Story Traces and Projections: Exploring the Patterns of Storytelling

Elizabeth Figa
School of Library and Information Sciences
University of North Texas
P.O. Box 311068
Denton, Texas 76203
E-mail: efiga@lis.admin.unt.edu
WWW: http://courses.unt.edu/efiga

Paul Tarau
Department of Computer Science
University of North Texas
P.O. Box 311366
Denton, Texas 76203
E-mail: tarau@cs.unt.edu
WWW: http://www.cs.unt.edu/∼tarau

Abstract. This paper describes a framework for building *story traces* (compact global views of a narrative) and *story projections* (selections of key story elements) and their applications in digital storytelling. Word and sense properties are extracted using the WordNet lexical database enhanced with Prolog inference rules and a number of lexical transformations. Inference rules are based on navigation in various WordNet relation chains (hypernyms, meronyms, entailment and causality links, etc.) and derived inferential closures expressed as boolean combinations of node and edge properties used to direct the navigation. The resulting *abstract story traces* provide a compact view of the underlying story's key content elements and a means for automated indexing and classification of story collections [1, 2]. *Ontology driven projections* act as a kind of "semantic lenses" and provide a means to select a subset of a story whose key sense elements are subsumed by a set of concepts, predicates and properties expressing the focus of interest of a user. Finally, we discuss applications of these techniques in story understanding, classification of digital story collections, story generation and story-related question answering. The main contribution of the paper consists in the use of a lexical knowledge base together with an advanced rule based inference mechanism for understanding stories, and the use of the information extracted by this process for various applications.

1 Introduction

We view stories as high-level natural language communication patterns with emergent properties beyond their underlying morphologic, syntactic and semantic building blocks – in a way similar to what expert chess players, mathematicians or programmers experience in their respective fields. In these types of mental processes, aggressive abstraction mechanisms need to be used to manage the complexity of the underlying ontology. This paper describes our experiments in extracting such abstract skeletons from stories. Our abstract skeletons are helpful in determining what a given story is *about* – as well as the *dynamics* of stories – the chains of temporal transitions stories describe. At an abstract level, stories share experiences in interpreting events and changes in time with focus on explanation and prediction. As such, they are improvised, ad-hoc *theories* trying to convey highly condensed forms of reusable experience. The skeletons we extract summarize this experience in the form of patterns and traces – usable for understanding, classifying and generating stories. Our skeletons – called abstract story traces – summarize a story as a sequence of abstractions derived from its key concepts, predicates and properties. In addition to the abstract story traces, we also need the ability to automatically focus on the actual story elements [3, 4] which are the most relevant with respect to a fixed ontology – a set of concepts, predicates and properties of interest to a given human or computer agent. We call these *"semantic lenses"* ontology driven projections. They provide a new form of dynamic computer assisted reading of a large corpora of narratives, like a library of stories, because they zoom in upon (and bring close) the objects of interest. This becomes important in interactive fiction – and in particular in game playing – as a mechanism to focus on fragments based on a user's interests.

2 Refactoring WordNet as a Lexical Knowledge Processor

The first step in building this functionality is quick access to the common sense knowledge provided by the WordNet database [5]. This database is also available in Prolog form (see `http://www.cogsci.princeton.edu/~wn`) and is therefore ready to be used as part of a rule-based inference system.

A lexicon consists of a set of word meanings and their semantic relationships. A systematic representation of the English lexicon based in psycholinguistic considerations has been put together in the database WordNet [6].

In a long-term collaborative effort that began in 1985, an interdisciplinary group at Princeton developed WordNet as a "machine readable lexical database organized by meanings" as WordNet defines itself. WordNet is not a dictionary – dictionaries help to settle issues of word use or sense priority – but do not address issues of synchronic organization of lexical knowledge as WordNet does [5]. WordNet divides the lexicon into five categories: *nouns, verbs, adjectives, adverbs* (which are connected by various semantic and lexical realtions), as well as *functions words* (which are not included, as they are assumed part of the syntactic component of the language).

Each of these lexical structures reflects a different way of categorizing experience. WordNet organizes lexical information in terms of word meanings rather than word forms in a way similar to a thesaurus.

WordNet maps word forms and word meanings as a many-to-many relation, which indicates that some forms have several different meanings, and some meanings can be expressed by different forms. Two difficult problems of lexicography, polysemy and synonymy, can be viewed as complementary aspects of this mapping. WordNet itself defines *polysemy* as the ambiguity of individual words or phrases that can be used in different contexts to express two or more different meanings and *synonymy* as the semantic relation that holds between two words that can (in a given context) express the same meaning. The common sense opposite of synonymy is antonymy. There are other semantic relationships (and their reverses) covered by WordNet: hypernymy/hyponymy and meronymy/holonomy. A hyponym is a meaning that acquires all the features of its hypernym, which is a more generic concept; for example, oak is a hyponym of tree. Meronymy is the relation of being part of; for example, arm is a meronym of body.

