  $e_b^1, e_b^2 \dots e_b^m$ , where  $m$  may not be equal to  $n$ . The two stories may be compared and reduced to a binary vector  $b$  via the following algorithm.<sup>4</sup> A binary vector encoding is used for the results since it is the format required for the machine-learning algorithms (detailed in Chapter 6) that are used to correlate event structures and other features with human grades.

---

<sup>3</sup>The indicator being the “verb” that denotes the relation between the entities in the event.

<sup>4</sup>We are using vector notation from linear algebra to denote our event structures and other mathematical objects. The subscript is used to identify the row components of the vector, and the superscript the column vector. So, this means  $e_1^2$  means the second row of the first column, not the value of  $e_1$  squared. Also, given that  $e$  denotes an event structure,  $e_1^2$  would represent the second entity of the first event.

For  $i = 1$  to  $d$

1. For  $j = 1$  to  $m$

(a) If the exemplar event  $e_a^i$ 's indicator  $v_a^i$  is equal to  $e_b^j$ 's indicator  $v_b^j$  at  $p$  then  $b^k = 1$  and  $b^k + 1 = 1$  and  $k = k + 2$

(b) If the exemplar event  $e_a^i$ 's indicator  $v_a^i$  is in the same WordNet synset as  $e_b^j$ 's indicator  $v_b^j$  at  $p$  then  $b^k = 1$  and  $b^k + 1 = 0$  and  $k = k + 2$ .

(c) If either of the two previous conditionals were true then

For  $s = 1$  to  $x$

For  $t = 1$  to  $y$

i. If exemplar entity  $n_s^i$  is equal to rewritten entity  $n_t^j$  then  $b^k = 1$  and  $b^k + 1 = 1$  and  $k = k + 2$

ii. Else if exemplar entity  $n_s^i$  is in the same WordNet synset as rewritten entity  $n_t^j$  then  $b^k = 1$  and  $b^k + 1 = 0$  and  $k = k + 2$

(d) If the now-point  $p$  is equal or less than  $j$  then  $p = p + 1$  and  $b_c^k = 1$  and  $k = k + 1$ . else  $b^k + 1 = 0$  and  $k = k + 1$ .

2. If the now-point  $p$  has not already been advanced due event  $i$  not being matched, then  $b^k + 1 = 0$ , with  $k = k + 1$ .

### *The Plot Comparison Algorithm*

The now-point  $p$  is currently initialized to  $e_a^1$  and transverses over events by their indicator only, not their respective entities. A counter for location in the binary vector  $b$  is given by  $k$ . Each event  $e^i$  is composed of an event identifier  $v^i$  (such as “saw”) and entities  $n_1^i, n_2^i, \dots, n_x^i$ , such as “Nils” and “stork”. Note that  $m$  is the total number of events in the rewritten story, while  $d$  is the total number of events in the exemplar story. For the purposes of this algorithm, and in the implementation, a simple coreference chain has been made for a few entities in the story and merged with the WordNet synset before running the Plot Comparison Algorithm.

The results are encoded as a binary feature vector so that they may be used with

both the Weka Machine Learning Toolkit and the OpenNLP Maximum Entropy modeller, since non-binary output may cause formatting problems with certain algorithms. However, in both those programs the features are demarcated by their unit, so that the now-point is demarcated as a “1” or “0”, while entities and events are demarcated as a class that contains “11”, “10”, or “00”. In other words, the machine-learners will not confuse the output of the plot comparison algorithms.

Below is a sample output from our algorithm given the exemplar story model  $e_a$  and a rewritten story  $e_b$ . The output feature vector  $b$ , shows the binary representation broken down at first per event indicator, entity, and temporal ordering variable and the second time with those bits merged for the complete binary feature vector.

$i,j$	$e_a$	$e_b$	$b$				
1	threw(Nils, coin)	threw(boy, coin)	11	10	11	1	1110111
2	saw(Nils, city)	walk(boy, beach)	00	00	00	0	0000000
3	enter(Nils, city)	enter(Nils, city)	11	11	11	1	1111111
4	ask(Nils, merchant)	want(everyone, money)	10	00	00	1	1000001
5	says(Nils)	flies(Nils, stork)	00	00		0	00000
6	leaves(Nils)	leaves(Nils, city)	11	11		1	11111
7	disappears(City)	talk(Nils, bird)	10	11		1	10111
8	talk(Nils, stork)	vanishes(city)	11	11	10	0	1111100
9	flies(stork)		11	11		0	11010

So the final output string of this comparison is a binary feature vector:

1110110000001111111100000100000111111011111110011010

There are a number of interesting behaviors in this algorithm. First, note that a complete match of an event is marked by a complete binary vectors of 1s and a complete failure to match is marked by complete 0s, as as in  $i = 3$  and  $i = 2$  respectively. The algorithm maintains two distinct internal counters ( $i$  for the exemplar story model and  $j$  in the rewritten story model), and a now-point  $p$  for the rewritten story model. This allows events to be detected even if they are out of order, such as  $i = 7$  and  $i = 8$  demonstrate, with  $i = 7$  matching  $j = 8$  and  $i = 8$  matching  $j = 7$ . However, there are a few limitations to this algorithm. It assumes temporal ordering mistakes usually have a local context. For example, if the rewritten story has events drastically out of order

such as having the first event last and the rest of the events relatively in order, since the now-point in the rewritten story model defaults towards first detected event, the now-point will move to the end of the exemplar story and so detect the rest of the events as out of order even though they are in order. This behavior points towards two notions of temporal ordering, a *global now-point* that discovers temporal ordering based on the global flow of the rewritten story and assumes the rewritten story is trying to follow the exemplar story closely, and a *local now-point* that keeps track of temporal ordering in context with other nearby events. For our purposes we will continue to use a global now-point in the plot-comparison algorithm since we do assume the rewritten story is trying to follow the exemplar story.

### 5.3 The Unconstrained Plot Comparison Algorithm

One obvious constraint to take away from the Plot Comparison Algorithm is its use of a now-point. Intuitively, this means two concrete changes to the operation of the algorithm. First, the algorithm no longer has to keep track of a now-point for the rewritten story and its attendant variable in the output. Less intuitively, it also means that the indicator is no longer used as the key to identify events. Since the events and their entities may be in any order, a full search of every entity and indicator of the rewritten story for a match is undertaken when searching for a event indicator or entity from the exemplar story. The entities and indicators are still searched for in the order they appear in the event structure. The now-point on the rewritten story was removed because the formalization of the now-point used by the XML pipeline is too primitive due to its inability to properly handle modality, aspect, and tense in the creation of its temporal ordering, and unable to discriminate between local now-points and global now-points. For example, in the sentence “Before the city disappeared, the merchants asked Nils for some money” would be improperly converted into event calculi by the XML pipeline into  $Disappear(t=1, city), Merchants(t=2, Nils, money)$  instead of the correct  $Merchants(t=1, Nils, money), Disappear(t=2, city)$ . This is because our pipeline can not detect the use of such temporal-ordering words as “before”. While this is a slightly more complex use of grammar than to be expected, it should not be discounted

as a plausible possibility. Secondly, there is reason to suspect that the XML pipeline would also perform poorly at accurately identifying the indicator verbs of the event calculus in raw text. Many of the verbs used in the text of the exemplar story, such as “went”, can be expressed in a large number of ways. WordNet has much poorer coverage of the synonyms of verbs than it does of nouns. Any detection that is dependent on finding the indicator *first*, and *then* the entities, might be flawed and miss out on what is actually the correct use of a synonym of the indicator verb. This would cause the algorithm to ignore its entities in the rewritten text. The constraint of finding an indicator before finding its entities has been removed, so that even if an indicator for an event is not matched, the rewritten story’s event calculi is searched for possible entities.

Using the same mathematical notation as in the original Plot-Comparison Algorithm, the Unconstrained Plot Comparison Algorithm is described below:

For  $i = 1$  to  $d$

1. For  $j = 1$  to  $m$ 
  - (a) If the exemplar event  $e_a^i$ ’s indicator  $v_a^i$  is equal to  $e_b^j$ ’s indicator  $v_b^j$  then  $b^k = 1$  and  $b^k + 1 = 1$  and  $k = k + 2$
  - (b) If the exemplar event  $e_a^i$ ’s indicator  $v_a^i$  is in the same WordNet synset as  $e_b^j$ ’s indicator  $v_b^j$  then  $b^k = 1$  and  $b^k + 1 = 0$  and  $k = k + 2$ .
  - (c) For  $s = 1$  to  $x$ 
    - For  $l = 1$  to  $m$
    - For  $t = 1$  to  $y$ 
      - i. If exemplar entity  $n_s^i$  is equal to rewritten entity  $n_t^j$  of  $e_l$  then  $b^k = 1$  and  $b^k + 1 = 1$  and  $k = k + 2$
      - ii. Else if exemplar entity  $n_s^i$  is in the same WordNet synset as rewritten entity  $n_t^j$  then  $b^k = 1$  and  $b^k + 1 = 0$  and  $k = k + 2$

*The Unconstrained Plot Comparison Algorithm*

Below is sample output from our algorithm given the same exemplar story model

$e_a$  and a rewritten story  $e_b$  as earlier. A detailed analysis of these results is not given since the Unconstrained Plot Algorithm is the Plot Algorithm without a now-point.

$i,j$	$e_a$	$e_b$	$b$			
1	threw(Nils, coin)	threw(boy, coin)	11	11	11	111111
2	saw(Nils, city)	walk(boy, beach)	00	11	11	001111
3	enter(Nils, city)	enter(Nils, city)	11	11	11	111111
4	ask(Nils, merchant)	want(everyone,money)	10	10	00	101000
5	says(Nils)	flies(Nils, stork)	00	10		0010
6	leaves(Nils)	leaves(Nils, city)	11	11		1111
7	disappears(city)	talk(Nils, bird)	10	11		1011
8	talk(Nils, stork)	vanishes(city)	11	11	11	111111
9	flies(stork)		11	11		1111

The final output string of this comparison is a binary feature vector:

111111001111111110100000111110111111111111

Notice that this algorithm allows it to identify many more entities than events. For example, when  $i = 2$ , it is unable to match the indicator of an event, but matches its entities. This allows more “matches” to be made in general than when using the Plot Comparison Algorithm, which is why this algorithm is less constrained.

## 5.4 Plot to Text Comparison Algorithm

The Plot Comparison Algorithm can become even less complex by removing the constraint that the rewritten story has to be represented as an event calculus at all. Instead of comparing the exemplar’s event calculus to the rewritten story’s event calculus, each entity and indicator of the exemplar’s event calculus is compared to every word in the original text of the rewritten story. The only pre-processing done of the rewritten story is that all of its words are reduced to their lemmas, so that WordNet-enabled synonym matching may be used to assign partial credit, and have their pronouns resolved. The intuition behind this algorithm is that the process of extracting an event calculus from the rewritten story may leave out vital information from the rewritten story due to the possibility of errors in chunking or other parts of the XML pipeline, and so an ac-

tual matching event may be marked as missing because it was not properly represented by the rewritten story's event calculus.

The following equation, following the same notation used in the Plot-Comparison Algorithm, demonstrates the behavior of the algorithm. Given a set of words  $w$  labeled  $w^1 \dots w^g$  from the rewritten story, the algorithm is described below:

For  $i = 1$  to  $d$

1. For  $h = 1$  to  $g$
2. If the exemplar event  $e_a^i$ 's indicator  $v_a^i$  is equal to word  $w^h$  then  $b^k = 1$  and  $b^k + 1 = 1$  and  $k = k + 2$
3. If the exemplar event  $e_a^i$ 's indicator  $v_a^i$  is in the same WordNet synset as word  $w^h$  then  $b^k = 1$  and  $b^k + 1 = 0$  and  $k = k + 2$ .
4. For  $s = 1$  to  $x$ 
  - (a) If exemplar entity  $n_s^i$  is equal to word  $w^h$  then  $b^k = 1$  and  $b^k + 1 = 1$  and  $k = k + 2$
  - (b) Else if exemplar entity  $n_s^i$  is in the same WordNet synset as word  $w^h$  then  $b^k = 1$  and  $b^k + 1 = 0$  and  $k = k + 2$

#### *The Plot to Text Comparison Algorithm*

Below is a sample output from our algorithm given the exemplar story model  $e_a$  and a rewritten story whose text is as follows:

*Nils took the coin and tossed it away, cause it was worthless. A city appeared and so he walked in. Everywhere was gold and silver and gold and silver and the merchant said Buy this Only one coin Nils has no coin. So he went to get the coin he threw away but the city vanished just like that right behind him. Nils asked the bird Hey where the city go? And the bird said, under the waves, deep under the waves. I was confused. But then it said to Nils Get on my back and go home. They did.*

The output feature vector  $b$  is displayed as usual.

$i$	$e_a$	$b$			
1	threw(Nils, coin)	10	11	11	101111
2	saw(Nils, city)	00	11	11	001111
3	enter(Nils, city)	11	11	11	111111
4	ask(Nils, merchant)	10	11	11	101111
5	says(Nils)	00	11		0011
6	leaves(Nils)	00	11		0011
7	disappears(city)	10	11		1011
8	talk(Nils, stork)	00	10	10	001010
9	flies(stork)	00	10		0010

The output of this comparison with the text is the binary feature vector:

1111111011110011001110110010100010

Note how this algorithm allows even more flexibility in finding events and entities. However, the disadvantage of this algorithm should also be apparent: the entities and indicators identified may have little relation to the event in the exemplar's event calculus.

## 5.5 Hand-made vs. Automatic Exemplars

There are several options as what to actually use as the exemplar story. One option would be to use the event of the XML pipeline of the storyteller's transcript. However, there are a number of factors that cause the transcript to produce an overly long and inaccurate event calculus.<sup>5</sup> The storyteller, while remaining fairly grammatical, engages in a number of storytelling techniques that while creating a better story for the pupils, create problems for the XML pipeline and the event calculus. The storyteller often has the characters engage in dialogue, and the event calculus cannot discriminate the subjects and objects of dialogue nearly as well it can from third-person sentences. The storyteller gives the pupils clues about the structure of the story, as when she says, "And that's the end of the story!". The event calculus treats this as a normal event. She

<sup>5</sup>The full event calculus for the storyteller's transcript is available on the web at [www.semanticstories.org](http://www.semanticstories.org). The text of the original transcript as cited in Appendix A.

also spends considerable time embellishing the story, such as when city is described as: “And there were people making the most wonderful jewelry with leaves on it, gold, silver, and jewels. And there were people weaving and they were weaving with silk and gold and silver”. The event calculus extractor in the XML pipeline is unable to handle this elaborate discourse, producing a repetitive and generally strange event calculus for these sections. Often objects and subjects only implicitly mentioned by the storyteller, such as when the stork says “That’s all right!” to Nils, yet the stork is never explicitly mentioned. There are two options to correct this. First, one can attempt to handcraft an event structure for the exemplar story. This was done by myself, and to construct it I limited myself to either combining elements of nearby events and deleting events in the automatically produced transcript’s event calculus. Second, we can use the event structure of the best pupil’s rewritten story as the exemplar. Unlike hand-crafting the exemplar, this method is fully automatic, and eliminates many of the problems that were created by the storyteller’s complex language and techniques. One rewritten story, that was chosen by all the raters<sup>6</sup> to be excellent, was also used as an exemplar. In summary, there are three options for exemplars:

1. Event structure of the storyteller’s transcript.
2. Event structure hand-crafted from the event calculus storyteller’s transcript.
3. Event structure automatically extracted from excellent pupil’s rewritten story

All three are used in the experiments presented in Chapter 7 of this thesis. The original text and event calculi for these all are present on the web-page [www.semanticstories.org](http://www.semanticstories.org), and the hand-crafted event structure is present in the second section of Appendix A.

