Story Retrieval and Comparison Using Concept Patterns

by

Caryn E. Krakauer

S.B., C.S. M.I.T., 2011

Submitted to the Department of Electrical Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the Massachusetts Institute of Technology

May 2012 [June 2012]

Author:
Department of Electrical Engineering and Computer Science
May 21, 2012

Certified by:
Professor Patrick H. Winston Thesis Supervisor
May 17, 2012

Accepted by:
Professor Dennis M. Freeman, Chairman, Masters of Engineering Thesis Committee

Story Retrieval and Comparison Using Concept Patterns

by

Caryn E. Krakauer

Submitted to the
Department of Electrical Engineering and Computer Science

May ?, 2012

In Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

To understand a new situation, humans draw from their knowledge of past experiences and events. For a computer to use the same method, it must be able to retrieve stories that shed light on a new situation. Traditional story retrieval uses keywords to determine similarity. Keywords are useful for determining whether stories share similar topics. However, they miss how stories can be structurally similar. In my work, I have used high level *concept patterns*, which are structures of causally related events. Concept patterns follow the Goldilocks principle, that the features should be of intermediate size. Given a story about cyber crime and another about traditional warfare, the wording will be different, as cyber crime involves viruses, DDOS attacks, and hacking, while traditional warfare involves armies, invasions, and weapons. However, both stories may involve instances of *revenge* and *betrayal*. Using a corpus of 15 conflict stories, I have shown that a similarity measure based on concept patterns differs substantially from a similarity measured based on keywords. In addition, I compared three concept-pattern methods with human performance in a pilot study in which 11 participants performed story comparison. My goal was to contribute to a human competence model, but I have also explored applications in story retrieval, prediction, explanation, and grouping.

Acknowledgments

Research on Genesis has been supported, in part, by the National Science Foundation (IIS-0413206), the Office of Naval Research (N00014-09-1-0597), the Air Force Office of Scientific Research (A9550-05-1-0321), and the Defense Advanced Research Projects Agency (FA8750-10-1-0076).

Professor Patrick Winston for acting as my supervisor and for letting me work on exactly what I wanted to.

My family for supporting me in all my ambitions and endeavors.

Table of Contents

Acknowledgments....3
1 Introduction....7
1.1 My Start with Artificial Intelligence....7
1.2 Genesis and the Similarity Module....8
2 Background....10
2.1.1 Ullman's Work with Intermediate Features in Facial Recognition....10
2.1.2 Expert vs. Novice in Domain Retrieval....11
2.1.3 Intermediate Feature in the Similarity Module....11
2.2 The Genesis System....12
2.2.1 What is the Genesis System?....12
2.2.2 Elaboration graph....13
2.2.3 A Modular System....13
2.2.4 Sequences and Events....14
2.3 The Conflict Story Set....16
3 Concept Patterns....19
3.1 User-Defined Concept Patterns....21
3.1.1 Advantages of User-Defined Concept Patterns....22
3.1.2 Disadvantages of User-Defined Concept Patterns....23
3.2 Genesis Generated Concept Patterns....23
3.2.1 Why generate concept patterns?....23
3.2.2 Examples of Genesis Generated Concept Patterns....24
3.2.3 Concept Pattern Discovery....25
3.2.4 Concept Pattern Filtering....26
3.2.5 Comparing Generated Concept Patterns....26
3.2.5.1 Finding an Alignment....27
3.2.5.2 Comparing Structure....28
3.2.5.3 Comparing Word Similarity....29
3.3 Methods of Comparison....32
3.3.1 Vector Angle....32
3.3.2 In-Order Comparison....32
3.3.3 Rarity....34
3.3.4 Comparing stories with generated concept patterns....36
4 Results....38
4.1.1 Keyword analysis....38
4.1.2 User-defined concept patterns....38
4.1.2.1 Vector Dot Product Comparison....39
4.1.2.2 In-Order Comparison....41
4.1.3 Genesis generated concept patterns....43
5 Comparing Methods to Human Study....45
5.1 Study....45
5.1.1 Keyword analysis....46
5.1.2 User-defined concept patterns....46