We have refactored the set of provided predicates to support constant time access to the set of *synsets* (equivalence classes of similar senses) associated to a given word phrase (indexed by a unique head word) and for the set of word phrases and relations associated to a given (unique) synset. While synsets are represented in the WordNet database as integers, for fast indexing purposes, they are in fact sets of synonynous words and word phrases collectively defining a sense element (concept, predicate or property). So, for example, the synset identifier `Id=100011413` maps to the following list of words and word phrases: `[[animal], [animate,being], [beast], [brute], [creature], [fauna]]`, which collectively define a synset.

2.1 WordNet as Two Many-to-Many Word-Sense Relations

The refactored WordNet Prolog database contains only two relations. The first Prolog predicate, `w(HeadWord,SynsetFrame)` maps to each HeadWord a list of words occurring after it, together with their associated sysnset lists. For example, a synset list on the term `woman` looks as follows

```
?- w(woman,SynsetFrame).
SynsetFrame=[
  ['''',s]-[
     [[body]]/[104467558],
     [[doctor]]/[108330802],
     [[hat]]/[103280816]
  ],
  [-,worship]-[
     []/[100795760]
  ],
  []-[
       [[chaser]]/[108828036],
       [[hater]]/[108464140],
       [[of],[the],[house]]/[108358979],
       [[of],[the],[street]]/[108590291],
        []/[106983459,108139544,108828291,108829560]
  ]
]
```

In the synset list above, the term "woman" is followed by a list of separate terms set in brackets. Note that these additional word phrases can select different, possible disjoint synsets. For instance, when the term "woman" is followed in the text, by "chaser" (thus becoming "woman chaser"), the resulting word phrase maps to a different list of synsets. The second relation `x(Synset,SenseFrame)` attaches to each unique synset a SenseFrame containing information collected from various WordNet predicates.

```
?- x(108358979,SenseFrame).
   SenseFrame=[
    hyp(108822648),
    ss(1,[[housewife]],n,1,5),
    ss(2,[[homemaker]],n,1,0),
    ss(3,[[lady],[of],[the],[house]],n,1,0),
    ss(4,[[woman],[of],[the],[house]],n,1,0),
    def([a,wife,who,who,manages,a,household,while,
         her,husband,earns,the,family,income])]

?- x(108828291,SenseFrame).
   SenseFrame=[
     ant(1,108436072,1),
     hyp(107901005),
     ss(1,[[woman]],n,1,480),
     ss(2,[[adult],[female]],n,1,0),
     r_hyp([107873135,....,108830376]),
     r_mp([104467558]),
     def([an,adult,female,person,'(',as,opposed,to,a,man,')']),
     ex([the,woman,kept,house,while,the,man,hunted])]
```

Note that pre-parsed *definitions* (def/1 slots) and *examples* (ex/1 slots) can be present in a SenseFrame together with a number of applicable relations (hypernyms, meronyms, antonyms, synonyms etc.) closely following the WordNet relation set (see `http://www.cogsci.princeton.edu/$\sim$wn/doc.shtml`).

This refactoring provides two sets of facts with the following properties:

– given a head word (and a possible few word context), we can extract in constant time all related information about their synsets (used in efficient parsing of textual data to synsnet lists)
– given a synset, we can extract in constant time all relations and related synsets or words provided by the WordNet database

Note also the presence of reversed relations like `r_hyp` (hyponyms=reverse hypernyms) and `r_mp` (reverse meronyms) which are precomputed to support high performace graph walk operations using BinProlog [7] and Jinni 2002's [8] built-in large database indexing mechanisms to provide constant time access to edges related to a given node for a given relation.

Overall, our refactoring simplifies WordNet while providing an efficient inference engine through Prolog rules that can digest the information contained in these basic relations. WordNet is extracted from a set of lexicographer files which put together a variety of different views on how meanings are organized. The level of precision it provides is statistical in nature. We will reduce some of this ambiguity by lifting to more general concepts following hypernym links (going "up" in the WordNet hierarchy via hypernyms, entailments and/or causality links) – or by selecting subtraces from a given ontology using hyponym and meronym links of subsumed concepts.