Three differing plot algorithm algorithms have been presented. First, an algorithm that supports temporal representation was presented. Another algorithm took this constraint away. A third algorithm removed the constraint that the plot of the story to be compared had to even be formalized. There is a careful trade-off between the level of abstraction being deployed in each of these algorithms and how “close” they stay to the data. By dealing with more complex levels of abstraction, the ability of the algorithm to deliver exact information about matching events is increased, but so is its chances

---

<sup>6</sup>See Chapter 7 for descriptions of the rating system and its reliability

of not identifying a truly correct match. Some matches are going to be obscured by increasing levels of abstraction. However, by reconciling our algorithm with the data also leads to a possible loss of data, since while the less abstract algorithms are more likely to find correct matches, they are also more likely to find incorrect matches. Since we do not what level of abstraction is appropriate, all three algorithms will be tested in the final experiment.

# Chapter 6

## Machine-Learning

The actual science of logic is conversant at present only with things either certain, impossible, or entirely doubtful, none of which we have to reason on. Therefore the true logic for this world is the calculus of Probabilities, which takes account of the magnitude of probability which is, or ought to be, in a reasonable man's mind.

*James Clerk Maxwell*

### 6.1 Why Use Machine-Learning?

Machine-learning is crucial to our experiment, as it will allow our model to discriminate what features (such as events in a rewritten story) are good predictors of plot quality as graded by a human teacher. One objection to even using machine-learning techniques in this project is that we could use the results of the plot comparison algorithms to grade the story directly. For example, if 9 of the 10 events in the event structure are present in the rewritten story model, then the grade would be 90%. However, this approach *presupposes* the hypothesis we are trying to answer: the question is how can we grade a rewritten story in a way that correlates with a human teacher? A teacher in all likelihood does not use something exactly like our deterministic plot-comparison algorithm to grade stories; we are hoping she will use something similar. Looking at the presence of events is clearly one important factor. It is not the only one, as teacher Senga Munro notes in the guidelines for rating plot in Chapter 7: the pupil

getting the “point” of the story is more important. The general hypothesis is we *can* model the human grading of a rewritten story, and it is one particular hypothesis that the event structure and plot comparison algorithm are good predictors of human grades for the story rewriting task. Another hypothesis is that the statistical word-distributions of the entire story when reduced by LSA provides a good predictor for the grade. We hypothesize that a combination of the event calculus and LSA similarity scores will provide a good predictor of the grades a human teacher would give. A way to discover the strength of various hypotheses in a principled manner is to use machine-learning, with the features reflecting the information deemed important by a particular model and the real teacher grades being the classes of training data that the machine-learner is trying to learn. Once the various models are run together on the same data, the one with the highest accuracy and prediction naturally embodies the best hypothesis.

Another reason outlined in Chapter 3 for using machine-learning is that there are some events in the story that are more significant than others, and focusing on these relevant events may provide good predictors of actual teacher grades. Also, the exact use of some words may be crucial (as is detected by the Plot-Comparison Algorithm, see Chapter 5 for details). One principled way to model these factors is to let a machine-learning algorithm learn what features (which in the case of our current model are the presence or absence of events, the wording used, and the temporal order) are good predictors of a teacher’s grade.

## 6.2 Hypotheses in Machine-Learning

The set of all possible hypotheses can be considered  $H$ , with each possible hypothesis  $H \ni h$ . Metaphysics aside, there are underlying factors, which we assume can be modeled, that teachers actually use to grade and these factors can be considered  $t$ .<sup>1</sup> The set of models that embody  $H$  can be considered in a “hypothesis space”, and ideally we want our hypothesis  $h$  to be  $t$ , and if that is not possible, at least minimize the distance between  $t$  and  $h$ . This is minimizing the Kullback-Leibler divergence  $D(t||h) = \sum_x^n \frac{t(x)}{h(x)}$ , with  $x$  being a random variable and  $n$  being the total number of random variables.

---

<sup>1</sup>Thanks to Miles Osborne for this convention, in which  $t$  stands for the “truth”.

The Kullback-Leibler compares two distributions by comparing their relative entropy. Entropy measures the amount of uncertain information in a random variable, and can be conceptualized as the probability of the random variable multiplied by the number of unknown values a variable may have:  $Entropy(x) = -\sum_x^n p(X=x)\log(p(X=x))$ .

Since our model has no direct access to  $t$ , we must instead use the training data,  $d$ . Also, we must take our hypotheses  $h$  and create computational implementations of them, or a model  $m$  for each  $h$ . In parametric modeling, such as Naive Bayes and Maximum Entropy, it is assumed there is an underlying distribution that generates the data, and we model the parameters of this distribution. Non-parametric methods, such as K-Nearest Neighbors, do not assume a parameterized distribution (like the normal or Poisson distribution).<sup>2</sup> In the case of parameterized models, the model attempts to fit to the data by maximizing the likelihood that the model would produce the data, in a process called Maximum Likelihood Estimation, using Bayes Theorem:  $p(m|d) = \frac{p(m)p(d|m)}{p(d)}$ . Since the data  $d$  is a given, Bayes Theorem is reduced to  $p(m|d) = p(m)p(d|m)$ . Since the story rewriting task outputs scores given a story, our model is a classification model. For a given instance of new data, our model must output a class  $c_j$  from some set of possible classifications  $C$ . The ideal model in a Bayesian framework is the Bayes Optimal Classifier:  $p(c_j|d) = \operatorname{argmax} \sum_{i=1}^n p(c_j|m_i)p(m_i|d)$ . This model selects a class by combining the classification predictions of all models weighted by the probability of the model itself. However, in real modeling it is usually impossible to know  $p(m|d)$ . Since this probability of a model given the data is unknown, the models are usually trained using Maximum Likelihood Estimation, which states  $p(m|d) \sim p(d|m)$  or that the probability of the model given the data is proportional to the probability of the data given the model. This methodology is not fool-proof, since it relies on having enough of a representative amount of data to make a good estimate of the model's parameters, which may be a problem due to the small number of stories in our training corpus. Also, the real predictors of the grades that our model is trained on may not be accurately encoded as the features in our training data. This is also likely, since the features are selected (even if indirectly through algorithms)

---

<sup>2</sup>Given an increase in the data points in the training data, the number of parameters of the model does not change. For non-parameterized models the underlying model changes with the number of data points.

by a human, raising all the issues of the Frame Problem, in particular the Quantification Problem and the Significance Problem. It is also possible that  $t$  is outside the hypothesis space  $H$  that our models explore due to their mathematical assumptions.

### 6.3 Overview of Machine-Learning Algorithms

The models tested use as their features either Latent Semantic Analysis Similarity Scores, which provide a measure of word-distribution similarity, or event structures that are the result of a plot comparison algorithm.

1. K-Nearest Neighbors with Latent Semantic Analysis Similarity Scores.
2. Naive Bayes Classifier with Event Calculus and Latent Semantic Analysis Similarity Scores.
3. Maximum Entropy Classifier with Event Structure and Latent Semantic Analysis Similarity Scores.

Each of the models are explained in the next section, as well as the use of Latent Semantic Analysis for similarity scores. The training data and test data come from a set of stories graded by a scale devised by a teacher.

The models are tested for the ability to correctly classify the stories by grade, and both accuracy and precision results per class are reported. Since the corpus from which the training and test data is taken is so small, all accuracy and precision scores use ten-fold cross validation. This means 90% of the stories are training data and 10% are test data for each the runs of the experiment. The experiment is run ten times so that for each run the test data is disjoint with all test data used in other runs. The entire set of stories is sampled for test data exactly once, with the remaining data used as training data each time. The final accuracy and precision scores are the average of each of these run.

## 6.4 Latent Semantic Analysis

Latent Semantic Analysis (LSA) is a type of linear dimension reduction that projects a collection of terms and documents upon a lower-dimensional “semantic” space in which “latent” similarity relations between words are computationally made explicit. It is also called Latent Semantic Indexing (LSI) in information retrieval and is closely related to Principal Components Analysis (Deerwester et al., 1990).<sup>3</sup> The driving concept behind LSA is that words that are similar in meaning appear in the same context and LSA projects these terms upon the same dimensions, while words that do not co-occur often are mapped to different dimensions. If one accepts this assumption that words appearing in the same context are actually *similar*, LSA maps similar words to the same dimension in a lower-dimensional space, the “semantic” space. One can find a numerical measure of similarity between words or sentences by mapping them onto this lower-dimensional “semantic” space as vectors (with each component of the vector being a value of one of the dimensions in the “semantic” space) and finding the Euclidean distance between the two vectors. Whole documents and collections can be viewed as the average of the vectors of their component words. A LSA similarity score can then be taken by measuring the distance between the vectors via a cosine measurement, producing a similarity score between 0 and 1, with 1 being an exact match.

More formally, LSA takes a collection of documents and counts the frequency of terms in each document, creating a usually sparse term by document matrix  $A$  with  $n$  (term) by  $d$  (document) dimensions, where the value of  $A(n_i, d_i)$  is a function of the occurrences of each word  $n_i$  in document  $d_i$ . The document is then reduced via Singular Value Decomposition (SVD), which extracts the linear structure of the matrix through decomposing the term by document matrix into three matrices:  $A_{t \times d} = T_{t \times n} S_{n \times n} V_{d \times n}^T$ , with  $T$  and  $D$  being representations of the original term and document matrix respectively and  $S$  having the singular values of  $A$  in decreasing order, with  $S_k$  being the singular value along the  $k$  axis.  $S_k$ 's value is the amount of variation in that dimension. To project  $A$  in to a space with  $k$  reduced dimensions where  $k$  is less than the origi-

---

<sup>3</sup>One difference is between LSA and PCA is that PCA requires the matrix to be reduced to be square, while Singular Value Decomposition, which is used by LSA, does not.

nal dimensions  $n$ , one selects the first  $k$  singular values of  $S$  and creates the reduced “semantic” space  $L$  by using the reduced singular values such that  $L_{t \times d} = T_{t \times d} S_{k \times k} V_{d \times k}$ . LSA thus preserves the dimensions with maximum variance and collapses the others.

One criticism of this method is that while LSA finds the optimal solution for a given  $k$  dimensions, this optimality is based on the data having a normal distribution, whereas word-frequency data is better characterized as a Poisson or negative binomial. There is also no algorithm for finding an optimal value of  $k$ , although researchers using LSA tend to use between 100-300 dimensions as a rule of thumb (Foltz, 1996). Lastly, the accuracy of the LSA similarity scores is dependent upon a proper distribution of similar and dissimilar documents in the original matrix  $A$  (Ando and Lee, 2001). In our experiment we avoid this problem by using a very large and diverse collection of text to construct  $A$ , a million words of text taken from the recommended reading list created by Touchstone Applied Science Associates for the University of Colorado at Boulder (Landauer and Dumais, 1997).

This “semantic” space created from the corpus of recommended readings for 18 year olds collected by the Univ. of Colorado at Boulder that has been shown to be good approximation of general English usage (Landauer and Dumais, 1997), we found the LSA similarity score between each rewritten story and the exemplar story. Note that we did not compare each document sentence by sentence, but compared the entire rewritten document to the entire exemplar story, whose representations in “semantic” space were the average of the vectors of all words in the story. This result was a LSA similarity score that quantified the similarity between the exemplar story and the rewritten story as a real number between 0 and 1, with 1 being exactly the same.

## 6.5 K-Nearest Neighbors

K-nearest neighbors is a non-parametric classification algorithm that assigns the class of an unclassified test story the class of those  $k$  neighbors which are nearest to the test story through some type of distance metric. In our experiment we used the LSA similarity score to find the “distance” between stories in “semantic space”. This distance metric can be used in the K-Nearest Neighbors algorithm using LSA below:

Given a new vector  $x^b$  and a set of classes  $c \in C$ , with the training set  $x^a$  and  $a = 1 \dots M$  where each  $d^a$  is assigned a class (rank or “grade”),  $M$  is the number of training points, and  $k$  an the number of points that are used to calculate the class of  $x^b$ .

1. Retrieve the LSA Similarity score to the exemplar story from each member of the training data  $x^a$  for  $a = 1 \dots M$
2. Keep the  $k$  closest similarity scores to  $x^b$ .
3. Assign  $x^b$  the class  $c$  which is the class of the majority of  $k$  training data points collected in the previous step. If there is no clear majority, then randomly select one of the tied classes.

Using  $k$  neighbors allows the data to be less sensitive to outliers. Using  $k = 1$  often allows a single extremer to cause an incorrect classification.

Our experiment used the LSA implementation of the University of Colorado at Boulder(Landauer and Dumais, 1997).

## 6.6 Naive Bayes

Naive Bayes is a Bayesian parametric machine-learning algorithm for learning the probability of a class (such as a grade) given examples of that class, assuming that the features of the training data and the test data are conditionally independent.<sup>4</sup> The Conditional Independence assumption can be stated formally: the probability of two variables  $a$  and  $b$  given class  $c$  is  $p(a, b|c) = p(a|c)p(b|c)$ . This means the features are not dependent on each other for classification purposes. From the training data, the parameters are estimated using Maximum Likelihood Estimation, which informally involves counting the frequency of features in each class of the training data. More formally, given the training data  $x^a$  with features  $i \dots F$  divided into classes  $c_j \in C$ , and an unclassified test data vector  $x^b$  also with features  $i \dots F$ , the probability of  $x^b$  being labeled class  $C \ni c_j$  is equal to  $p(x^b|c_j) = \prod_{i=1}^F p(x_i^b|c_j)$ . The next step is to calculate

---

<sup>4</sup>In other words, have been derived independently from the identical probability distribution whose parameters are unknown.

the values of the class  $p(x_i^b|c_j)$  from the training data and assign the class with the maximum probability to the test data:  $c_j = \operatorname{argmax}_p(c_j) \prod_{i=1}^F p(x_i^a|c_j)$ . This follows from the Conditional Independence Assumption since for any two features of  $x$ , the  $p(x_1|x_2, c_j) = p(x_1|c_j)$ , so that  $p(x_1, x_2|c_j) = p(x_1|c_j)p(x_2|c_j)$ , and the Naive Bayes classifier generalizes this to  $F$  features. Due to the Conditional Assumption, Naive Bayes can classify a new test data point by comparing its features to the frequencies of features for each class and the frequency of each class in the test data. Thus, Naive Bayes is a quick machine-learning algorithm, and despite its Conditional Independence assumption, it is often very effective. For our experiments, we use both the LSA similarity scores to the exemplar story and the event structure as features in a Naive Bayes learner.<sup>5</sup> Since Naive Bayes relies solely on the training data to form its model, it can have problems modeling “scarce” data, and data that does not follow the Conditional Independence assumption. Our experiment uses the Naive Bayes implementation from the Weka machine-learning toolkit (Witten and Frank, 1999).

## 6.7 Maximum Entropy

One problem with using Naive Bayes is that the Conditional Independence Assumption does not hold for all of our data. For example, it does not hold in the case of using the binary feature vector output by the Plot Comparison Algorithm when an entity in the rewritten story model is not matched. If a rewritten story is not matched, then the algorithm will output 00 for that entity. However, the algorithm can never output 01, which means that the entity was not matched and WordNet was used to match it: a contradiction in terms. So, the second feature bit of an entity in the feature vector is dependent on the first bit. Naive Bayes would not model this dependency. However, the algorithms were set-up to avoid this and any other independence assumptions violated by 1 – in –  $M$  encoding (turning  $M$  multinomial results into a binary form). Other violations of Conditional Independence have to do with the stories themselves. Due to the causal flow of events in the story, it is highly likely that single events and entities will be forgotten by the pupil in groups that are causally connected, which is

---

<sup>5</sup>Although there are serious problems in this approach, as noted in the next section on “Maximum Entropy”

another violation of Conditional Independence. For example, the pupil forgets the city disappears, and so will leave out the conversation in which the stork explains why the city disappeared. The most frequent violation of Conditional Independence will most likely be caused by the fact that some events are more important to a particular grading scheme. For example, excellent stories may require Nils asking the stork about the city being mentioned as an event, while poor stories may always forget the stork. For example, the distinguishing characteristic of a good story may be a group of four or five events, but one of those events (such as the stork explaining to Nils why the city disappears) is crucial to the classification of a story as excellent. Results are still presented for Naive Bayes despite these failings of Conditional Independence since the performance of Naive Bayes is often surprisingly good even when Conditional Independence is violated and as a comparison to another model, Maximum Entropy, that does not make the Conditional Independence assumption.