5.1.2.1 Vector Dot Product Comparison....46
5.1.2.2 In-Order Comparison....46
5.1.3 Genesis generated concept patterns....46
6 Conclusion....48
6.1 Discussion....48
6.2 Applications and Future Work....48
6.2.1 Expert Study....48
6.2.2 Precedent-Based Reasoning....49
6.2.3 Predictions....49
6.2.4 Grouping....51
6.3 Contributions....52
6.3.1 Similarity Module....52
6.3.2 Conflict Stories....54
6.3.3 Human Study....54
Bibliography....55
Appendix A: Conflict Stories....56
American Revolution....56
Afghanistan Civil War....57
American Civil War....58
Cambodia-Vietnam Invasion....59
Chad-Libyan War....60
China Border War with India....61
China Border War with USSR....62
China Invasion of Tibet....63
China War with Vietnam....64
Congo Civil Conflict....65
Cuba – Bay of Pigs Invasion....66
Czechoslovakia Soviet Invasion....67
Nigerian Civil War....68
Persian Gulf War....69
Romania and Ceausescu....70

List of Figures

1. Different Sizes of Facial Features....11
2. The Elaboration Graph of the "American Revolution" Story as Created by Genesis.....15
3. Examples of Events....16
4. Example Conflict Story....18
5. Possible Structure of Concept Patterns....21
6. Basic Structure of a User-Defined Concept Pattern....22
7. Examples of User Defined Concept Patterns....23
8. Examples of Genesis Generated Concept Patterns....25
9. The Concept Pattern Discovery Algorithm....26
10. Two Concept Patterns Being Compared....28
11. Concept Patterns with Different Structures....29
12. Pseudocode for the Structure Matching Algorithm....30
13. Examples of Word Threads from WordNet....31
14. Samples Structures of Concept Patterns....32
15. An Example of In-Order Concept Pattern Matching....34
16. In-Order Matching of the "China War with Vietnam" and the "Congo Civil Conflict". 35
17. Keyword Analysis Results. 40
18. User-Defined Concept Pattern Results Using the Vector Dot Product....41
19. In-Order Comparison of "American Revolution" and "American Civil War"....43
20. Results for In-Order Comparison of User-Defined Concept Patterns....44
21. Results from Genesis Generated Concept Patterns Using the Vector Dot Product....45
22. Human Testing Results....46
23. Story Prediction....51
24. Comparison of the "American Revolution" and the "American Civil War" for Prediction 51

List of Tables

1. Stories Used....16
2. Concept Pattern Rarity....36
3. The mean and standard deviation of similarity scores generated by each method....39

1 Introduction

1.1 My Start with Artificial Intelligence

For almost my entire life, I have been interested in intelligent machines. When I was young, I watched *Star Trek: The Next Generation*, and was fascinated by the character of the android Data. In middle school, I began reading Isaac Asimov, the man who created the word *robotics* and who features intelligent machines in many of his works. When I first applied to MIT five years ago, I wrote my college essay on how I wanted to build systems that were capable of reasoning.

The problem of creating a truly intelligent machine is mind boggling. Most modern day artificial intelligence systems are based on statistics and large quantities of data. While these systems are very successful, they lack a true understanding of concepts involved. How can you make a computer understand what a dog is? Or that water is wet? Or even basic emotions? To understand even simple situations, a computer will need basic knowledge, common sense, visual knowledge, basic reasoning, and many more systems. I focused my thesis research on precedent-based reasoning.

1.2 Genesis and the Similarity Module

The Genesis system, created by Professor Patrick Winston and the Genesis group, reasons about stories in a human-like manner. Given a story in plain English, Genesis is able to analyze it, applying common sense reasoning as well as reflection level analysis. It is able to determine the cause and effects of events in a story based on common sense reasoning. Genesis is also able to answer questions about the stories it reads, such as "Why did person A harm person B?" or "Why did A want to defeat B?". The goal of the Genesis system is to reason about stories and situations like a human does.

A large part of human reasoning is the ability to recall similar situations and apply them to the current event. The use of precedents is very important in understanding stories, as a person's history will affect how they perceive the world. A pacifist may see a violent act as insanity, while a more violence-accepting person may see a violent act as necessary. Each person's personal history and precedents shape their understanding of the current story. A person is able to easily recall similar stories, and apply them to the current situation.

In order for a program to engage in precedent-based reasoning, a system for finding relevant precedents must be developed. In this project, I have written a Similarity Module for the Genesis system. The Similarity Module is capable of recalling similar stories from memory when given a new story.