2.2 From Words and Phrases to Senses (Synsets)

WordNet provides a many-to-many relationship from words and word phrases to their independent senses called *synsets*. The first step after tokenizing and splitting a story in sentences is to compute the list of synsets associated to various words and word phrases. Although we extract maximal word phrases from a given story, the attached synset lists contain multiple synsets – as a result of lexical ambiguity.

```
?- tokens2synsets([the,story,goes,on],Is).
Is=[
 [the]-[],
 [story]-[102932667,105358470,105455815,105590213,105650568,106020196],
 [goes,on]-[200271654,200621853,201209119,201565591,202113765]
]
```

2.3 Using WordNet Definitions and Examples as a Training Set

As part of the refactoring process, we have parsed the WordNet definitions of synsets as well as their usage examples. They provide a fairly large corpora of natural language text usable for learning properties of sentences ranging from word frequencies to synset disambiguation hints [9, 10].

2.4 Using depth-K graph walks

In the directed graph G of synset nodes N_G we can see various WordNet relations as elements of a set of edges E_G. We implement our abstraction operators as depth-K walks in such graphs. Besides the primitive WordNet relations we use *closures*[1] as edge generators. Such closures are used, for instance, to generate edges leading to synsets which are hyponyms of a given set of synsets – to project a view of a story restricted to a given ontology.

[1] Our closures are predicate name+argument combinations, which receive two graph nodes as extra arguments to make-up a callable predicates.

3 Story Abstractions

The view of stories as *programs* usable to *replay* or understand chains of external events suggests a technique similar to Abstract Interpretation [11] which has been used to elegantly infer program properties in the field of Programming Languages. The technique consists of lifting properties from a *concrete domain* – in this case the lexical material of a story, seen as a sequence of word phrases – to an *abstract domain* obtained by following an abstraction operator and then propagating back the results of the (easier) analysis from the (smaller) abstract domain to the more complex concrete domain.

For instance, lifting from *rabbit* to its hypernym *animal* could enable an inference rule stating that *children like stories about animals* and project the result back as a rule that *children like stories about rabbits.*

3.1 Abstraction through the WordNet Hypernym Hierarchy

This is a simple but useful abstraction mechanism – with obvious applications to indexing. It consists of *lifting* words and word phrases extracted from the story transcripts to more general equivalents obtained by following upward links in the WordNet hypernym hierarchy – as in the following example:

```
?- super_of(wolf,2,S).
S=[
  [[carnivore]],[[wrongdoer],[offender]],
  [[libertine],[debauchee],[rounder]],
  [[man],[adult,male]],
  [[consume],[ingest],[take,in],[take],[have]],
  [[eat]]
]
```

This transformation allows, for instance, in a program trying to digest the *Little Red Cap (Little Red Riding Hood)* story to know more about the *wolf* character's actions and motivations. Note that we have not constrained the senses to only follow *noun* links, but the resulting listing of synsets (each represented on a different line in our example) actually provide meaningful links even if the syntactic category becomes *verb.*

3.2 K-level Noun Abstractions

Hypernym relations are more meaningful for nouns than for other syntactic categories. Noun-related synsets form relatively deep (up to 10-12) hierarchies coming from fairly reliable common sense and natural sciences classifications. By restricting a story trace to nouns, one can get an approximation of what the story is about – at different levels of abstractions.

3.3 Verb Abstractions

Pure verb traces (obtained by selecting only verb sequences) provide an abstract view of a story's dynamics. Like Web links, WordNet graphs exhibit *small-world* structure (strong clustering and small diameter). At a deep enough level (5-10), all stories will map to sequences like `[act] [transfer] [change] [rest]` and similar verbs, organized as collections of story-independent patterns. Such patterns indicate dramatic intensity and can be used to spot out climactic points in a story.

Causality/Entailment Verb Abstractions and Story Understanding WordNet provides a `cs(Cause,Effect)` and an `ent(Action,Consequence)` relation applying only to verbs. By using them in the context of a story, on can derive hints about what might happen next or explain why something has happened.

Answering *Why* Questions through Causality Abstractions Causal relations provide possible explanations, and as such, answers to *why* questions about a story or generate explanatory sentences usable in abstracts.

Following the Logic of a Story using Entailment Abstractions Entailment relations provide predictions for future states and can help reducing polysemy by selecting sysnets which fit in the entailment neighborhood of previously seen verbs, as the story advances. This is consistent with the fact that, to some extent, plots develop as possibilities are selected from the set of consequences of what happened. This allows following the logic of a story and possibly answering some "what-if" questions involving contrafactual developments and, in particular, questions about possible alternate endings of a story.