Log-linear models do not make the assumption of Conditional Independence. The log-linear models use weighted features over a log-linear distribution, with the weights allowing the features to interact in a dependent manner. This can be formalized when given a test data point  $x^b$  each having  $1..F$  features denoted  $x_i^b$ , and a class being  $c_j \in C$ :  $p(x^b|c_j) = \frac{1}{z} e^{\sum_{i=1}^F x_i^b \lambda_i}$  where  $z$  is a normalizing constant and  $\lambda_i$  the weight of feature  $i$ . Log-linear models are not conditionally independent because the weights of all the features can model dependencies, and so model dependencies in the data the model is trained on.<sup>6</sup> Maximum Entropy is a particular log-linear model that makes no assumptions about the data other than those directly observable in Maximum Likelihood Estimation. It does this by assigning for each variable about which nothing is known an equal probability, i.e. a probability from a uniform distribution. Since the uniform distribution is the distribution that maximizes entropy, the name of the model is Maximum Entropy. It makes minimal assumptions since by maximizing entropy one is also following the Minimal Description Length principle.<sup>7</sup> As mentioned ear-

---

<sup>6</sup>One interesting metaphor from Miles Osborne for the weights is that of variables in linear equations. For example,  $x_1 + x_2 = 10$ . If  $x_1 = 3$  then it clearly follows  $x_2 = 7$ . Thus,  $x_1$  and  $x_2$  are dependent.

<sup>7</sup>Minimum Description Length is simply the inverse of Maximum Entropy. If maximum entropy is defined as maximizing the negative value of  $-\sum_x^n p(X=x) \log(p(X=x))$ , then Minimum Description Length is minimizing  $\sum_x^n p(X=x) \log(p(X=x))$ , which provides a mathematical formalism for expressing Occam's Razor and the minimalist principle we have used in guiding this project.

lier, Maximum Entropy models the distributions for particular features via weighting the features, with the weights determined by the Generalized Iterative Scaling (GIS)<sup>8</sup> algorithm that discovers the global optimal value for the weights by using Expectation Maximization without any hidden variables. This guarantees finding the global optima for the model given the training data, although if the training data is poor the model will likewise be inaccurate. Expectation Maximization progresses in two phases. In the expectation part, the expected output of the model is calculated for the given weights and features. In the maximization process, these values are compared to the data and then their values are “hill-climbed” until they coincide with the training data. However, note that GIS to find multiple global optima with differing performance (Bancarz and Osborne, 2002).<sup>9</sup> To implement a maximum entropy model, I used the Java-based MaxEnt program developed by Jason Baldridge for OpenNLP.

Three differing machine-learning algorithms with differing assumptions were presented. The first, K-Nearest Neighbors, makes no assumptions about the data whatsoever, and also uses no formal symbolic features other than a LSA similarity score. The second, Naive Bayes, makes the assumption of Conditional Independence and can use the symbolic event calculus as a feature as well as the LSA similarity scores. The last, Maximum Entropy, by virtue of being slightly more mathematically complex, does not make the assumption of Conditional Independence and can also use the symbolic event calculus. By testing all three of these machine-learning algorithms in the final experiment, we have a good chance of discovering what mathematical assumptions and features best describe children’s stories.

---

<sup>8</sup>Being related to the more popular Improved Iterative Scaling, which improves upon IIS by increasing its step-size and so making model training quicker.

<sup>9</sup>Returning to Osborne’s linear equation metaphor for Maximum Entropy, it is easy to show how multiple global optima could be reached. Given an equation  $x_1 + x_2 = 10$ , both  $x_1 = 10$  and  $x_2 = 10$  are perfectly legitimate answers that provide a global optima, as does  $x_1 = 3$  and  $x_2 = 7$ . However, if  $x_1 = 10$  and  $x_2 = 0$  is found to be the global optima and in reality in the test data both features  $x_1$  and  $x_2$  are needed for correct classification, then the model will not provide optimal performance, or even good performance. This points to a crucial subtlety when using the word “global optima” found by GIS. It is a “global optima” of the model given only the training data, not necessarily the global optima needed by real-world test data.

# Chapter 7

## Results and Evaluation

I often say that when you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind; it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced to the stage of science, whatever the matter may be.

*Lord Kelvin*

### 7.1 Ratings and Reliability

A scale for grading the plot of the story was devised with the aid of the teacher Senga Munro. The stories were gathered from Methilhill and Torbain Primary Schools, Kirkcaldy, Scotland by Judy Roberston. The classes were told a story by a storyteller, and were asked to rewrite the story in their own words. These stories are all variants of the “Nils’s Adventure”, a story from “The Wonderful Adventures of Nils Holgersson” by Selma Lagerlof(1907). The complete transcript of the storyteller’s story, the exemplar story, is available in Appendix A.

In all instances, the lead of Senga was followed in creating the rating scale. The complete text of the grading scale given to each of the raters is included in Appendix B. The scale takes the following form, with four ranks of plot. The following scale was used, verbatim, by the raters:

1. *Excellent*: An excellent story shows that the reader understands the “point” of the story, such as why particular events happened. The story should flow and make sense to the marker, and should demonstrate some deep understanding of the plot. Since the “point” is often used in the pupil’s memory to organize events, the pupil should be able to retrieve all the important links and, not all the details, but the right details.
2. *Good*: A good story shows that the pupil was listening to the story, and has recall of the main events and links in the plot. However, the child shows no deeper understanding of the plot, which can often be detected by the pupil leaving out an important link or emphasizing the wrong details. For example, a pupil can often parrot off the main events without understanding why these events are important. The story should in general still flow even if there is a lack of deeper understanding.
3. *Fair*: A fair story shows that the pupil is missing more than one link or chunk of the story, and not only lacks an understanding of the “point” but also lacks recall of vital parts of the story. The pupil’s use of supporting details will often be more patchy, with the pupil emphasizing details in the chunks they remember but forgetting entirely other parts of the plot. The fair story does not really flow.
4. *Poor*: A poor story has definite problems with recall of events, and is missing substantial amount of the plot. Characters will be misidentified and events confused. Often the child writes on the wrong subject or starts off reciting only the beginning of the story. These stories are often much shorter in length than the other types.

Three raters were used, one being the teacher Senga Munro (Rater *A*), the others being Judy Roberston (Rater *B*) and myself (Rater *C*). Rater *A* rated 46 stories according to this scale, and Rater *B* and Rater *C* rated all 102 stories. From now on the words “excellent”, “good”, “fair”, and “poor” will refer to the technical criteria in this rating scale, and not their normal English usage. Due to their frequent use, the words will not be italicized or put within quotes.

Between Rater *A* and *B* there was a Cronbach's  $\alpha$  statistic of .8840 and a Kendall's  $\tau_b$  statistic of .821 between Rater *B* and *C* there was a Cronbach's  $\alpha$  statistic of .9327 and Kendall's  $\tau_b$  statistic of .869. These statistics show our rating scheme to be fairly reliable.<sup>1</sup> Although it varies from study to study due to differing criteria, generally a reliability statistic  $> .75$  is reliable, one between  $.5$  and  $.75$  is usable with caveats, and a statistic  $< .5$  is unreliable.

At first glance at the data, it appears that there is not much agreement between Rater *A* and Rater *B*, with only 39 (18 of 46) percent agreement. Upon inspection, it is clear that Rater *A* was just systematically more harsh than Rater *B*. Stories that Rater *B* would classify as fair would be classified by Rater *A* as poor. In fact, almost half (13 of 28) of their disagreements fall into this category. Upon further inspection of the stories that were the sources of disagreement, it seems that Rater *A* would never give partial credit to an incomplete story and would always mark it as poor, even if it was very near completion. Rater *B* often gave it partial credit, marking incomplete stories as fair. Rater *A* in general tended to grade down, often grading a rating scale one less than that of Rater *B*. However, large digressions between the two were rare, as an excellent story was never identified as a poor story and only twice were stories ranked as good by Rater *B* marked as poor by Rater *A*. There were only 3 two-rank differences in total. The rest of the errors were evenly dispersed between disagreements over excellent and good stories (7 disagreements) and good and fair stories (5 disagreements). The final results are easily explainable, if one accepts the explanation that more than four-fifths of their disagreements was just Rater *A* marking one rank lower than Rater *B*.

Rater *C* and Rater *B*, who rated all the stories, had a high agreement of 77 percent (79 out of 102). Rater *C* was consistently marking stories higher than Rater *B*, with 19 stories marked higher and only 4 lower than Rater *B*. Rater *C* tended to be less harsh than Rater *B*, although he was more in agreement with her than Rater *A*. The largest difference is that stories rated as fair by Rater *B*, Rater *C* would tend to mark as good. He also tended to rank as fair those that Rater *B* would mark as poor. Only in very rare circumstances (once) did Rater *C* rate a story fair that Rater *B* would rate as poor. Again, the raters would usually differed by one ranking, and in one direction. The

---

<sup>1</sup>For explanation of why these reliability statistics were used and the differences between them, please see Appendix D.

largest area of disagreement was in between good and fair stories, with 11 disagreements, although disagreements between fair and poor stories came in a close second with 8 disagreements. Fair and excellent stories had only 3 disagreements. Overall, Rater *C* apparently was often in close agreement with Rater *B*, just giving more partial credit.

Overall, the rating scheme is far from perfect, but fairly reliable. It is imperfect enough for human raters to have relatively large disagreements on the rating scale, yet good enough that the raters tend to not disagree by more than one ranking. The rating scale, at the suggestion of the teacher Senga Munro, does take into account the comprehension of the “point” of the story for excellent stories, going beyond the recall of plot events that our event calculus was created to handle. Also, some of the differences between the raters could be explained by their experience. The least experienced rater rated stories leniently, while the most experienced rater rated the most harshly.

## 7.2 Baseline

A number of different baselines would be suitable for this the story rewriting task. The ability of the baseline to classify stories is highly dependent on the distribution of the original ratings in the data. The distribution is as follows:

Class	Probability	Number of Class
1 (Excellent)	0.175	18
2 (Good)	0.320	33
3 (Fair)	0.184	19
4 (Poor)	0.320	33

*Table of Rating Frequencies in Corpus*

Classifying all the stories as good would thus produce an accuracy of .320, as would just classifying every story as poor. Note that these two distributions are much larger than those of excellent and fair stories.

The most primitive baseline would classify a story by randomly choosing a rating

from an uniform distribution, such that each story had a 25% chance of being classified as either poor, fair, good, or excellent. This baseline performs poorly, producing only a 15.53% accuracy. A more intelligent baseline would be to take the rater preferences from the corpus into account, so that each story is classified by randomly choosing a class from the corpus distribution. A story would thus have a .175% chance of being classified as excellent, a .320% chance of being classified as good, and so on. The baseline in this case has a higher accuracy of 20.38% on a sample run using ten-fold cross-validation. This second baseline is the more sensible choice.

### 7.3 LSA with K-Nearest Neighbors

A K-Nearest Neighbors algorithm that used LSA similarity scores as its distance metric (as described in Chapter 6) was run using ten-fold cross validation with the Weka machine-learning tool-kit(Witten and Frank, 1999). The variable  $k$ , which identifies how many neighbors are taken into account when “voting” for the classification of a new test point, was varied from 1 to 5. In this range, the greatest percentage correct was identified when  $k = 4$ . Overall, it correctly classified 44.7% of the stories, and incorrectly classified 55.33%. Precision and recall rates per grading rank are presented below:

Class	Precision	Recall
1 (Excellent)	0.111	0.167
2 (Good)	0.417	0.455
3 (Fair)	0.3	0.158
4 (Poor)	.833	0.758

*K-Nearest Neighbors: Precision and Recall per Rank*

From this chart, it can be seen that overall LSA and K-nearest neighbors classify the stories fairly poorly overall. However, a number of trends can be observed. The actual recall and precision varies highly per class. Excellent stories were almost never classified properly. Stories that were rated as poor were generally classified correctly.

Stories rated as good were correctly identified almost half the time, and stories that were fair were almost as difficult to identify as excellent stories. A confusion matrix for the scores was also produced:

Class	1	2	3	4
1 (Excellent)	3	10	4	1
2 (Good)	13	15	2	3
3 (Fair)	9	6	3	1
4 (Poor)	2	5	1	25

*K-Nearest Neighbors: Confusion Matrix*

From this confusion matrix, it should be clear that the LSA similarity score by itself is not a perfect metric for classifying stories. First, unlike human raters, K-nearest neighbors has errors spread throughout the entire range of grades. It rarely classifies excellent stories as poor and poor stories as excellent stories. It also seems to have especially little grasp of what makes an excellent story, classifying most excellent stories as good stories, which is quite unusual, since it classifies some good stories as excellent. It seems mysteriously very lenient on fair stories, classifying them as excellent. These errors can be mostly traced to an overly mixed distribution of LSA similarity scores in the “semantic space”. The poor stories form one loose cluster, and all the other stories form another loose cluster.

From this analysis, the K-Nearest Neighbors algorithm can be on some level be relied upon to classify stories as either poor or non-poor, although even with poor stories there is a small chance the story will be misclassified dramatically. This confirms our earlier theoretical suspicions of both the limitations LSA as a metric for measuring story coherence. Poor stories tend to be either very short, incomplete, or simply off-subject. This leads them to have radically differing word distributions than other stories, so poor stories cluster together in LSA’s “semantic space”. This allows them to be easily identified by K-Nearest Neighbors. However, the criteria used to identify other types of stories are much more subtle, as these stories often use many of the same general words as the exemplar story, but vary in which parts of the story they recall and their understanding of the “point” of the story. These stories would on a coarse-level

have similar word-distributions since they at least agree on the subject matter, and their defining features would be more fine-grained, involving causality, temporal ordering, and particular incidents being present. Non-poor stories cluster around each other, and stories that are numerically dominant (such as good stories) will dominate the results. The similarity scores given by LSA does not give enough information to compare and rank these stories very precisely.

## 7.4 Naive Bayes

The XML pipeline produces an event calculus that we predict will more effectively discriminate between the various rankings of a story than just word distributions in the “semantic” space produced by LSA. This is because an event calculus is a more plausible model of narrative similarity than word distributions. After extracting the event calculus automatically from each story, we compared each story to the exemplar using all three variants of the Plot Comparison Algorithm (as detailed in Chapter 5). The results of these plot comparison algorithms, in combination with the LSA similarity scores, is given to the Naive Bayes machine-learner via ten-fold crossover. Over all models, each Naive Bayes model with the event calculus outperformed the K-nearest Neighbors. The full lists of models with exemplar types and their performances is given by the below table.

Note that the exemplar types were varied, with three distinct exemplars being tested: “Automatic”, “Handcrafted”, and “Best Story”. As explained in Chapter 5, the “Best Story” exemplar refers to a hand-chosen best story (all raters agreed it was excellent) from the corpus. The “Automatic” exemplar was automatically created via running the XML pipeline to extract an event calculus from a transcript of the story told by the storyteller to the pupils. The “Handcrafted” exemplar was created by reducing the event calculus of the “Automatic” exemplar by hand.

Type of Algorithm	Automatic	Handcrafted	Best Story
Plot Comparison Algorithm:	51.46	50.49	53.40
Unconstrained Plot Comparison:	51.46	58.25	49.51
Plot to Text Algorithm:	54.37	55.34	52.47

*Naive Bayes: % Classified Correctly by Algorithm and Exemplar*

The worst performing Naive Bayes learner was the Unconstrained Plot Comparison Algorithm with the “Best Story” exemplar, while the best was the Unconstrained Plot Comparison Algorithm using the “Handcrafted” exemplar. The best non-handcrafted result, and thus the one most likely to be used in any fully automated version of the plot-checker, was the Plot to Text Algorithm with an “Automatic” exemplar that scored 4% less than the best handcrafted result.