The basis of the Similarity Module is the *concept pattern*. Concept patterns are groups of causally connected events. A concept pattern encodes both the events and their relations to each other. An example of a concept pattern is a *revenge*. A *revenge* is defined as: *person A harms person B leads to person B harms person A*. The *revenge* concept pattern is made of two events, where the first event causes the second. The concept pattern is based on the idea of an *intermediate feature*, which follows the Goldilocks Principle. The Goldilocks principle states that a feature should not be too small and not be too large (Finlayson and Winston, 2006). The idea of an intermediate feature has been shown in other studies to be a powerful tool for retrieval. A concept pattern is an ideal intermediate feature, as it encodes themes and patterns often found in stories, but is not so large that it becomes too specific to match any story.

An advantage of concept patterns as opposed to keyword comparison is that the underlying structures can be compared regardless of story topic. For example, take the classic Shakespearean play "The Taming of the Shrew" and the modern movie "10 Things I Hate About You". "10 Things I Hate About You" is widely known to be a modern retelling of the Shakespearean classic, as it shares many concept patterns. However, a traditional keyword comparison would be unable to find the similarity in the stories. "The Taming of the Shrew" takes place in 16th century Italy, while "10 Things I Hate About You" takes place in modern day California. The modern film contains skateboards, college applications, and sports practice. None of these things are seen the 16th century Italy. However, by looking at the larger structural elements, the similarities can be seen. In both, the younger sister is more desirable, and thus has three suitors. In order for her to be dated/married, the older sister must be wooed. In order to facilitate this, the suitors hire a man to woo the older sister. The older sister is eventually wooed at the end. By looking at the plot and not the individual elements, the similarities of the stories are easily seen.

The Similarity Module has been tested with 15 conflict stories. Each story is about a historical military or political battle between entities.

2 Background

2.1.1 Ullman's Work with Intermediate Features in Facial Recognition

The use of intermediate features in this project was inspired by Shimon Ullman's work on recognizing faces in images (Ullman et al., 2002). Ullman's program used facial features to aid in retrieval. Given an image, the program would attempt to match the feature to every location in the image. If a match was found between a location in the image and the facial feature, then the program determined that the image contained a face.

The facial features used ranged from a single eye to an entire face. Ullman found that intermediate features, such as a nose and mouth combination, worked better at retrieval than a small feature such as an eye or a very large feature such as an entire face. The intermediate features that worked best were combinations of overlapping features.

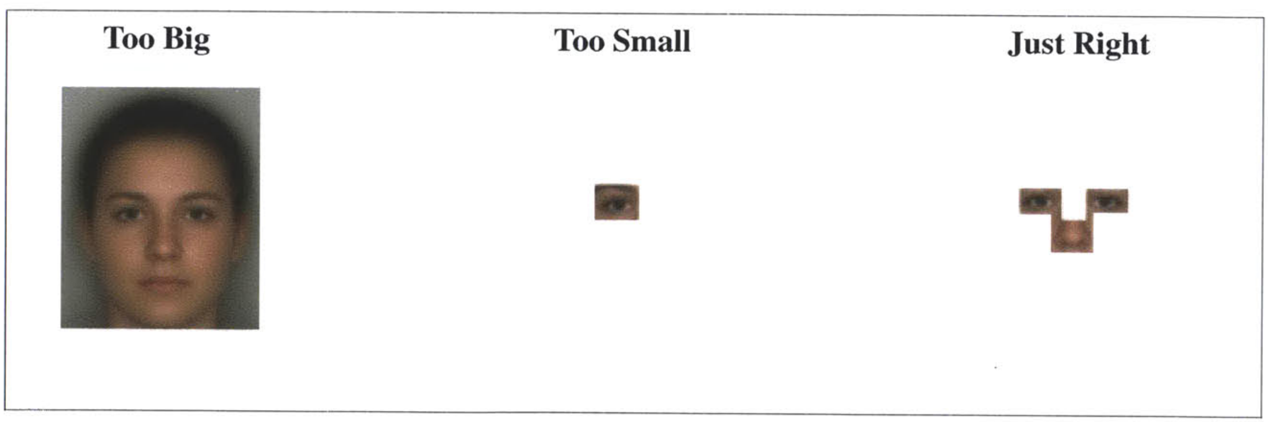


Figure 1: **Different Sizes of Facial Features.** *Ullman's facial recognition software used intermediate features, such as eyes paired with a nose for the most powerful retrieval. Features that were too small, such as an eye produced too many false positives. Features that were too large such as an entire face did not find matches.*

A feature that was too small would have too many false positives. An eye for example, would match two concentric circles. A feature that was too large would match nothing, as it was too specific. By using intermediate features, such as eyes and a nose, the program was able to achieve a high success rate.