3.4 Customizing Abstraction Operators

Depending on the lexical category of a given synset, different abstraction operators can be used for fine tuning the results to what a human reader would consider relevant. Our customized abstraction operator follows the following algorithm, detailed per lexical category:

– **nouns**: hypernym links
– **verbs**: hypernym links, and hypernym links following causality and entailment links
– **adjectives**: attribute links to nouns followed by hypernym links and reverse attribute links back to adjectives
– **adjective satellites**: synonymy links to adjectives, followed by adjective abstractions as previously described
– **adverbs**: pertinence links to adjectives from which adverbs have been derived, followed by adjective abstractions and links back to adverbs

Abstraction operators have to deal with a fairly large set of possible meanings associated to each sentence. As we have not yet implemented an ambiguity reducing module using the global context of a story, the output contains a superset of what a human reader would pick as the most relevant skeleton describing a story at a given level of abstraction:

The toad kisses the beautiful girl and suddenly transforms into a prince.

```
level = 2

[toad] => [[craniate],[vertebrate]]
[kisses] => [[confection],[confectionery],[deed],[effort],[exploit],
             [feat],[sweet]]
[girl] => [[child],[female,person],[female],[human],[individual],[kid],
           [mortal],[person],[somebody],[someone],[soul]]
[prince] => [[leader]]

level = 5
[toad] => [[being],[organism]]
[kisses] => [[act],[food],[human,action],[human,activity],[nutrient]]
[girl] => [[animate,thing],[entity],[human],[individual],[living,thing],
           [mortal],[object],[person],[physical,object],[physical,thing],
           [somebody],[someone],[soul]]
[prince] => [[animate,thing],[entity],[living,thing],[physical,thing]]

level = 7
[toad] => [[object],[physical,object]]
[kisses] => [[entity],[physical,thing]]
[girl] => [[animate,thing],[entity],[living,thing],[physical,thing]]
[prince] => [[entity],[physical,thing]]
```

One can see that as abstraction levels increase, story traces become more and more generic – which makes them usable as a semantics driven grammar for recognizing or generating story elements.

4 Ontology Driven Semantic Projections

An ontology driven semantic projection specializes a text with respect to a given ontology. We represent an ontology as *a set of synsets and projection rules* describing how to collect sentences whose content can be derived from the synsets. This generalizes hyponym or meronym traces by allowing the user to fine-tune through a set of rules what kind of links will be followed for each concept or verb.

4.1 Concept Specific Abstract Semantic Traces

The previous traces have been driven by syntactic concept classifications. We will explore now the traces generated by restricting the focus of interest to a given concept's semantic neigborhood.

Hyponym Traces Given a concept C we can look at its hyponyms and their contexts occurring in a text. This specializes a text by only retaining concepts subsumed by a given concept.

Meronym Traces Given a concept C we can look at its meronyms and their contexts occurring in a text. This specializes a text by only retaining concepts which are "parts" of the entities refered to by a given concept.

4.2 Dominant Theme Projection

We obtain the *dominant theme* of a story by collecting the most frequent 1-level noun abstractions occuring in the story. This gives a good approximation of what the story "is about". We can now use the concept subsumption projection to pick the sentences where hyponyms of the dominating concepts occur.

5 Narratology and the Use of Story Traces and Projections

Professionals developing interactive digital stories or storytelling tools benefit from understanding what stories are and how they are interpreted by audiences so they can develop successful products. Walter Benjamin [12] distinguishes between the art of storytelling and the dissemination of information, raising questions about narrative ontology – what makes stories tellable as stories apart from what makes them tellable as information. Whether or not there are features that spell out this differentiation, stories are discursive models and what they display and how they display it is what engages and brings satisfaction to a *story audience*. The audience (story readers, interactors or users) perceives some type of value (or lack thereof) in a story, thus defining personal story aesthetics. These aesthetics are evaluated on differing criteria developed by the audience and inform the quality of the narrative experience. For the audience, there is a level of acceptance in the repetitive/ritualistic dimensions of stories, but the spatial, temporal, and ephemeral nature of stories makes the story-centered individual beg for more and new stories. One challenge for developers is to understand that stories, inherently, have narrative ambiguity and that successful narratives are not necessarily the ones that spell everything out in a perfect chain of events. Successful stories are written with just enough specification to allow the audience to become engaged by using their imagination and "making the story their own" through the dynamic process of interacating with the story and supplying connections and filling in the gaps of the storyline. This process of "filling out the story" is called the *audience narrativity* [13].

Stories are structured from their inception as a basic skeleton (or storyline) that is further developed by the addition of a narrative world, a plot, characters, action, resolution toward an ending, etc. Stories adopt rules, and even if the rules are broken, a story is only a story if it is still tellable and meaning-making in context. Rules of tellability define minimums of acceptability for a narrative to retain story properties allowing it to also be sense-making and coherent. While stories may not have perfect integrity, there is an obligation to make some parts understood and to bring the story to some form of closure.