These results are difficult to interpret, although a few general points can be made. First, all of these algorithms outperform the K-Nearest Neighbors with LSA algorithm by at least 5% but more often by about 10%. Due to this improvement in performance, there must be relevant information in the event calculus that is not present in the LSA similarity scores. The unmodified Plot Comparison Algorithm, despite its restrictive now-point rules, performs fairly well with remarkable little variance over exemplar types. The Unconstrained Plot Comparison algorithm has wide-ranging variance, having both the best and worst error rates. The Plot to Text Comparison Algorithm is more stable and generally outperforms the Plot Comparison algorithm. The interactions between the various comparison algorithms and their exemplars provide no consistent insights. Handcrafted exemplars tend to perform better than “Best Story” and “Automatic” exemplars, but little else is obvious.

Class	Precision	Recall
1 (Excellent)	0.444	0.222
2 (Good)	0.451	0.697
3 (Fair)	0.545	0.316
4 (Poor)	0.813	0.788

*Handcrafted Naive Bayes: Precision and Recall per Rank*

The results of Naive Bayes seem to closely follow the results of the K-Nearest Neighbors algorithm, with noticeable improvements. The resolution of the classifier has improved significantly. First, only on good stories are the errors of more than two ranks, with every other classification error being only off by one rank. This is a large improvement, since it allows the machine-learner to classify with a level of accuracy that reflects a human rater. Naive Bayes has an advantage in being able to accurately classify the large majority of poor stories and the large majority of good stories.

Class	1	2	3	4
1 (Excellent)	4	13	1	0
2 (Good)	5	23	2	3
3 (Fair)	0	10	6	3
4 (Poor)	0	5	2	26

*Handcrafted Naive Bayes: Confusion Matrix*

However, the excellent and fair stories present a problem for Naive Bayes. Excellent and fair stories were most often just classified as good stories. Two distinct clusters of classes have developed (much like with K-Nearest Neighbors), one cluster for poor stories and another for good stories. The crucial advantage that Naive Bayes has over K-Nearest Neighbors is that its clusters are much more precise, making fewer mistakes because it classifies all non-poor stories as good stories. This delivers surprisingly good results, mainly because the good stories outnumber the fair and excellent stories by a considerable margin. Because of this, the mistakes it does make are less severe than those of K-Nearest Neighbors. The same general pattern holds in the fully automated Naive Bayes machine-learner that uses an automatically generated event calculus instead of a handcrafted one.

Class	Precision	Recall
1 (Excellent)	0	0
2 (Good)	0.433	0.879
3 (Fair)	0.455	0.263
4 (Poor)	0.917	0.667

*Automatic Naive Bayes: Precision and Recall per Rank*

The results of using an automatic exemplar compare well with those achieved using an hand-crafted exemplar. The difference in accuracy is shown to be somewhat illusory: the automated Naive Bayes simply does not detect excellent stories, and simply collapses the distinction between good and excellent stories. It is even better than the “handcrafted” Naive Bayes in identifying good stories, although it mistakenly classifies some poor stories as good. I think that erring on the side of leniency is better than misidentifying a good story as a poor story.

Class	1	2	3	4
1 (Excellent)	0	17	1	0
2 (Good)	1	29	2	1
3 (Fair)	0	13	5	1
4 (Poor)	0	8	3	22

*Automatic Naive Bayes: Confusion Matrix*

The fully automatic Naive Bayes does not identify any stories as excellent (except one mistake), and consistently identifies excellent stories as good. It also identifies correctly almost all good stories, but classifies most fair stories as good. Lastly, it identifies most poor stories correctly as poor, although not with the same level of accuracy as the best Naive Bayes algorithm or even LSA with k-nearest neighbors. What is happening is that the good and poor stories are dominating the training data, and thus the statistical model, causing most stories to be split fairly accurately into either class. Yet with the exception of the good and fair categories, its misclassification errors are remarkably low compared to K-Nearest Neighbors.

## 7.5 Maximum Entropy

As explored in length in Chapter 6, Naive Bayes is called “naive” because it assumes all the features, such as the events of a story in our event calculus, are independent. One would suspect that some events in a story are not independent, and that by introducing some type of explicit weighing events that ranks events for importance would help to classify them correctly. For example, the event of the stork explaining to Nils why the city disappears is most likely to be *the* defining event of an “excellent” story, and the presence of this event in the event calculus should be weighed appropriately when classifying stories. A machine-learning algorithm that does this weighing automatically, Maximum Entropy, was run over the same data. It produced the following percentages of correctly classified stories:

Type of Algorithm	Automatic	Handcrafted	Best Story
Plot Comparison Algorithm:	57.28	47.57	54.36
Unconstrained Plot Comparison:	60.19	58.25	56.44
Plot to Text Algorithm:	52.43	52.43	53.40

*Maximum Entropy: % Classified Correctly by Algorithm and Exemplar*

Maximum Entropy generally did better than Naive Bayes. Only in a few instances, such as when using hand-crafted exemplars or the Plot to Text algorithm, did Maximum Entropy do worse than Naive Bayes. The greatest gains in performance were achieved with the automatically constructed exemplars using either the Plot Comparison Algorithm or the Unconstrained Plot Comparison Algorithm. The results of the Unconstrained Plot Comparison Algorithm with an “Automatic” exemplar, which had the best performance, will be inspected.

Class	Precision	Recall
1 (Excellent)	0.571	0.444
2 (Good)	0.486	0.515
3 (Fair)	0.257	0.474
4 (Poor)	0.757	0.848

*Maximum Entropy: Precision and Recall per Rank*

The recall and precision of poor stories remains very good, and the recall of both good and fair stories has noticeably gone up, almost to fifty percent. However, this is done at the expense of accurately identifying good stories, whose recall goes down considerably from that of the Naive Bayes classifier. From these results it is clear that some events in the event calculus definitely do have dependent relationships, and so some events are more “key” in classifying plot quality than others. This makes sense given that both the excellent and fair ranks categorize stories by having certain key casual relations or “flows”. Unlike Naive Bayes, Maximum Entropy is able to create a statistical model, if not a very accurate one, of these fair and excellent stories.

Class	1	2	3	4
1 (Excellent)	8	9	1	0
2 (Good)	6	17	4	6
3 (Fair)	0	7	9	3
4 (Poor)	0	2	3	28

*Maximum Entropy: Confusion Matrix*

The confusion matrix presents a less clear picture. Errors tend to be much more scattered than in Naive Bayes, especially with good stories. The biggest gains are in the fair and excellent categories, and the variance in poor stories is identified correctly more often than in Naive Bayes. While the clustering into good and poor stories performed by Naive Bayes is still present, compared to Naive Bayes, Maximum Entropy has a much greater tendency to mislabel stories as wrong by a considerable margin. For example, six good stories were labeled as poor by Maximum Entropy, as compared to

only one by Naive Bayes. Unlike Naive Bayes, Maximum Entropy can build a model of excellent and fair stories. However, due to relatively small numbers of excellent and fair stories in the corpus, the statistical model is inaccurate.

Maximum Entropy can only model what is in the data, and makes no additional assumptions. This is the reason for both its success and failure. For a small data set (only 102 instances) and a large, if sparse, feature space (around 100-200 features per algorithm), Maximum Entropy's Generalized Iterative Scaling is both overwhelmed by the number of features and underwhelmed by the lack of training data. Thus, it may overly weigh dependencies that may not be nearly as significant in the test data, while ignoring important dependencies that are not evident in the test data. If the rating scale is stable and the features capture the relevant properties of the plot, then with more training data one would expect Maximum Entropy to become more successful in ranking plot.

## 7.6 Evaluation

The best results of all three algorithms are summarized in the table below:

Type of Machine-Learner	Correct Number	Correctly Classified
K=4 Nearest Neighbors with LSA	46	44.6602%
Naive Bayes Unconstrained Handcrafted	59	57.2816%
Naive Bayes Plot-to-Text Automatic	56	54.3689%
Maximum Entropy Unconstrained Automatic	62	60.1942%

### *Final Comparison of Machine Learners*

In our final analysis, the particular type of machine-learning outweighs in importance which plot comparison algorithm or exemplar is used. Also, what type of machine-learner was used was outweighed in importance by the features given to the machine-learner. The addition of the event calculus led to significant gains over just using the LSA similarity scores. This confirms our methodological hypothesis that using hybrid models would lead to better performance than using only statistical models.

Other tentative conclusions may be drawn. The now-point used in the original Plot Comparison algorithm is, as guessed, too restrictive. The ideal algorithm for comparing plots is most likely the Unconstrained Algorithm, although there are interactions between the exemplar story and the algorithm that make a thorough analysis difficult and the results uneven.

Regardless of the algorithm used, even in the worst case the addition of an event calculus and some type of plot comparison algorithm helps classification accuracy. Holding the machine-learning model constant and inspecting the algorithm and the exemplar, there is surprisingly little variance. The classification success in almost all cases are within ten percent of each other for a given machine-learner, varying the algorithm and exemplar only impacts the results by a few percentage points. Lastly, while the change from K-Nearest Neighbors to Naive Bayes was a significant improvement, Maximum Entropy modeling only managed a relatively small improvement given its superior mathematical assumptions. While the machine-learners all score well above base-line, none are able to produce truly spectacular results.

Besides lack of training data, there are other reasons why the results are more moderate than spectacular. No machine-learner will improve if the feature set does not adequately capture the characteristics of the rating ranks in the mind of the rater. While the event calculus captures some of the relevant defining characteristics of stories, it does not capture all of them. The types of stories that give the machine-learners the most difficulty are those which are excellent and fair. One reason is that these stories are numerically less significant in the training data than poor and good stories, and so any machine-learning model of them will be less accurate since it had less training data from which a statistical model can be built. Another reason is that there are features particular to these stories that are not accounted for by an event calculus. Both excellent stories and fair stories rely on very subtle features to distinguish them from good and poor stories. Good stories were characterized by the teacher Senga Munro in the grading criteria used by the raters as “parroting off of the main events”, and the event calculus naturally is good at identifying this. Poor stories have “definite problems with the recall of events”, and so are also easily identified. They have much different word distributions than the rest of the stories. However, fair stories show both a lack of

“understanding of the point” and “do not really flow”. In contrast, the excellent story shows an “understanding of the point” and the pupil should “retrieve not all the details, but the right details”. Clearly, these characteristics involve not just recall of events, which is captured by the event calculus, but higher-order relations such as the “point” of the story and connections between events. These types of relations are considered, as examined in Chapter 2, coherence as opposed to the event-based cohesion. The excellent story would, among other relations, have the macro-structural relations of a “beginning”, “climax”, and “end”, while the fair story would not. While cohesion is often a good predictor of coherence, it is not the same thing (Hickmann, 2003). Until progress is made in identifying these so that they can be extracted as features, and work has been done in this direction (Teufel and Moens, 1999), further progress needs to be made for these techniques to be implemented on children’s stories. The event calculus is a necessary but not sufficient formalism for characterizing stories.

Another possibility is that the grading criteria themselves have problems. The grading of stories, especially defining the truly excellent story, is obviously a subjective process. There is no exact criterion for what makes an excellent story, and the diverse criteria can range from “flowing” to “getting the point”, all very informal concepts. There are some stories that are not captured adequately by the grading criteria at all. One noticeable disagreement is over what to do with incomplete stories, in which the pupil begins writing the story but does not finish. Rater *A* always labeled them as poor while Rater *B* would give them partial credit, labeling them fair. Incomplete stories should be either dealt with through macro-structural features, removed from the corpus, or given their own “Incomplete” category. The criteria for excellent stories is in need of refinement. Lastly, there is an upper limit to the reproducibility of this grading. The agreement between human raters ranged between at worst 39% and 77% at best, so it is unsurprising that the best machine-learner has an agreement of 60%. If the machine-learner scored much lower, would might suspect it is not learning anything, but if it scored higher one might suspect cheating. The machine-learners seem to be classifying to the best of their ability given the feature set and the rating scale. Their successful classification of the stories into good and poor stories is significant, not in the least because the results confirms both our hypothesis that hybrid models are better

than just statistical models and that a machine-classifier can provide feedback similar to that of a trained teacher.

This can be viewed as a way of doing machine-learning over small data sets. The reason the event calculus performs better than the pure statistical methods like LSA is that it *pre-selects* the right features for the problem. Normally, in statistical machine-learning, one “throws” as much data at the learner as possible, and the learner discriminates the best features for the problem. This requires huge data-sets, and with a small data set such as ours this is impossible. For example, if one were treating a child’s story as a “bag-of-words”, the typical story would have at least over five hundred words, with excellent stories sometimes near a thousand. This compares to the hundred to two hundred features that the event calculus outputs from the plot comparison algorithm. A typical machine-learner would not have enough training data to sort through the thousands of features required by a “bag-of-words” approach. A machine-learner *can* form a reasonable model if the feature space is made much smaller, by using a symbolic model such as the event calculus to filter out the “noise” of irrelevant features. The entire point of LSA is to reduce that “bag-of-words” to a smaller feature set. However, what features LSA blindly selects the event calculus carefully selects. The LSA features have no “meaning”, while the event calculus features are specific events that our theory suggests has an effect on plot. Since our theory was carefully thought through, we chose the right features by using the event calculus. In the end, choosing the right features intelligently through a sensible symbolic framework not only helps explains the results, but delivers better results.

# Chapter 8

## Conclusions

The true difference in nature is not between the symbolic and the imaginary, but between the real elements of the machine, which constitute the production of desire, and the structural whole of the imaginary and the symbolic, which forms stories and their variants.

*Gilles Deleuze and Felix Guattari, "Anti-Oedipus"*

### 8.1 Conclusion

A hybrid model combining both symbolic and statistical knowledge does to a degree allow automated teacher-like ranking of plots in the story rewriting task. First, it successfully extracts a symbolic knowledge representation, the event calculus, from the raw text of each rewritten story using a XML pipeline. This gives the system developed here an advantage over many other story understanding systems in that it can deal with real text as typed by children. It also gathers LSA similarity scores between the exemplar story and each rewritten story. The event calculus of each of the rewritten stories is compared to the event calculus of the exemplar story using a number of symbolic plot comparison algorithms. A rating scale used to measure the performance of the machine-learners was devised with the help of a teacher. A K-Nearest Neighbors classifier was built using only the LSA scores. The results of these plot comparison algorithms were used with the LSA scores as features for Naive Bayes and Maximum Entropy classifiers. These classifiers, which embody a hybrid framework that attempts

to overcome classic problems within artificial intelligence such as the Frame Problem, outperformed the K-Nearest Neighbors classifier. The Naive Bayes and Maximum Entropy classifiers classified the stories into poor or good stories successfully, although further fine-grained classification proved to be difficult. In final analysis, since the performance of the best machine-learners approaches the reliability of human raters, the project may be viewed as a success. The system itself is very complex, with many of its sub-components such as knowledge extraction from raw text and anaphora resolution being quite difficult problem in of their own right.

There are important limitations and strengths to the particular model we have pursued in this thesis. In this task, although the Maximum Entropy machine-learner has the best overall accuracy on the corpus, for implementation purposes with StoryStation the Naive Bayes machine-learner has definite advantages. Unlike Maximum Entropy which complexly clusters the data, Naive Bayes makes very few errors in distinguishing between poor stories and good stories by keeping its division binary. Since both Maximum Entropy and Naive Bayes are Bayesian, they give not only the results but a probability of the classification being true. Although inspection of these particular probabilities is beyond the scope of the present study, they would present interesting options to the algorithm. The system could be made to deliver only results which it is fairly sure of, and ignore or ask for human assistance on others. Naive Bayes is much faster to train on new training data than Maximum Entropy. The Unconstrained Plot Comparison Algorithm allows the entire process to automated, without any manual work by the teacher other than giving StoryStation the text of the exemplar story, which can just be the transcript of the exemplar story in the story rewriting task.