2.1.2 Expert vs. Novice in Domain Retrieval

Intermediate features, such as *concept patterns*, are more likely to be used by an expert in a domain than a novice. Psychological research, reviewed in Finlayson and Winston (Finlayson and Winston, 2005), shows that experts and novices in a domain retrieve stories using various features. An expert will retrieve using structures, while a novice will retrieve using superficial features. Going along the spectrum of novice to expert corresponds to an increase in the maximum chunk size used by the matcher. There is also an increase in complexity, going from an individual object to a structure of related objects. Experts match based on features of intermediate size, following the Goldilocks principle. In the conflict stories used in this project, a novice may retrieve using features such as countries involved, while an expert would retrieve using larger themes, such as a vacuum of power created by a super power.

I have developed a program capable of comparing stories based on the underlying structure rather than the superficial features.

2.1.3 Intermediate Feature in the Similarity Module

The Similarity Module uses concept patterns as intermediate features used for story retrieval. In story retrieval a small feature would be a keyword found in the story, or a single event. A large feature would be the majority of the structure of a story. The ideal size found for a concept pattern was around 2 to 3 events connected, with most stories containing around 10-15 events. An event or keyword would match too many stories or wrong stories, while a large structure would match none at all. Because of this, concept patterns are well suited for story retrieval.

2.2 The Genesis System

All work done has been integrated into the Genesis system, a system that is being developed by Professor Patrick Winston and the Genesis Group. The system is written in Java.

2.2.1 What is the Genesis System?

The purpose of Genesis is to apply human-like reasoning to the world around it by using language, narrative, and vision. The Genesis group believes that story understanding is the key to intelligence, and by using written and visual stories, a system can be developed that can reason about events and situations like a human. Genesis is designed to answer questions about its own thought process, and is based off a human computational model rather than a statistical model.

Genesis is a highly modular system. It is made up of modules which perform various tasks, such as parsing, doing comparisons, finding concept patterns, and finding relations between events. Each module is wired to other modules, and communicates with them via *signals*. Signals are messages such as a sequence of events or concept patterns. A module takes in the signals its inputs are wired to, and send signals to any module wired to its output. Modules can be swapped out easily for improved versions without affecting the rest of Genesis. For example, as the Genesis parser increases in power, other modules can take advantage of its ability to read more complex stories. In addition, parallel work can be done on separate modules. The modular structure of Genesis allows for highly paralleled work as well as improvement with minimal upkeep.

2.2.2 Elaboration graph

The Genesis system produces an *elaboration graph* of stories it reads. An elaboration graph is a representation of a story. It encapsulates the events, predictions, explanations, and the relations between each part of a story. The graph connects events that causally lead to each other. So if *A harms B because B harms A*, then the two harm events will be connected as one caused the other. However, if a *harm* events comes after another *harm* event, but does not cause it, then they will not be connected. Only events that are connected to other events appear in the elaboration graph. Concept patterns are made from the events in the elaboration graph. Figure 2 shows an elaboration graph from the story "American Revolution" as created by Genesis.