Story tracing techniques can help a developer understand the basic core of a story under development because it gives a compact view of story elements filtered at a given level of abstraction, while retaining the sequence of events. These tracing results can also be compared with traces of other stories as a way to focus on similar or different concept development and goals for the project.

The use of ontology driven projections with a text serves as an advanced form of information retrieval to find sections of a story with particular themes or ideas. These "semantic lenses" can search by inference rules for concepts or themes that typical word searches or strings of terms might otherwise miss. They also project what is imbedded conceptually or thematically in a narrative so that that a developer can confirm what is in the story or reveal what might not be known to be imbedded in a story.

Both story traces and projections provide deeper insight into what a story "is about". As well, they have potential to reveal both the core elements and broad concepts/themes within stories.

Storytellers, movie makers, interactive fiction writers, and virtual reality game developers need better tools to help them draw upon sound narrative principles to find and develop tellable stories that will work with their audiences. New tools to understand and compare narrative content will enhance the development of stories containing the concepts, predicates and properties expressing the focus of interest of a potential audience group.

6 Applications

We plan to use our abstract story traces to build a library of *story patterns*, by extracting sentence patterns and then organizing them into intersentential abstraction chains – patterns that give a synthetic view of the whole story. A key application of this technique is in story generation, ranging from story variants to alternate ending to new stories based on a set of patterns. Patterns will be also used for answering questions about a story and to automatically generate story indexes and a story database. We plan to achieve this by using manually prepared story indexes as a training set and then extracting new indexes based on statistical properties of story traces. The technique will also provide story classification combined with clustering of similar traces. By restricting our focus to *linear stories* (sequences of natural language sentences about chronologically ordered events) and using them as a training set to extract causal and logical entailment rules, we plan to obtain *story line predictors* usable, for instance to generate new stories or alternative endings of existing stories.

7 Conclusion

The operations described in the paper use a psycholinguistics inspired lexical database, WordNet, and a number of derived relations and inference mechanisms to extract high-level information about stories as a compact view of their key content elements. Story projections provide selections of key story elements which work as a kind of "semantic lenses" focusing on sentences related to a given ontology. These techniques have applications in story understanding, classification of digital story collections, story generation, and story-related question answering and are likely to be extensible to various other types of narratives.

References

1. A. Aarne. *The Types of Folktales A Classification and Bibliography. Translated and revised by Stith Thompson.* Helsinki: Academia Scientiraum Fennica, 1961.
2. D. Ashliman. *A Guide to Folktales in the English Language: Based on the Aarne-Thompson Classification System.* Westport, CT: Greenwood Press, 1987.
3. S. Thompson. *Motif-Index of Folk Literature: A Classification of Narrative Elements in Folktales, Ballads, Myths, Fables, Medieval Romance, Exampla, Fabliaux, Jest-Books, and Local Legends. 2 vols.* Westport, CT: Greenwood Press, 1956.
4. M. MacDonald. *Storytellers Sourcebook: A Subject, Title, and Motif Index to Folklore Collections for Children, 1983-1999.*
5. George Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. Five papers on WordNet. CSL Report 43, Cognitive Science Laboratory, Princeton University, July 1990.
6. C. Felbaum. *Wordnet, an Electronic Lexical Database for English.* Cambridge: MIT Press, 1998.
7. Paul Tarau. BinProlog 9.x Professional Edition: Advanced BinProlog Programming and Extensions Guide. Technical report, BinNet Corp., 2002. Available from http://www.binnetcorp.com/BinProlog.
8. Paul Tarau. Inference and Computation Mobility with Jinni. In K.R. Apt, V.W. Marek, and M. Truszczynski, editors, *The Logic Programming Paradigm: a 25 Year Perspective*, pages 33–48. Springer, 1999. ISBN 3-540-65463-1.
9. X. Li, S. Szpakowicz, and S. Matwin. A wordnet-based algorithm for word sense disambiguation. In *Proc. of the 14th IJCAI*, pages 1368–1374, Montreal, Canada, 1995.
10. Rada Mihalcea and Dan I. Moldovan. A highly accurate bootstrapping algorithm for word sense disambiguation. *International Journal on Artificial Intelligence Tools*, 10(1-2):5–21, 2001.
11. P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *4th ACM Symp. Principles of Programming Languages*, pages 238–278, 1977.
12. Walter Benjamin. *The Storyteller.* New York: Schocken, 1969.
13. Thomas Leitch. *What Stories Are: Narrative Theory and Interpretation.* University Park, PA, The Pennsylvania State University Press, 1986.