The real issue is not one of just grading the rewritten stories, but of adapting the ratings given by the machine-learner to the context of StoryStation. If the plot analysis system was just a machine-learner that was used only by teachers to give grades to pupil's stories, the machine-learning solutions explored would not be up to the task, since it can only reliably distinguish two types of stories: good and poor. However, the machine-learner is embodied as an agent that interactively helps the pupil with their writing. As such, the ability of it to accurately identify poor stories allows it to identify the pupils that are in need of help. It could remind the pupil that they may have for-

gotten a part of the plot or give a pupil who is struggling to write the story suggestions about what part of the plot to focus on next. The agent could suggest to pupils who are having difficulty to ask for help from a teacher. While the automatically extracted event calculus is too simple a representation to capture many important elements of the plot, its results can be very useful if viewed as a source of concrete suggestions to a pupil. These suggestions could be embodied using techniques from natural language generation. This is one advantage our hybrid model gives the plot analysis agent for the story rewriting task over a non-hybrid statistical model. The agent can use statistical techniques to help identify the story, and since it uses a symbolic knowledge representation, it can identify<sup>1</sup> what parts of the plot are present and which parts are not. Such an agent could develop a feedback-loop with the student and their rewritten stories, using stories graded by teachers to improve its accuracy and using its agent embodiment to communicate possible suggestions for improving the plot to the pupil.

## 8.2 Future Work

The comparison of plots of children's stories is clearly a field that is in need of further research, but the results of this thesis are a solid foundation for further work. One problem that has plagued many story-generation and story-understanding programs is lack of any realistic implementation, and this is both one of the main strengths and weaknesses of this work. By not dealing with any idealized "deep" representations, our system is able to both bypass to an extent the Frame Problem and its attendant knowledge engineering bottleneck. By dealing with the actual text of stories, we can use statistical techniques that require word counts, such as LSA. There is still much work to be done, but such a plot-analysis agent in StoryStation is a social good and can serve as a useful demonstration of artificial intelligence and natural language processing to a "real" world problem. This is in contrast to more traditional story-understanding and story-generation systems, which are usually used as testing grounds for theoretical ideas in artificial intelligence.

Due to its practical nature, the plot analysis of our system is very limited in na-

---

<sup>1</sup>With some degree of error given by the inexact results of the XML pipeline.

ture, focusing on just the story rewriting task. Traditionally, “deep” representation systems have attempted to be powerful general-purpose story understanding or generation systems. Clearly, a general plot-analysis agent would be much more useful than our current system, which is only functional for the much easier story rewriting task. The possibility of generating a full-fledged story-understanding agent is increased by our success in the story rewriting task, but there are even larger problems to tackle with it. With no story to compare to, how does one judge quality? While we can detect cohesion with our event calculus, there is still no way to automatically detect coherence. Also, the ability to detect “common-sense” logic and “fantastical” logic become much more crucial in story understanding. The entire plot analysis system presented here have yet to fully integrated into StoryStation, and so no results pertaining to its ability to give feedback to and interact with children are available. Also, the current system is exceedingly complex, composed of many programs that will be difficult to integrate in a way that will be easily usable by schools. Many of the elements, in particular LT XML, LT TTT and the Cass Chunker in the XML-pipeline, are not Java based. Another option would be to use an architecture that allows multi-platform communication such as the OpenAgent architecture.<sup>2</sup> StoryStation is used in schools, often without reliable Internet connections, it may be easier to stay with one coherent programming language and a design that does not rely on using the Internet, so that StoryStation can be installed and configured easily. It is often these practical questions that will determine if the system is actually used in a “real-world” context, and it is this use more than any theoretical use that will prompt the most interesting development in story understanding systems.

There are also many technical aspects of the current system that could be improved. The Plot Comparison Algorithm itself could be substantially upgraded, as could the event calculus extractor beneath it. The limits of the various plot comparison algorithms are mostly related to the advantages and limits of the event calculus, which is in turn dependent on the XML pipeline and what information is chosen to put in the event calculus that can be extracted from raw text. There is not yet a broad coverage way of identifying temporal information in free text, although there has been some initial

---

<sup>2</sup>For more information see <http://www.ai.sri.com/oaa/>.

steps in that direction (Hachey and Korycinski, 2003). Due to this limitation, the event calculus is unable to accurately represent the flow of time in a narrative, even when using a “now-point”. Many of the theoretical mechanisms needed to represent the stories are not present in the event calculus. Also, our system is unable to represent the idea of embedded stories or dialogue, much as Rumelhart’s original story grammars (1975). Theories such as those of Segmented Discourse Representation Theory have made these issues theoretically clear, but they lack a computational implementation that can deal with possibly ungrammatical raw text (Asher and Lascarides, 2003). Some problems also lack any strong theoretical basis, such as the concepts of local cohesion and global cohesion. This theoretical fuzziness has definite computational consequences: due to an inability for theorists to define levels of temporal cohesion, computational work cannot be further pursued in this area.<sup>3</sup> Other problems are merely technical: the coverage of WordNet is often limited for certain words, and certain synonyms are limited to particular stories and their co-reference chains. The greedy behavior of the plot-comparison algorithm and its inability to group events into episodes also doubtlessly hurt its performance. An ability to do a more fine-grained episodic analysis of events, such as checking just the plot of particular sections of the story, could be very useful. Lastly, the performance of the machine-learners, especially Naive Bayes and Maximum Entropy, will be usually correlated directly with the number of stories in the corpus, which is currently very small.

### 8.3 Meta-theorizing and Artificial Intelligence

As Charniak discovered, appearances can be deceiving: children’s stories are not easy for a computer to understand (1972). As noted by Dreyfus, the failure of Charniak’s thesis to understand stories in a frame-based system was one of the first death knells of traditional symbolic artificial intelligence (1992).<sup>4</sup> If a computer cannot model a child’s story, then how can it possibly model the much more complex real world?

There are a number of crucial errors in this viewpoint. First, children’s stories are

---

<sup>3</sup>See the discussion in Chapter 5 for the local vs. global now-point problem.

<sup>4</sup>Which can perhaps explain why Charniak is currently a currently a vocal proponent of statistical natural language processing.

*complex.* Children are not simplified versions of adults, children are living human beings that are dynamically developing, and their stories embody complexity as much as any other natural phenomena. Their pronoun usage may be simpler, but their syntax tends to have an ungrammatical but endearing complexity all of its own that foils traditional parsers. The fantastical logic of children's stories violates "common-sense" logic in every other sentence, yet it is as deep and as mysterious as the myths studied by Claude Levi-Strauss. It is no wonder that children's stories defeated the efforts of some of the best minds of artificial intelligence.

All hope is not lost. The ability to use the failures of the past, such as story grammars, to develop a theory about theories, a *meta-theory*, is important. The proposal for hybrid systems, that this plot analysis system is just an example of, is a meta-theoretical or meta-methodological proposal. Notice that through the entire thesis, care has been done to find the exact level of abstraction that suits the problem. The levels of abstraction involved in the story rewriting task were carefully analyzed in Chapter 3. In Chapter 5, three different plot comparison algorithms were systematically developed that each operated on a varying level of abstraction. In Chapter 6, three different machine-learners were developed to show how an hybrid system (Naive Bayes and Maximum Entropy with the event calculus) could out-perform a non-hybrid system (K-Nearest Neighbors with only LSA similarity scores). These machine-learners were not chosen randomly, but each embodied differing levels of complexity mathematical assumptions. Although the final results showed how interwoven models, complexity, and abstraction are, the final results also clearly showed that some levels of abstraction are clearly better suited for this problem than others. The thesis was not just about the story rewriting task, but also about the nature of abstraction and models in artificial intelligence.

The field of natural language processing historically is divided between those who believe that statistical modeling is the only way forward and those who continue to deal in symbolic logic and linguistics. This is a meta-theoretic controversy. We proposed that both these approaches are important, and that they can engage in a mutually beneficial relationship through hybrid models. In this particular experiment, statistical models such as Naive Bayes and Latent Semantic Analysis were successfully com-

bined with a symbolic knowledge representation of children's stories, the event calculus, which created a successful AI agent for grading the plot of children's stories in the story rewriting task. The statistical model allows classifier systems to be easily built that incorporate informal human knowledge, and the symbolic model gives us some insight into what features of a story make one story better than another. The problem with traditional symbolic approaches such as frames is that they can never find quite the *right* level of abstraction. As the Frame Problem notes, they are unable to represent a complex and dynamic world. Finding the perfect frame is like finding a perfect lover or perfect job, an impossible task. Statistical models are often akin to a "witches brew", they often give good performance, but rare is the researcher who can explain why. By recognizing their complimentary advantages, we can use a mixture of statistics and symbolic frameworks to create *flexible representations* that change in response to real-world dynamic environments. Such flexible representations capture via statistics aspects of the data that are difficult to formalize without surrendering the insight of a symbolic framework. The event calculus representation is excessively simple. Yet, one can imagine much more complex knowledge representations with adequate conceptions of time, reasoning, planning, and agency would be more successful. These levels could even work on multiple levels of abstraction, mediated by layered learning, creating dynamic hierarchies of representation that will surely capture the nature of stories better than the framework presented here.

These meta-theoretical inquiries are vital to practical language engineering. In this particular problem, our corpus was tiny compared to most corpora used in machine-learning. If we were to stick to traditional "bag-of-words" techniques, our approach would have failed due to lack of training data. Even LSA, the intelligent way to approach a "bag-of-words", performed poorly by itself. Using any of the complex symbolic models traditionally assumed in story-understanding would have made it impossible to deal with the ungrammatical, creative, and raw text of children's stories. Combining them using meta-theoretical insights about their relative strengths and weaknesses proved to be successful. I believe a similar approach would work well on other moderate sized corpora where a sensible symbolic framework can be found.

Even then, there are parts of a story that escape our hybrid models, even with the

envisioned possible extensions. In the corpus, one child uses the word “crinkle” to describe the noise Nils’s shoe makes when it hits the coin. Another story in the corpus changes the entire plot of the story, having Nils’s wake up in “St. Elf’s Academy”. These acts of imagination should not be penalized, yet because they leave the proper domain of the story rewriting task, the first example goes unnoticed by the machine-learners and the second is ranked as poor by both raters and machine-learners. Imagination escapes easy formalization, although progress has been made in attempting to formalize these concepts (Ritchie, 2001). Even with the most sophisticated models, the unusual and the extraordinary have a tendency to disrupt the most well-laid formalisms.

One should not forget there are social aspects to allowing a computer to grade stories. Do we really want our children to be graded by machines that may not recognize important factors in plot such as creativity? Given the current ratio of teachers to students, it makes sense that a teacher would like some automation to alert them to students that are having serious problems with their writing, so that the teacher can concentrate on the struggling student. Such struggling students often lack the metacognitive ability to perceive that they are struggling, and so often do not ask the teacher for help themselves (Luckin and Boulay, 2001). It also allows the teacher to focus their attentions on teaching the human element of writing, emphasizing creativity and quality writing technique. Although machines may never be able to grade stories with the same care as humans, one can imagine interactions between children and machines that allow both to grow and reach new heights of complexity and skill in development. As mentioned before, StoryStation is not intended to grade stories per se, but allow the pupil access to an artificially intelligent agent that can produce constructive feedback on the stories to the pupil. An animated agent may remind a child of a forgotten point or help a child with a particularly difficult word, and the child’s writing will in turn influence the behavior of the agent, allowing the agent to learn how human writing works better. For example, every new story could be added to the training corpus of the machine-learner.

The search for a perfect model of human stories on computers may be doomed to failure by its nature, but the study of the interaction and mutual development of hu-

mans and computers will lead to more insightful research in artificial intelligence, a post-humanist cognitive science that is concerned with evolving communicative networks. Story understanding, of which this story rewriting task is but a sub-domain, led artificial intelligence researchers to realize that they were in the dark about essential aspects of intelligence. The ability to narrate events, to tell stories, is fundamental to human intelligence(Charniak, 1972). In this thesis, we propose and build hybrid frameworks that can lead to a better understanding and appreciation of computation and narration. While our plot analysis system here is far from perfect, its creative use of statistics, symbolic frameworks, and human knowledge allow us to make progress on what appears to be an impossible task. We hope this system will serve as a framework for future work into narration and computation, as various components may be refined and added. Narration and computation are *difficult* subjects as the failure of Story Grammars and the Frame Problem show. It is an AI-complete problem, and artificial intelligence has barely scratched the surface of its depths. However, rewarding applications like StoryStation can be enhanced through careful investigation of stories and computation. There's one part of nature that we must not forget. The darkest hour always comes before the dawn.

# Appendix A

## Exemplar Story

This is the complete text of the story from “The Wonderful Adventures of Nils” that serves as the exemplar story in the story re-writing task(Lagerlof, 1907).

### A.1 The Text

This story takes place on the day before Easter. Now, Nils Holgersson was changed into an elf so that he could fly on the back of the geese and travel all over adventure and have adventures.

Now, it was Easter Eve and as everybody knows, on Easter Eve there is always a full moon. And it was getting dark and the geese flew back to the mountain where they rested each night. And Nils jumped off the goose’s back and immediately changed into his boy size.

And he lay on the soft of tufty grass and he didn’t feel like going to sleep at all!

And he looked up at the moon.

And he saw a shape silhouetted against the moon, a black shape against the yellow moon. And it had a long neck, a long beak, it had wide wings and it had long legs just tucking themselves under. I wonder what it was. What do you think it might be?

It was a stork, and storks are believed to be very lucky in Sweden.

So the stork came nearer and nearer and he landed. And he stood on one leg.

And he looked at Nils and said “How about going for a wee trip?”

And Nils was up for that. He said, “There is only one thing, there’s only one thing...I’ve got to be back before dawn”.

“That’s all right!”

And this was fine because the moon was shining, it was like day!

So. Onto the stork’s back and of course he changed back into his elf shape. And off they went.

And they flew from the mountain and you could see when you looked down a patchwork quilt of fields and rivers! Little houses and churches with spires.

And they flew following the river all the way down until they got to the sea.

And the sea, it was wonderful, because the moon laid a silver path all the way across the sea to the land.

And they landed!

And Nils thought, “Oh, I’ll just slide down these dunes!”

And the stork said, “Well, I’m going to have a little rest”. And then Nils thought, “No, I’m not. I’ll just explore a minute.”

And he just walked along and his wooden shoe hit a coin...clink! He looked down and it was small and green and thin.

And he thought, “Huh! It’s no going to be worth much...”

And he threw it away.

And he turned round...and just at the edge of the sea there was a city.

And it had walls and big gates and inside you could see the tops of the houses and you could see flags flying from some of them!

And Nils thought, “I must be dreaming”

So Nils turned round again and he looked back again and it was still there!

So he went over there, at the gates of the city, stood two guards. And Nils looked at them, and they didn’t say anything, so he went straight in.

The houses were beautiful! They had windows with little panes, you know the bottle glass, the wee circles in some of them. And they had painted walls and beautiful painted doors. And over the doors they had names and dates.

And it was wonderful!

And Nils wondered, “This is strange!”

And no-one seemed to be paying any attention to him, so he wandered everywhere.

And he ran past the place where there were armourers and they were beating out breast plates and making spears and swords. And there were people making the most wonderful jewellery with leaves on it, gold, silver, and jewels. And there were people weaving and they were weaving with silk and gold and silver!

And then Nils got into the market square and it was full of merchants. And when he stood still, all of the merchants were looking at him and they kept waving and saying “Come and buy this!!!” And they offered him beautiful cloth, and they offered him silver jugs, fruit, and the most wonderful fruit and vegetables.

And Nils sort of shrugged his shoulders. And then one of the merchants took out a small coin just like the one that Nil’s had dropped.

And the merchant said “Take anything you like, just give us one small coin.”

And Nils thought “Well I don’t want to be disrespectful...”. So he emptied out his pockets and there was nothing in his pockets at all.