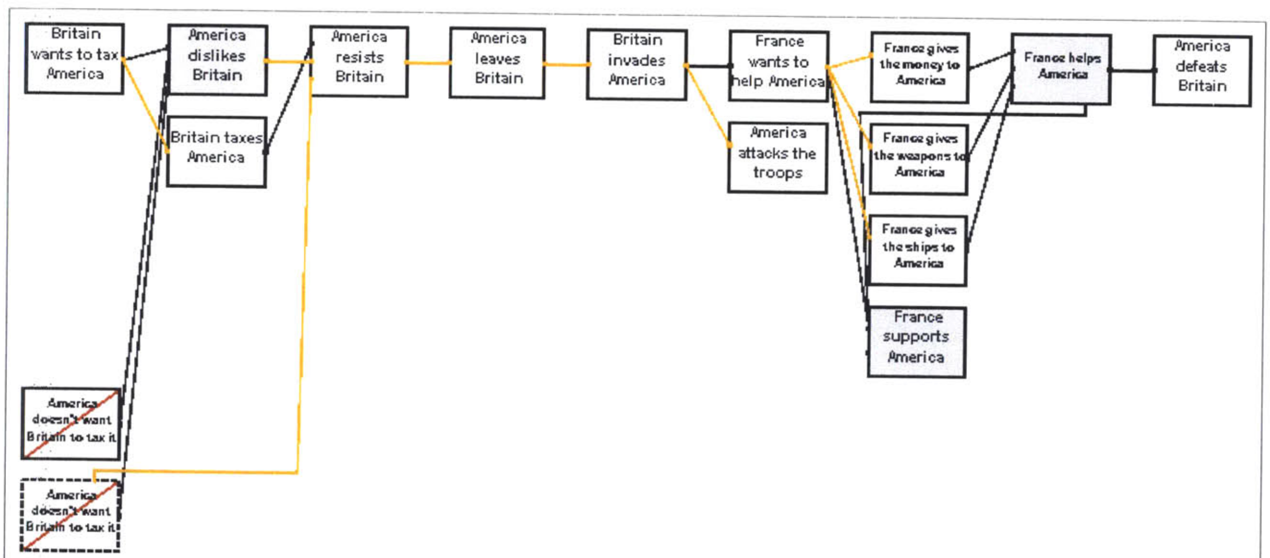


Figure 2: The Elaboration Graph of the "American Revolution" Story as Created by Genesis. *Events are connected by causal relations.*

2.2.3 A Modular System

Genesis works as a fully modular system with multiple processors. The Similarity

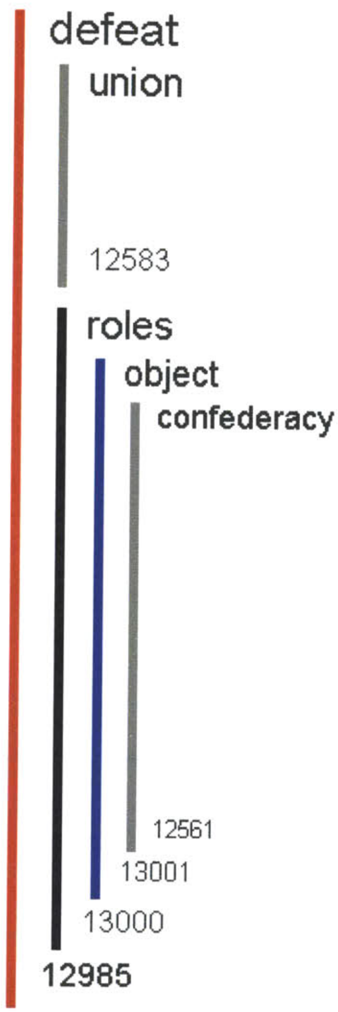
Module I have created works as one module in the Genesis system. Modules are "wired" together, where they receive input signals and send output signals.

Currently, the stories used for story recall with concept patterns are relatively simple in both plot and grammar. However, concept patterns do not depend on the simplicity of the stories. Once a more complex parser is introduced, the concept pattern module will be able to work with more complex stories with minimal upkeep.

2.2.4 Sequences and Events

Stories enter the Similarity Module in the form of a sequence of events. An event contains a single action, although the action itself can be complex. In figure 3, there are two examples of events of different complexities. Each event has been parsed and put into a form that makes it easy to traverse and analyze. The structure of an event encodes the relations between the words in the event, including the subject, object, action, and modifiers. Genesis also generates visuals that represent the structure of events.

The Union defeated the Confederacy.	The Union forced the Confederacy to return to the United States.
(t union-12583) (s roles (d object (t confederacy-12561))))	(r coerce (t union-12583) (s roles (d object (r return (t confederacy-12561) (s roles (d object (s path (d to (d at (t united_states-12537))))))))))



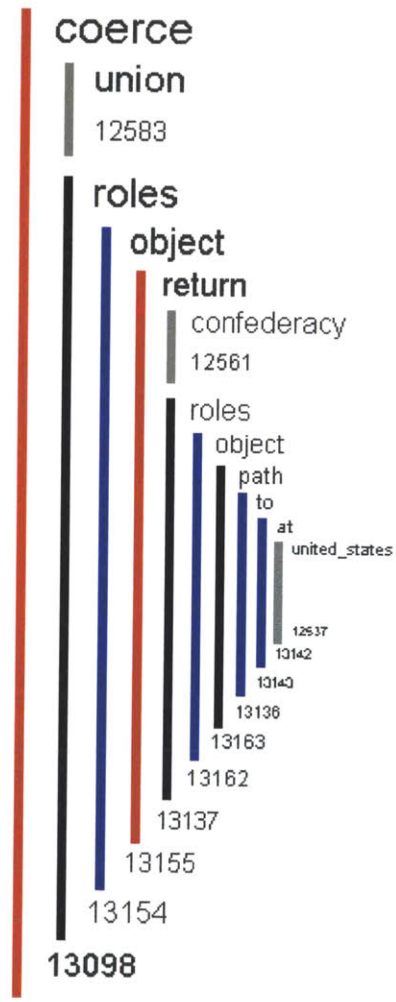


Figure 3: Examples of Events. *Two examples of events from the "American Civil War" story. The first form is their standard English form, the second form is their internal representation used by Genesis, and the third form is their image form which makes analysis by a human user easier. The numbers next to object names are unique identifiers for those specific objects for use by Genesis. Every event has a recursive structure, that makes traversing and comparing events efficient. "The Union defeated the Confederacy" is a simpler event, while "The Union forced the Confederacy to return to the United States" is a more complicated one. This can be seen in both their text forms and image forms.*

Events make up the concept patterns that are used in this project. Their recursive structure makes comparing events efficient and easy to standardize.

2.3 The Conflict Story Set

Concept pattern comparison was tested on 15 short stories. Each story was about a historical armed conflict between countries, political groups, or political entities. All stories except two take place after World War II. The stories were originally written by Mark Finlayson, and were updated for Genesis's current parser by me. Table 1 contains the titles of the 15 conflict stories. Appendix X contains the full text for all stories.

American Revolution	Afghanistan Civil War	American Civil War
Cambodia-Vietnam Invasion	Chad-Libyan War	China Border War with India
China Border War with USSR	China Invasion of Tibet	China War with Vietnam
Congo Civil Conflict	Cuba - Bay of Pigs Invasion	Czechoslovakia Soviet Invasion
Nigerian Civil War	Persian Gulf War	Romania and Ceausescu

Table 1: Stories Used. *The titles of the 15 conflict stories used to test the Similarity Module.*

Each story has been simplified because the Genesis parser cannot handle the complex language they were originally written in. In addition, the simpler stories can be processed faster. However, the Similarity Module is indifferent to story simplicity. The Similarity Module takes in signals that contain events, and does not interact with the story wording itself. As long as the parser can create events out of the text, the Similarity Module can find concept patterns and compare stories.

Start story titled "American Revolution".

Britain is a country.
America is a country.
France is a country.
America is a country.

America was a colony of Britain.
Britain wanted to tax America.
Britain taxed America.

America resisted Britain because Britain taxed America.
America left Britain.
Britain invaded America.
America attacked Britain's troops because Britain invaded America.
France wanted to help America because Britain invaded America.
France gave money to America because France wanted to help America.
France gave weapons to America because France wanted to help America.
France gave ships to America because France wanted to help America.
America defeated Britain because France helped America.

The end.

Figure 4: Example Conflict Story. *The "American Revolution" story from the conflict story set.*

One effect of story simplification is that the person simplifying the story may change the interpretation of the story. For example, a story involving country A sending troops against country B could be simplified to *A invades B* or *A performs a preemptive strike on B*. While both wordings might be technically true, an invasion has slightly negative connotations while a preemptive strike is slightly positive. When the wording of a story changes, the comparison through concept units also changes. A story characterized as an invasion is different than one characterized as a preemptive strike, so the stories may not be considered similar. However, this is a feature and not a bug. Cultural bias in story simplification is a useful tool in story retrieval. One person may see an attack as justified, while another person may see an attack as unjustified. This will affect how they find stories to be similar. This can be used to understand how other cultures would react to a story. By loading Genesis with one culture's stories and biases, it can compare new stories base on its cultural story collection. The simplification and interpretation of stories allows the similarity module to work with cultural bias.

3 Concept Patterns

The basis of the similarity module is the concept pattern. A concept pattern is a group of events that are connected to each other through causal relations. Each event must be connected to at least one other event with a causal relation. The concept pattern can be of any size, and can have any structure as long as there is no unconnected event. Concept patterns do not contan loops. When two events are connected, it has a "leads to relation", where one event leads to another. In figure 5, examples of some possible structures are shown for concept patterns of size 2, 3, and 4.