And then he said “Just wait a minute, wait a minute, wait a minute!”

And he turned and he ran out of the square, he ran over the cobbles, past the the houses, past the people, past the guards. And he ran straight to where he went. He picked up the coin.

And he turned round...and the city had gone.

‘And just then, Nils remembered he he had to be for dawn. So he poked the stork and the stork woke up.

Nils said to the stork, “I knew that you brought me here for a special reason. I’m very sorry. Could you explain to me what was going on? Cos I dropped the coin...”

And he said, “Well, this was the city of Visby and it was full of proud, rich, selfish merchants. And a curse fell on the city and it was destined to go under the waves, but the people did not die. They just were there. And on Easter Eve, every hundred years, the city reappears from the water.

“Oh” said Nils. If only I had kept the coin.

“Never mind! It’s getting near dawn...we’d better go back to the mountain’.”

So, Nils got on the stork’s back and they flew past the fields, up the river, which got narrower and narrower and narrower, and the first glimmer of dawn was just showing

pink in the sky. And the stork said, “Goodbye, Nils!” And Nils was ready to go off on another adventure.

And that’s the end of the story!

## A.2 Hand-crafted Exemplar Event Calculus

```
<?xml version="1.0" encoding="UTF-8"?>
<EventStructure>
  <Event indicator="take" time="1">
    <Entity name="day" />
    <Entity name="easter" />
  </Event>
  <Event indicator="change" time="2">
    <Entity name="elf" />
    <Entity name="nil" />
  </Event>
  <Event indicator="fly" time="3">
    <Entity name="back" />
    <Entity name="nil" />
    <Entity name="geese" />
  </Event>
  <Event indicator="have" time="5">
    <Entity name="adventure" />
  </Event>
  <Event indicator="fly" time="10">
    <Entity name="nil" />
    <Entity name="geese" />
  </Event>
  <Event indicator="jump" time="12">
    <Entity name="nil" />
    <Entity name="stork" />
  </Event>
</EventStructure>
```

```
</Event>
<Event indicator="change" time="14">
  <Entity name="size" />
</Event>
<Event indicator="see" time="20">
  <Entity name="shape" />
  <Entity name="nil" />
</Event>
<Event indicator="be" time="31">
  <Entity name="stork" />
</Event>
<Event indicator="go" time="39">
  <Entity name="trip" />
</Event>
<Event indicator="shine" time="49">
  <Entity name="moon" />
</Event>
<Event indicator="jump" time="50">
  <Entity name="nil" />
  <Entity name="back" />
</Event>
<Event indicator="change" time="51">
  <Entity name="nil" />
  <Entity name="elf" />
</Event>
<Event indicator="have" time="66">
  <Entity name="stork" />
  <Entity name="rest" />
</Event>
<Event indicator="think" time="67">
  <Entity name="nil" />
```

```
</Event>
<Event indicator="explore" time="69">
  <Entity name="nil" />
</Event>
<Event indicator="walk" time="70">
  <Entity name="nil" />
</Event>
<Event indicator="find" time="71">
  <Entity name="coin" />
  <Entity name="nil" />
</Event>
<Event indicator="be" time="80">
  <Entity name="city" />
</Event>
<Event indicator="stand" time="91">
  <Entity name="guard" />
</Event>
<Event indicator="go" time="94">
  <Entity name="straight" />
  <Entity name="nil" />
</Event>
<Event indicator="make" time="110">
  <Entity name="jewellery" />
  <Entity name="people" />
</Event>
<Event indicator="don't" time="131">
  <Entity name="want" />
</Event>
<Event indicator="empty" time="133">
  <Entity name="nil" />
  <Entity name = "coin" />
```

```
</Event>
<Event indicator="turn" time="139">
  <Entity name="nil" />
</Event>
<Event indicator="run" time="142">
  <Entity name="nil" />
</Event>
<Event indicator="pick" time="144">
  <Entity name="nil" />
  <Entity name = "coin" />
</Event>
<Event indicator="disappear" time="146">
  <Entity name="city" />
</Event>
<Event indicator="wake" time="151">
  <Entity name="stork" />
  <Entity name="nil" />
</Event>
<Event indicator="say" time="152">
  <Entity name="stork" />
  <Entity name="nil" />
</Event>
<Event indicator="explain" time="157">
  <Entity name="nil" />
</Event>
<Event indicator="be" time="162">
  <Entity name="full" />
  <Entity name="merchant" />
  <Entity name="visby" />
</Event>
<Event indicator="fall" time="163">
```

```
<Entity name="city" />
  <Entity name="curse" />
</Event>
<Event indicator="go" time="165">
  <Entity name="wave" />
</Event>
<Event indicator="die" time="166">
  <Entity name="people" />
</Event>
<Event indicator="reappear" time="168">
  <Entity name="water" />
  <Entity name="city" />
  <Entity name="year" />
</Event>
<Event indicator="get" time="172">
  <Entity name="back" />
  <Entity name="nil" />
  <Entity name="dawn" />
</Event>
</EventStructure>
```

# Appendix B

## Sample Stories

This appendix contains both the original texts of sample stories from the corpus that were ranked on a rating scale (defined in Chapter 7) following the lead of former Scottish schoolteacher Senga Munro. Note that British school systems do not actually have a procedure for just grading the plot of a story. Instead, students are grouped into different groups based on their literacy, with the goal of the story-rewriting task to help them in gaining the writing abilities needed to pass to the next level. However, Senga agreed that plot was a vital factor and helped this project devise a scale that she believed reflected how she and most teachers would grade for plot. Each example is chosen because all three raters agreed that story belonged to that rank.

### **B.1 Excellent (Rank 1) Story**

Nils was a young boy who loved going on adventures. He went on different adventures any time he could. Nils used to turn himself into an elf so could fly on a geese's back. After he had a ride on the geese's back he turned back into a boy. He lay on the mountain thinking. Then he looked up at the moon and saw a shape. It had a long neck, big beak and very long legs. The stork came over to the mountain and said to Nils 'would you like a ride' and Nils replied 'yes' to the stork. So Nils turned into an elf and got on the stork's back. The stork took Nils to the island. When they got there Nils got off the stork and went to explore then in front of a big rock he stood on

something he bent down and picked it up. When he saw what it was he just threw it away. Then he saw a city he thought he was dreaming so he turned away. Then he looked back and saw it was still there so he went over to the city and went in the gates. He went in and looked around every one was saying come and buy this so he ran out of the gates to go and get the coin and he went and picked it up then the city was gone. Then he told the stork and the stork said it was a rich city but they were cursed to go under the waves of Sweden and every hundred years the city comes up.

## **B.2 Good (Rank 2) Story**

Once upon a time there was a boy his name was called Nils. One day Nils and goose went on an adventure at night they went to a building. And goose and Nils lay on the moss and Nils saw that it was a full moon. And said I'll not be able to sleep tonight. And then he saw something in the sky it had a long neck and long legs. And then it landed next to Nils. Nils got up on the Stork's back. It landed on the beach and when Nils got off stork's back stork collapsed. Nils walked along the beach and his foot got stuck on something and he picked it up it was an old coin. It was green and he chucked it away and he looked towards the water but it turned to a city. He rubbed his eyes and looked again it was still there. So he walked over and the guards let him through nobody cared until he went to the market and everybody said come to me he saw robes and gold necklaces and armour he looked at a man he said all I need is one coin only. So Nils ran out and to find the coin but when he tried to go back the city of Visby was gone for an hundred years he saw the stork wakening up he takes him back before dawn.

## **B.3 Fair (Rank 3) Story**

One day a little boy called Nils was looking out of the window. He was looking at the moon and saw a Goose next to the moon and it flew down and he went on adventure he went on a gooses' back and went over rivers, mountains, houses and fields. The goose stopped at a river. And Nils found a coin and he said it would not be worth much and

threw it away and hopped back on the geese's back and they got to a big city and they went in and there was people making things with silver, and gold and they said Buy every thing and give us one little coin and he checked his pocket so he ran back were he threw the coin and picked it up and he looked back and it wasn't there. So he left it because it was noon.

#### **B.4 Poor (Rank 4) Story**

One day there was a stork flying in the sky at night the wild geese went to the top of the hill where Nils was with the geese the moon was like the sun.

# Appendix C

## Sample Output from XML Pipeline

This appendix contains the transformation of a single story from raw text into the event-calculus as processed through the XML pipeline as described in Chapter 4 of the thesis. The story is an actual rewritten story, and the event calculus used is one of the event calculi used in the final evaluation of the system. The complete corpus of stories along every level of processing of the XML pipeline, including intermediate stages left out of this example, is available at [www.semanticstories.org](http://www.semanticstories.org).

### C.1 Raw Text

Once upon a time Nils turned his self in to and elf so he could go flying on the Geese's back. When he came back to the mountain. He turned his self back in to a boy this was on Easter eve. After He lay down on the rough grass watching the full moon when something struck his mind that he had to be back for dawn but he had plenty of time. Just about 5 minutes later he saw a dark black shadow that shone against the room. There was a large neck a huge beak and 2 magnificent wings. It came towards the hill and landed on one leg and it said 'I am the stork of Easter eve do you want to go on an adventure. So Nils jumped to his feet with excitement and said yes please I am up for an adventure. So Nils turned his self in to an elf and struggles to get on the stork's back so the stork lay down and Nils got on eventually. So the stork took off and they saw all the fields and Nils saw his house. The stork landed next to the bottom sea and

Nils was wondering why they had landed there and the stork said you will see Nils looked around and saw a green shiny coin that was thinner than a ruler. I threw it away and I turned round and there was a huge beautiful City and he didn't believe it so he walked in and there was beautiful houses then Nils walked in to the middle of the City and lots of people were making him buy all sorts of stuff but he didn't have any money so he ran as fast as he could to go and get the green coin and when he did the City had disappeared. So he jumped back on the stork and flew back to they mountain and turned back in to his human form. He asked the stork what it was and he said the City rises every one hundred days on the day before Easter.

## C.2 XML mark-up with Pronouns Resolved by GlenCova

```

<DOCS>
<TEXT>
<P>
<SENT>
<W P="RB" L="SL" T="w" S="Y" C="W">Once</W> <W P="IN" C="W">upon</W>
<W P="DT" C="W">a</W>
<W C="W" LM="time" P="NN">time</W><W C="W" LM="nil" P="NNP">Nils</W>
<W C="W" LM="turn" P="VBD">turned</W> <W P="NNP" C="W">Nils</W>
<W C="W" LM="self" P="NN">self</W>
<W P="IN" C="W">in</W> <W P="TO" C="W">to</W><W P="CC" C="W">and</W>
<W C="W" LM="elf" P="NN">elf</W>
<W P="IN" C="W">so</W><W P="NNP" C="W">Nils</W><W P="MD" C="W">could</W>
<W C="W" LM="go" P="VB">go</W><W C="W" LM="fly" P="VBG">flying</W>
<W P="IN" C="W">on</W>
<W P="DT" C="W">the</W><W C="W" LM="Goose" P="NNP">Geese</W>
<W P="POS" C="APOSS">'s</W>
<W C="W" LM="back" P="NN">back</W><W P="." T="." C=".">.</W>
</SENT>

```

```

<SENT>
<W P="WRB" L="SS" T="w" S="Y" C="W">When</W><W P="NNP" C="W">Nils</W>
<W C="W" LM="come" P="VBD">came</W> <W P="RP" C="W">back</W>
<W P="TO" C="W">to</W>
<W P="DT" C="W">the</W> <W C="W" LM="mountain" P="NN">mountain</W>
<W P="." T="." C=".">.</W>
</SENT>

```

```

<SENT>
<W P="NNP" L="SL" T="w" S="Y" C="W">Nils</W>
<W C="W" LM="turn" P="VBD">turned</W>
<W P="NNP" C="W">Nils</W><W C="W" LM="self" P="NN">self</W>
<W P="RB" C="W">back</W>
<W P="IN" C="W">in</W><W P="TO" C="W">to</W><W P="DT" C="W">a</W>
<W C="W" LM="boy" P="NN">boy</W><W P="RB" C="W">this</W>
<W C="W" LM="be" P="VBD">was</W>
<W P="IN" C="W">on</W><W C="W" LM="easter" P="NNP">Easter</W>
<W C="W" LM="eve" P="NN">eve</W>
<W P="." T="." C=".">.</W>
</SENT>

```

```

<SENT>
<W P="IN" L="SL" T="w" S="Y" C="W">After</W><W P="NNP" C="W">Nils</W>
<W C="W" LM="lay" P="VBD">lay</W><W P="RP" C="W">down</W>
<W P="IN" C="W">on</W><W P="DT" C="W">the</W><W P="JJ" C="W">rough</W>
<W C="W" LM="grass" P="NN">grass</W><W C="W" LM="watch" P="VBG">watching</W>
<W P="DT" C="W">the</W><W P="JJ" C="W">full</W>
<W C="W" LM="moon" P="NN">moon</W><W P="WRB" C="W">when</W>
<W C="W" LM="something" P="NN">something</W>
<W C="W" LM="strike" P="VBD">struck</W>
<W P="NNP" C="W">Nils</W><W C="W" LM="mind" P="NN">mind</W>

```

<W P="IN" C="W">that</W><W P="NNP" C="W">Nils</W>  
 <W C="W" LM="have" P="VBD">had</W>  
 <W P="TO" C="W">to</W><W C="W" LM="be" P="VB">be</W>  
 <W P="RB" C="W">back</W>  
 <W P="IN" C="W">for</W><W C="W" LM="dawn" P="NN">dawn</W>  
 <W P="CC" C="W">but</W>  
 <W P="NNP" C="W">Nils</W><W C="W" LM="have" P="VBD">had</W>  
 <W P="JJ" C="W">plenty</W>  
 <W P="IN" C="W">of</W><W C="W" LM="time" P="NN">time</W>  
 <W P="." T="." C=".">.</W>  
 </SENT>

<SENT>  
 <W P="RB" L="SL" T="w" S="Y" C="W">Just</W><W P="IN" C="W">about</W>  
 <W P="CD" C="CD">5</W><W C="W" LM="minute" P="NNS">minutes</W>  
 <W P="RB" C="W">later</W>  
 <W P="NNP" C="W">Nils</W><W C="W" LM="see" P="VBD">saw</W>  
 <W P="DT" C="W">a</W>  
 <W P="JJ" C="W">dark</W><W C="W" LM="black" P="NN">black</W>  
 <W C="W" LM="shadow" P="NN">shadow</W><W P="WDT" C="W">that</W>  
 <W C="W" LM="shine" P="VBD">shone</W><W P="IN" C="W">against</W>  
 <W P="DT" C="W">the</W><W C="W" LM="room" P="NN">room</W>  
 <W P="." T="." C=".">.</W>  
 </SENT>

<SENT>  
 <W P="EX" L="SS" T="w" S="Y" C="W">There</W><W C="W" LM="be" P="VBD">was</W>  
 <W P="DT" C="W">a</W><W P="JJ" C="W">large</W>  
 <W C="W" LM="neck" P="NN">neck</W>  
 <W P="DT" C="W">a</W><W P="JJ" C="W">huge</W>  
 <W C="W" LM="beak" P="NN">beak</W>

```

<W P="CC" C="W">and</W><W P="CD" C="CD">2</W>
<W P="JJ" C="W">magnificent</W>
<W C="W" LM="wing" P="NNS">>wings</W><W P="." T="." C=".">.</W>
</SENT>