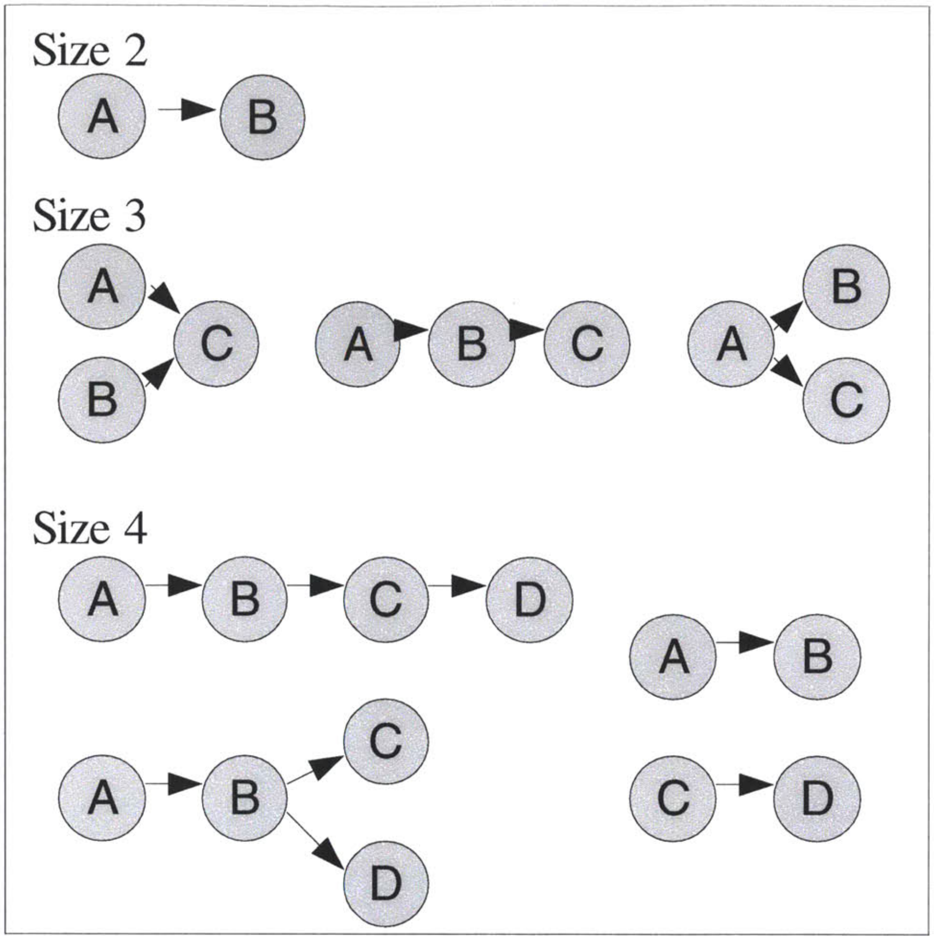


Figure 5: Possible Structure of Concept Patterns. *A concept pattern must have every event be connected to at least one other event through a causal relation.*

The concept pattern is the representation of an intermediate feature in a story. It can represent commonly found features in stories, such as *revenge*, *defensive aid*, and *border dispute*.

Concept patterns are created in two ways. The first type of concept pattern is user-defined, where a user of Genesis defines the concept patterns they want to find in stories in a separate text file. The second type of concept pattern is Genesis generated, where Genesis creates concept patterns from the story elaboration graph, and does not rely on any user input.

3.1 User-Defined Concept Patterns

User-defined concept patterns are named concept patterns identified by a user of Genesis, an example of which is *revenge*. Before I began my work on Genesis, the ability to discover user-defined concept patterns was already implemented. User-defined concept patterns are written by a user in a separate text file. They follow a prescribed pattern, shown in figure 6.

User-Defined Concept Pattern Structure
 	
`Start Description of X.`
`Define actors.` *(Example: xx is a country)*
`Define leads-to relations.` *(Example: xx harms yy leads to yy harms xx)*
`The end.`

Figure 6: Basic Structure of a User-Defined Concept Pattern.

First the concept pattern is named. Next the actors are defined in the form of *x is a person*, or *y is a country*. After the actors are defined, the leads-to relations between events are defined. One type of *revenge* for example, contains the leads-to event *a harms b leads to b harms* a. Examples of concept patterns used in the stories are shown in figure 7.