```

```

<SENT>
<W P="PRP" L="SL" T="w" S="Y" C="W">It</W>
<W C="W" LM="come" P="VBD">came</W>
<W P="IN" C="W">towards</W><W P="DT" C="W">the</W>
<W C="W" LM="hill" P="NN">hill</W>
<W P="CC" C="W">and</W><W C="W" LM="land" P="VBD">landed</W>
<W P="IN" C="W">on</W>
<W P="CD" C="W">one</W><W C="W" LM="leg" P="NN">leg</W>
<W P="CC" C="W">and</W>
<W P="PRP" C="W">it</W><W C="W" LM="say" P="VBD">said</W>
<W P="'" C="LQUOTE">'</W>
<W P="NN" L="SL" T="w" S="Y" C="W">hill</W>
<W C="W" LM="be" P="VBP">am</W>
<W P="DT" C="W">the</W><W C="W" LM="stork" P="NN">stork</W>
<W P="IN" C="W">of</W>
<W C="W" LM="easter" P="NNP">Easter</W><W C="W" LM="eve" P="NN">eve</W>
<W C="W" LM="do" P="VBP">do</W><W P="NNP" C="W">Easter</W>
<W C="W" LM="want" P="VB">>want</W><W P="TO" C="W">to</W>
<W C="W" LM="go" P="VB">go</W>
<W P="IN" C="W">on</W><W P="DT" C="W">an</W>
<W C="W" LM="adventure" P="NN">adventure</W>
<W P="." T="." C=".">.</W>
</SENT>

```

```

<SENT>
<W P="CC" L="SL" T="w" S="Y" C="W">So</W><W C="W" LM="nil" P="NNP">Nils</W>

```

<W C="W" LM="jump" P="VBD">jumped</W><W P="TO" C="W">to</W>  
 <W P="NNP" C="W">Nils</W><W C="W" LM="foot" P="NNS">feet</W>  
 <W P="IN" C="W">with</W><W C="W" LM="excitement" P="NN">excitement</W>  
 <W P="CC" C="W">and</W><W C="W" LM="say" P="VBD">said</W>  
 <W P="RB" C="W">yes</W><W C="W" LM="please" P="VB">please</W>  
 <W P="NNP" C="W">Nils</W><W C="W" LM="be" P="VBP">am</W>  
 <W P="RB" C="W">up</W><W P="IN" C="W">for</W><W P="DT" C="W">an</W>  
 <W C="W" LM="adventure" P="NN">adventure</W><W P="." T="." C=".">.</W>  
 </SENT>

<SENT>

<W P="CC" L="SL" T="w" S="Y" C="W">So</W><W C="W" LM="nil" P="NNP">Nils</W>  
 <W C="W" LM="turn" P="VBD">turned</W><W P="NNP" C="W">Nils</W>  
 <W C="W" LM="self" P="NN">self</W><W P="IN" C="W">in</W>  
 <W P="TO" C="W">to</W><W P="DT" C="W">an</W><W C="W" LM="elf" P="NN">elf</W>  
 <W P="CC" C="W">and</W><W C="W" LM="struggle" P="NNS">struggles</W>  
 <W P="TO" C="W">to</W><W C="W" LM="get" P="VB">get</W><W P="IN" C="W">on</W>  
 <W P="DT" C="W">the</W><W C="W" LM="stork" P="NN">stork</W>  
 <W C="APOSS" LM="be" P="VBZ">'s</W><W P="RB" C="W">back</W>  
 <W P="RB" C="W">so</W><W P="DT" C="W">the</W>  
 <W C="W" LM="stork" P="NN">stork</W>  
 <W C="W" LM="lay" P="VBD">lay</W><W P="RP" C="W">down</W>  
 <W P="CC" C="W">and</W>  
 <W C="W" LM="nil" P="NNP">Nils</W><W C="W" LM="get" P="VBD">got</W>  
 <W P="IN" C="W">on</W><W P="RB" C="W">eventually</W>  
 <W P="." T="." C=".">.</W>  
 </SENT>

<SENT>

<W P="RB" L="SS" T="w" S="Y" C="W">So</W><W P="DT" C="W">the</W>  
 <W C="W" LM="stork" P="NN">stork</W><W C="W" LM="take" P="VBD">took</W>

<W P="RP" C="W">off</W><W P="CC" C="W">and</W>  
 <W P="NNS" C="W">struggles</W><W C="W" LM="see" P="VBD">saw</W>  
 <W P="PDT" C="W">all</W><W P="DT" C="W">the</W>  
 <W C="W" LM="field" P="NNS">fields</W><W P="CC" C="W">and</W>  
 <W C="W" LM="nil" P="NNP">Nils</W><W C="W" LM="see" P="VBD">saw</W>  
 <W P="NNS" C="W">struggles</W><W C="W" LM="house" P="NN">house</W>  
 <W P="." T="." C=".">.</W>  
 </SENT>

<SENT>  
 <W P="DT" L="SS" T="w" S="Y" C="W">The</W>  
 <W C="W" LM="stork" P="NN">stork</W>  
 <W C="W" LM="land" P="VBD">landed</W><W P="IN" C="W">next</W>  
 <W P="TO" C="W">to</W><W P="DT" C="W">the</W><W P="JJ" C="W">bottom</W>  
 <W C="W" LM="sea" P="NN">sea</W><W P="CC" C="W">and</W>  
 <W C="W" LM="nil" P="NNP">Nils</W><W C="W" LM="be" P="VBD">was</W>  
 <W C="W" LM="wonder" P="VBG">wondering</W><W P="WRB" C="W">why</W>  
 <W P="PRP" C="W">they</W><W C="W" LM="have" P="VBD">had</W>  
 <W C="W" LM="land" P="VBN">landed</W><W P="RB" C="W">there</W>  
 <W P="CC" C="W">and</W><W P="DT" C="W">the</W>  
 <W C="W" LM="stork" P="NN">stork</W><W C="W" LM="say" P="VBD">said</W>  
 <W P="PRP" C="W">you</W><W P="MD" C="W">will</W>  
 <W C="W" LM="see" P="VB">see</W><W C="W" LM="nil" P="NNP">Nils</W>  
 <W C="W" LM="look" P="VBD">looked</W><W P="IN" C="W">around</W>  
 <W P="CC" C="W">and</W><W C="W" LM="see" P="VBD">saw</W><W P="DT" C="W">a</W>  
 <W P="JJ" C="W">green</W><W P="JJ" C="W">shiny</W>  
 <W C="W" LM="coin" P="NN">coin</W><W P="WDT" C="W">that</W>  
 <W C="W" LM="be" P="VBD">was</W><W P="JJR" C="W">thinner</W>  
 <W P="IN" C="W">than</W><W P="DT" C="W">a</W>  
 <W C="W" LM="ruler" P="NN">ruler</W><W P="." T="." C=".">.</W>  
 </SENT>

<SENT>  
 <W P="NNP" L="SL" T="w" S="Y" C="W">Nils</W>  
 <W C="W" LM="throw" P="VBD">threw</W><W P="PRP" C="W">it</W>  
 <W P="RB" C="W">away</W><W P="CC" C="W">and</W>  
 <W P="NNP" C="W">Nils</W><W C="W" LM="turn" P="VBD">turned</W>  
 <W P="RB" C="W">round</W><W P="CC" C="W">and</W><W P="EX" C="W">there</W>  
 <W C="W" LM="be" P="VBD">was</W><W P="DT" C="W">a</W>  
 <W P="JJ" C="W">huge</W><W P="JJ" C="W">beautiful</W>  
 <W C="W" LM="city" P="NNP">City</W><W P="CC" C="W">and</W>  
 <W P="NNP" C="W">Nils</W><W C="W" LM="didn't" P="VBD">didn't</W>  
 <W C="W" LM="believe" P="VB">believe</W><W P="PRP" C="W">it</W>  
 <W P="IN" C="W">so</W><W P="NNP" C="W">Nils</W>  
 <W C="W" LM="walk" P="VBD">walked</W><W P="IN" C="W">in</W>  
 <W P="CC" C="W">and</W><W P="EX" C="W">there</W>  
 <W C="W" LM="be" P="VBD">was</W><W P="JJ" C="W">beautiful</W>  
 <W C="W" LM="house" P="NNS">houses</W><W P="JJ" C="W">then</W>  
 <W C="W" LM="nil" P="NNP">Nils</W><W C="W" LM="walk" P="VBD">walked</W>  
 <W P="IN" C="W">in</W><W P="TO" C="W">to</W><W P="DT" C="W">the</W>  
 <W C="W" LM="middle" P="NN">middle</W><W P="IN" C="W">of</W>  
 <W P="DT" C="W">the</W><W C="W" LM="city" P="NNP">City</W>  
 <W P="CC" C="W">and</W><W C="W" LM="lot" P="NNS">lots</W>  
 <W P="IN" C="W">of</W><W C="W" LM="people" P="NNS">people</W>  
 <W C="W" LM="be" P="VBD">were</W><W C="W" LM="make" P="VBG">making</W>  
 <W P="NNP" C="W">Nils</W><W C="W" LM="buy" P="VB">buy</W>  
 <W P="DT" C="W">all</W><W C="W" LM="sort" P="NNS">sorts</W>  
 <W P="IN" C="W">of</W><W C="W" LM="stuff" P="NN">stuff</W>  
 <W P="CC" C="W">but</W><W P="NNP" C="W">Nils</W>  
 <W C="W" LM="didn't" P="VBD">didn't</W><W C="W" LM="have" P="VB">have</W>  
 <W P="DT" C="W">any</W><W C="W" LM="money" P="NN">money</W>  
 <W P="IN" C="W">so</W><W P="NNP" C="W">Nils</W>

<W C="W" LM="run" P="VBD">ran</W>  
 <W P="RB" C="W">as</W><W P="JJ" C="W">fast</W><W P="IN" C="W">as</W>  
 <W P="NNP" C="W">Nils</W><W P="MD" C="W">could</W><W P="TO" C="W">to</W>  
 <W C="W" LM="go" P="VB">go</W><W P="CC" C="W">and</W>  
 <W C="W" LM="get" P="VB">get</W><W P="DT" C="W">the</W>  
 <W P="JJ" C="W">green</W><W C="W" LM="coin" P="NN">coin</W>  
 <W P="CC" C="W">and</W><W P="WRB" C="W">when</W>  
 <W P="NNP" C="W">Nils</W><W C="W" LM="do" P="VBD">did</W>  
 <W P="DT" C="W">the</W><W C="W" LM="city" P="NNP">City</W>  
 <W C="W" LM="have" P="VBD">had</W>  
 <W C="W" LM="disappear" P="VBN">disappeared</W>  
 <W P="." T="." C=".">.</W>  
 </SENT>

<SENT>

<W P="RB" L="SS" T="w" S="Y" C="W">So</W><W P="NNP" C="W">Nils</W>  
 <W C="W" LM="jump" P="VBD">jumped</W><W P="RP" C="W">back</W>  
 <W P="IN" C="W">on</W><W P="DT" C="W">the</W>  
 <W C="W" LM="stork" P="NN">stork</W><W P="CC" C="W">and</W>  
 <W C="W" LM="fly" P="VBD">flew</W><W P="RP" C="W">back</W>  
 <W P="TO" C="W">to</W><W P="PRP" C="W">they</W>  
 <W C="W" LM="stork" P="NN">stork</W><W P="CC" C="W">and</W>  
 <W C="W" LM="turn" P="VBD">turned</W>  
 <W P="RB" C="W">back</W><W P="IN" C="W">in</W>  
 <W P="TO" C="W">to</W><W P="PRP\$" C="W">his</W>  
 <W P="JJ" C="W">human</W><W C="W" LM="form" P="NN">form</W>  
 <W P="." T="." C=".">.</W>  
 </SENT>

<SENT>

<W P="NNP" L="SL" T="w" S="Y" C="W">Nils</W>

```

<W C="W" LM="ask" P="VBD">asked</W>
<W P="DT" C="W">the</W><W C="W" LM="stork" P="NN">stork</W>
<W P="WDT" C="W">what</W><W P="PRP" C="W">it</W>
<W C="W" LM="be" P="VBD">was</W><W P="CC" C="W">and</W>
<W P="NNP" C="W">Nils</W><W C="W" LM="say" P="VBD">said</W>
<W P="DT" C="W">the</W><W C="W" LM="city" P="NNP">City</W>
<W C="W" LM="rise" P="VBZ">rises</W><W P="DT" C="W">every</W>
<W P="CD" C="W">one</W><W P="CD" C="W">hundred</W>
<W C="W" LM="day" P="NNS">days</W><W P="IN" C="W">on</W>
<W P="DT" C="W">the</W><W C="W" LM="day" P="NN">day</W>
<W P="IN" C="W">before</W><W C="W" LM="easter" P="NNP">Easter</W>
<W P="." T="." C=".">.</W>
</SENT>
</P>
</TEXT>
</DOCS>

```

### C.3 Event-Calculus Representation

```

<?xml version="1.0" encoding="UTF-8"?>
<EventStructure>
  <Event indicator="go" time="2">
    <Entity name="elf" />
  </Event>
  <Event indicator="fly" time="3">
    <Entity name="back" />
  </Event>
  <Event indicator="come" time="4">
    <Entity name="nil" />
  </Event>
  <Event indicator="turn" time="5">

```

```
    <Entity name="nil" />
</Event>
<Event indicator="be" time="6">
    <Entity name="eve" />
</Event>
<Event indicator="lie" time="7">
    <Entity name="nil" />
</Event>
<Event indicator="watch" time="8">
    <Entity name="moon" />
</Event>
<Event indicator="strike" time="9">
    <Entity name="mind" />
</Event>
<Event indicator="have" time="10">
    <Entity name="nil" />
</Event>
<Event indicator="be" time="11">
    <Entity name="back" />
    <Entity name="dawn" />
</Event>
<Event indicator="have" time="12">
    <Entity name="plenty" />
    <Entity name="time" />
</Event>
<Event indicator="see" time="13">
    <Entity name="shadow" />
    <Entity name="nil" />
</Event>
<Event indicator="shine" time="14">
    <Entity name="room" />
```

```
</Event>
<Event indicator="be" time="15">
  <Entity name="neck" />
</Event>
<Event indicator="come" time="16">
  <Entity name="hill" />
</Event>
<Event indicator="land" time="17">
  <Entity name="leg" />
</Event>
<Event indicator="be" time="19">
  <Entity name="stork" />
  <Entity name="hill" />
</Event>
<Event indicator="do" time="20">
  <Entity name="eve" />
</Event>
<Event indicator="want" time="21">
  <Entity name="easter" />
</Event>
<Event indicator="go" time="22">
  <Entity name="adventure" />
</Event>
<Event indicator="jump" time="23">
  <Entity name="foot" />
  <Entity name="excitement" />
  <Entity name="nil" />
</Event>
<Event indicator="be" time="26">
  <Entity name="adventure" />
  <Entity name="nil" />
```

```
</Event>
<Event indicator="turn" time="27">
  <Entity name="nil" />
</Event>
<Event indicator="get" time="29">
  <Entity name="stork" />
</Event>
<Event indicator="lie" time="30">
  <Entity name="stork" />
</Event>
<Event indicator="get" time="31">
  <Entity name="eventually" />
  <Entity name="nil" />
</Event>
<Event indicator="take" time="32">
  <Entity name="stork" />
</Event>
<Event indicator="see" time="33">
  <Entity name="struggle" />
</Event>
<Event indicator="see" time="34">
  <Entity name="house" />
  <Entity name="field" />
</Event>
<Event indicator="land" time="35">
  <Entity name="sea" />
  <Entity name="stork" />
</Event>
<Event indicator="say" time="38">
  <Entity name="stork" />
</Event>
```

```
<Event indicator="see" time="39">
  <Entity name="you" />
</Event>
<Event indicator="look" time="40">
  <Entity name="nil" />
</Event>
<Event indicator="see" time="41">
  <Entity name="coin" />
</Event>
<Event indicator="thinner" time="42">
  <Entity name="ruler" />
</Event>
<Event indicator="throw" time="43">
  <Entity name="nil" />
</Event>
<Event indicator="turn" time="44">
  <Entity name="round" />
  <Entity name="nil" />
</Event>
<Event indicator="didn't" time="46">
  <Entity name="city" />
</Event>
<Event indicator="walk" time="50">
  <Entity name="middle" />
  <Entity name="house" />
</Event>
<Event indicator="buy" time="52">
  <Entity name="nil" />
</Event>
<Event indicator="didn't" time="53">
  <Entity name="stuff" />
```

```
</Event>
<Event indicator="run" time="55">
  <Entity name="money" />
</Event>
<Event indicator="get" time="58">
  <Entity name="coin" />
</Event>
<Event indicator="do" time="59">
  <Entity name="nil" />
</Event>
<Event indicator="disappear" time="60">
  <Entity name="city" />
</Event>
<Event indicator="jump" time="61">
  <Entity name="nil" />
</Event>
<Event indicator="turn" time="63">
  <Entity name="back" />
  <Entity name="form" />
</Event>
<Event indicator="ask" time="64">
  <Entity name="stork" />
  <Entity name="nil" />
</Event>
<Event indicator="say" time="66">
  <Entity name="nil" />
</Event>
<Event indicator="rise" time="67">
  <Entity name="day" />
  <Entity name="easter" />
  <Entity name="city" />
```

```
</Event>  
</EventStructure>
```

# Appendix D

## Statistical Tests of Reliability

There are a number of different statistics one may be used for measuring the reliability of a rating scale, and the world of computational linguistics seems yet to have come to any definite conclusions as regards standard statistics. What is perhaps more important than any statistic of reliability is that this statistic should be accompanied by a proper qualitative and quantitative exposition of the distribution that gives the reader a good quantitative grasp of the data as done in Chapter 7. Also, by making our data publicly available on [www.semanticstories.org](http://www.semanticstories.org), any interested party may do further statistical tests. However, there are a number of common statistics recommended by the statistics community for reliability, all of which are reported in Chapter 7 of this thesis and are explained below.

Some computational linguists might suggest the kappa statistic upon first glance upon the data (Carletta, 1996). The kappa statistic is defined as the actual agreement minus the agreement by chance divided by the one minus the agreement by chance. If the kappa statistic is actually calculated for the data, it is quite low:  $\kappa = .360$  between Rater *A* and Rater *B*, and  $\kappa = .637$  between Rater *B* and Rater *C*. However, appearances can be confusing. First, the kappa statistic should only be used for nominal data, and there are reservations about even that. The rating scale used for plot analysis is clearly at least an ordinal scale, although possibly some claims to an interval scale could be made as well. As Klaus Krippendorff<sup>1</sup> notes to Carletta, "I am suggesting to you that

---

<sup>1</sup>See the web-page: <http://www.cogsci.ed.ac.uk/jeanc/krippendorff.txt>

this (kappa) is an inappropriate statistic, at least in the domain of content analysis or the coding of qualitative data more generally, as it excludes unreliabilities due to unequal coder preferences for categories (manifest in unequal marginal frequencies of a contingency table)". Also, as noted by Uebersax<sup>2</sup>, "(Kappa) is not really a chance corrected measure...it is only suitable for testing whether agreement exceeds chance levels for binary and nominal ratings." This means it is not suitable for ordered categorical rankings like our own, since our rankings range between excellent (1) and poor, with excellent being the best and poor being the worst. Using arbitrarily chosen weights is the only way to even attempt to make  $\kappa$  useful on ordered data. Krippendorff continues, "in its (kappa's) denominator, kappa conforms to association and correlation statistics, e.g. chi square, while in its numerator it conforms to agreement statistics, e.g. Scott's pi. This makes kappa hybrid whose values are difficult to interpret." Alpha, unlike kappa, does take into account penalizing disagreements based on their distance, and also takes into account the differing preferences (base-rates) of the raters.

This is exactly why kappa performs so badly and alpha so well on our data. There *are* many differences between the ratings of the data, which explains why kappa does so poorly. However, such rankings are systematic between raters. Rater *C* is generally more lenient than Rater *B*, grading more stories as good and excellent, while Rater *A* is more harsh than Rater *B*, marking more stories as poor that Rater *B* would grade as fair. In our data there are very few (3) incidents where one rater differs from another rater by two ranks. Given the already mentioned systematic rater bias by one rank, alpha makes more sense than kappa. Although it varies quite from rater to rater, usually excellent stories are less common than good and fair stories. A story marked by one rater as good and another rater as a poor should be a much more greater cause for statistical alarm than a fair story being marked as a poor story. An excellent story being marked as poor story should call the whole scale into immediate question! Yet, this does not happen among human raters (although it does among machine-learners). The reason kappa does not work is because it assumes our ordered scale is just "labels", but they in fact contain more information about ranking than mere labels.

However, things become fairly murky. Our scale does not explicitly tell the raters

---

<sup>2</sup>On his web-page <http://ourworld.compuserve.com/homepages/jsuebersax/agree.htm>

whether it is an ordinal or interval scale. In fact, it appears that our scale would like to be at most an interval scale in best case, but is at least an ordinal scale. There are two main ways to test reliability for this type of scale. The first weaker assumption is to calculate correlation, seeing whether the values of one rater are high when the other rater is high. One accepted statistic to measure agreement is to observe when the paired observations vary together (concordances) and when they vary differently (discordances), which is measured by Kendall's  $\tau_b$  (tau-b). Kendall's tau-b is a non-parametric statistic that applies over non-normal distribution. Kendall's ( $W$ ) coefficient of concordance is, like Kendall's  $\tau_b$ , a measure of interrater agreement, but sensitive to variation in the scores of a rater because it is a normalized non-parametric equivalent to a two-way ANOVA. Between Rater  $A$  and Rater  $B$  the  $W$  is .194, and between Rater  $B$  and Rater  $C$  the  $W$  is .202. Both of these are surprisingly low, but remember that they just measure ranked concordances, which vary in two different ways depending on which two raters are being compared. However, the preferences are quite strong, and the rating scale is actually much more tightly constrained, so that just because "excellent" is less than "poor" and "fair" is less than "poor" does not mean that excellent stories and fair stories are equivalent. The more strong assumption is one based not just on concordance but on generalization based on distance between types of errors, so that an discordance between excellent and poor stories is penalized more than a discordance between excellent and fair stories, and that is penalized more than a discordance between excellent and good stories. By taking this into account, Cronbach's alpha determines the internal consistency of the rating scale and also takes into account the average correlation of items within the test. It takes into also account unequal rating preferences by raters. Lastly, although irrelevant to our present study, alpha is equivalent to Scott's  $\pi$  over nominal data with a large enough sample size, making it an elegant statistic that is easily interpretable over different studies. Kendall's tau-b is also a standard statistic for ordinal data.

Carletta has written a clear exposition of the kappa statistic (1996). A good introduction to the use of Cronbach's alpha is given in Krippendorff's book "Content Analysis" (1980). Finally, the SAS manual presents clear derivations of both Cronbach's alpha, Kendall's tau-b, and Kendall's coefficient of concordance.

# Bibliography

- Abney, S. (1995). Chunks and dependencies: Bringing processing evidence to bear on syntax. In Cole, J., Green, G., and Morgan, J., editors, *Computational Linguistics and the Foundations of Linguistic Theory*, pages 145–164.
- Allen, J. (1987). *Natural Language Understanding*. Menlo Park, CA, Benjamin/Cummings Publishing.
- Ando, R. K. and Lee, L. (2001). Iterative Residual Rescaling: An analysis and generalization of LSI. In *Proceedings of the 24th SIGIR*, pages 154–162.
- Applebee, A. (1978). *The child's concept of story*. University of Chicago Press, Chicago.
- Asher, N. and Lascarides, A. (2003). *Logics of Conversation*. Cambridge University Press.
- Bailey, P. (1999). Searching for storiness: Story-generation from a reader's perspective. In *AAAI Fall Symposium on Narrative Intelligence*.
- Baldwin, B. (1997). CogNIAC : A High Precision Pronoun Resolution Engine.
- Bancarz, I. and Osborne, M. (2002). Improved iterative scaling can yield multiple globally optimal models with radically differing performance level. In *Proceedings of COLING 2002*.
- Bartlett, F. (1932). *Remembering*. Cambridge University Press, Cambridge.
- Beaugrande, R. d. (1980). *Text, Discourse, and Process: Towards a Multidisciplinary Science of Texts*. Ablex, Norwood, NJ.
- Black, J. and Wilensky, R. (1979). An evaluation of story grammars. *Cognitive Science*, 3:213–229.
- Burstein, J., Marcu, D., and Knight, K. (2003). Finding the WRITE Stuff: Automatic Identification of Discourse Structure in Student Essays. *IEEE Intelligent Systems*, pages 32–39.

- Carletta, J. (1996). Assessing the agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–255.
- Charniak, E. (1972). *Toward a Model of Children's Story Comprehension*. PhD thesis, MIT.
- Charniak, E. (1999). A Maximum-Entropy Inspired Parser. Technical Report CS-99-12.
- Christian, D. and Young, M. (2001). Comparing Cognitive and Computational Models of Narrative Structure. Technical report, North Carolina State University.
- Colby, B. (1973). A Partial Grammar of Eskimo Folktales. *American Anthropologist*, (75):645–663.
- Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society For Information Science*, (41):391–407.
- Dennett, D. (1984). Cognitive Wheels: The Frame Problem of AI. In C. Hookway, editor, *Minds, Machines and Evolution*. Cambridge University Press, Cambridge, UK.
- Dreyfus, H. (1992). *What Computers Still Can't Do: A Critique of Artificial Reason*. MIT Press.
- Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.
- Foltz, P. (1996). Latent Semantic Analysis for Text-Based Research. *Behavior Research Methods, Instruments, and Computers*.
- Graesser, A. (2003). How does the mind construct and represent stories? In *Narrative Impact*. Erlbaum.
- Grover, C. and Lascarides, A. (2001). XML-based Data Preparation for Robust Deep Parsing. In *Meeting of the Association for Computational Linguistics*, pages 252–259.
- Grover, C., Matheson, C., Mikheev, A., and Moens, M. (2000). LT TTT - A Flexible Tokenisation Tool. In *Proceedings of the Second Language Resources and Evaluation Conference*.
- Hachey, C. G. B. and Korycinski, C. (2003). Summarising Legal Texts: Sentential Tense and Argumentative Roles. In *Proceedings of HLT-NAACL 2003 Workshop on Text Summarization*, Edmonton, Canada.

- Hickmann, M. (2003). *Children's Discourse: person, space and time across language*. Cambridge University Press, Cambridge, UK.
- Hobbs (1977). Resolving pronoun references. *Lingua*, (44):311–338.
- Kintsch, W. and van Dijk, T. A. (1978). Toward a model of text comprehension and production. *Psychological Review*, 85(5):363–394.
- Korb, K. B. (1998). The Frame Problem : An AI fairy tale. *Minds and Machines, Journal for Artificial Intelligence, Philosophy, and Cognitive Science*, 8(3).
- Krippendorff, K. (1980). *Content Analysis: An Introduction to Its Methodology*. Sage Publications, London.
- Labov, W. (1972). *Sociolinguistic patterns*. University of Pennsylvania Press, Philadelphia, PA.
- Lagerloff, S. (1907). *The Wonderful Adventures of Nils*. Doubleday, Page, and Company, Garden City, New York.
- Landauer, T. K. and Dumais, S. T. (1997). A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*.
- Levi-Strauss, C. (1963). *Structural Anthropology*. Basic Books, New York.
- Lin, D. and Pantel, P. (2001). DIRT: Discovery of inference rules from text. In *Knowledge Discovery and Data Mining*, pages 323–328.
- Luckin, R. and Boulay, B. D. (2001). How to we provide learners with effective help? In *Proceedings of AIED*.
- Mandler, J. and Johnson, N. (1977). Remembrance of Things Parsed: Story Structure and Recall. *Cognitive Psychology*, 9:111–151.
- Mandler, J. and Johnson, N. (1980). On throwing out the baby with the bathwater: A reply to Black and Wilensky's evaluation of story grammars. *Cognitive Science*, 4:305–312.
- Marcus, M., Kim, G., Marcinkiewicz, M., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. (1994). The Penn Treebank: Annotating predicate argument structure. In *ARPA Human Language Technology Workshop*.
- McCarthy, J. and Hayes, P. (1969). Some philosophical problems from the standpoint of Artificial Intelligence. In Meltzer, B. and Michie, D., editors, *Machine Intelligence*, volume 4.

- Meehan, J. (1976). The metanovel: Writing stories by computer. Technical report, Yale.
- Mikheev, A., Grover, C., and Moens, M. (1998). Description of the LTG system used for MUC. In *Seventh Message Understanding Conference: Proceedings of a Conference*.
- Minnen, G., J. C. and Pearce, D. (2001). Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–203.
- Minsky, M. (1975). A framework for representing knowledge. In Winston, P. H., editor, *The Psychology of Computer Vision*, pages 211–277. McGraw-Hill, New York.
- Mitkov, R. (1998). Robust Pronoun Resolution with Limited Knowledge. In *COLING-ACL*, pages 869–875.
- Moore, J. D. and Pollack, M. E. (1992). A problem for rst: The need for multi-level discourse analysis. *Computational Linguistics*, 18(4):537–544.
- Morgenstern, L. (1996). The Problem with Solutions to the Frame Problem. In Ford, K. M. and Pylyshyn, Z., editors, *The Robot's Dilemma Revisited: The Frame Problem in Artificial Intelligence*, pages 99–133. Ablex Publishing Co., Norwood, New Jersey.
- Mueller, E. T. (2003). Story understanding through multi-representation model construction. In Hirst, G. and Nirenburg, S., editors, *Text Meaning: Proceedings of the HLT-NAACL 2003 Workshop*, pages 46–53, East Stroudsburg, PA. Association for Computational Linguistics.
- Propp, V. (1968). *Morphology of the Folktale (translated by Laurence Scott)*. University of Austin of Texas.
- Riloff, E. (1999). Information extraction as a stepping stone toward story understanding. In Ram, A. and Moorman, K., editors, *Computational Models of Reading and Understanding*. MIT Press.
- Ritchie, G. (2001). Assessing creativity. In *Proceedings of AISB Symposium on AI and Creativity in Arts and Science*.
- Roberston, J. and Wiemar-Hastings, P. (2002). Feedback on children's stories via multiple interface agents. In *International Conference on Intelligent Tutoring Systems*, Biarritz, France.
- Rumelhart, D. (1975). Notes on a schema for stories. In Bobrow, D. and Collins, A., editors, *Representation and Understanding: Studies in Cognitive Science*. Academic Press, New York.

- Rumelhart, D. (1980). On evaluating story grammars. *Cognitive Science*, 4:313–316.
- Schank, R. C. and Abelson, R. P. (1977). *Scripts, Plans, Goals and Understanding*. Erlbaum, Hillsdale, NJ.
- Shanahan, M. (1997). *Solving the Frame Problem*. MIT Press, Cambridge, MA.
- Smith, B. C. (1985). Limits of Correctness in Computers. *SIGAS*, 14(4).
- Smith, B. C. (1996). *On the Origin of Objects*. MIT Press, Cambridge, MA.
- Steedman, M. (2000). The Productions of Time. Draft 4.1. Available from <http://www.informatics.ed.ac.uk/steedman/papers.html>.
- Stein, N. and Trabasso, T. (1982). What's in a story? an approach to comprehension. In Glaser, R., editor, *Advances in the psychology of instruction*, volume 2, pages 213–268. Erlbaum, Hillsdale, NJ.
- Teufel, S. and Moens, M. (1999). Discourse-level argumentation in scientific articles: human and automatic annotation. In *Towards Standards and Tools for Discourse Tagging. ACL 1999 Workshop*.
- Thompson, H., Tobin, R., McKelvie, D., and Brew, C. (1996). LT XML: Software API and toolkit for XML processing.
- Trabasso, T. and Sperry, L. (1985). Causal relatedness and the importance of story events. *Journal of Memory and Language*, 24:595–611.
- Wilensky, R. (1982). Points: A theory for the structure of stories in memory. In Lehnert, W. and Ringle, M., editors, *Strategies for Natural Language Processing*, pages 345–375, Hillsdale, N.J. Lawrence Erlbaum.
- Witten, I. H. and Frank, E. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.