Land Dispute	Unsuccessful Rebellion
Start description of "Land Dispute". *xx is a country.* *yy is a country.* *zz is a region.* *xx's wanting to possess zz leads to xx's attacking yy.* *The end.*	*Start description of "Unsuccessful Rebellion".* *xx is a country.* *yy is a country.* *xx's disobeying yy leads to yy's controlling xx.* *The end*
Revenge Attack	***Unsuccessful Allied Help***
Start description of "Revenge attack". *xx is an anything.* *yy is an anything.* *zz is an anything.* *xx's defeating yy leads to zz's attacking xx.* *The end.*	*Start description of "Unsuccessful Allied Help".* *xx is a country.* *yy is a country.* *zz is a country.* *xx's wanting to help yy leads to zz's stopping xx.* *The end.*

Figure 7: Examples of User Defined Concept Patterns. *Pictured are* land dispute, unsuccessful rebellion, revenge attack, *and* unsuccessful allied help.

Once the concept patterns are defined, Genesis will search the story elaboration graph for examples. The Similarity Module then saves the found concept patterns in memory along with the story.

3.1.1 Advantages of User-Defined Concept Patterns

The advantages to using user-defined concept patterns are two-fold. The first is that the concept patterns are what the user deems most important to the comparisons. If a user only cares about comparing stories based on their conflicts, they can write concept patterns that only contain events related to conflict.

The second advantage is that the concept patterns are named. This makes it easy to count two different patterns as the same one. For example, the user can define revenge as *a harms b leads to b harms a* or *a is friends with c and b harms a leads to c harms b.* While both examples are revenge, they are different patterns. If a user wants them to be counted as the same, they simply name them the same.

3.1.2 Disadvantages of User-Defined Concept Patterns

The limitation to user-generated concept patterns is that an average user may not know what the correct concept patterns to use are. In a conflict story, what types of patterns should the user be looking for? Most users are not experts at the type of story being compared, so they may choose poorly, and there is no good way to determine if their choices of concept patterns are correct. In order to mitigate this disadvantage, I developed a way for Genesis to generated concept patterns in an unsupervised manner.

3.2 Genesis Generated Concept Patterns

Instead of having concept patterns defined by the user, Genesis can automatically generate concept patterns from the story elaboration graph. These concept patterns can be used exactly like user-defined concept patterns.

3.2.1 Why generate concept patterns?

User-defined concept patterns suffer from their reliance of the user choosing good patterns for story comparison. An expert in the field of stories being compared would choose good patterns, but this is impractical for anything but very small collections of stories. In addition, for new story domains there may be very few or no experts available. In order to compare large numbers of stories across disciplines, Genesis is able to automatically create concept patterns. Genesis generated concept patterns are created without any input from the user, and uses the story elaboration graph to find connected events.

3.2.2 Examples of Genesis Generated Concept Patterns

Genesis generated concept patterns are structured exactly the same as user-defined concept patterns, but they are not given an English label. The following are examples of concept patterns found by Genesis in the conflict stories, and the names have been added as a label for this thesis, and are not seen or used by Genesis.

Examples of Concept Units

Giving Aid (two events):
American revolution:
France helps America leads to *France gives money to America*
Cambodia-Vietnam invasion:
China helps Cambodia leads to *China gives weapons to Cambodia*

Revenge Attack (two events):
Afghanistan-civil-war:
Najibulla attacks Mujahideen leads to *Mujahideen attacks Najibulla*
American civil war:
Confederacy attacks Union leads to *Union attacks Confederacy*

Wanting an entity to stay, and dislike between entities, leads to a defeat (three events):
Nigerian civil war:
Nigerian-East dislikes Nigeria and *Nigeria wants Nigerian-East to stay* leads to *Nigeria defeats Nigerian-East*

American civil war:
Confederacy dislikes Union and *Union wants Confederacy to stay* leads to *Union defeats Confederacy*

Wanting an invasion leads to an invasion, which is defeated (three events):
Cuba bay of pigs invasion:
United States wants exiles to invade Cuba leads to *Exiles invade Cuba* leads to *Soldiers defeat Exiles*
China war with Vietnam:
Vietnam does not want China to invade Vietnam leads to *China invades Vietnam* leads to *Vietnam defeats China*

Figure 8: Examples of Genesis Generated Concept Patterns.

3.2.3 Concept Pattern Discovery

Concept patterns are discovered from the story elaboration graph. A concept pattern is essentially a sub-graph of the elaboration graph where every event connects to at least one other event. The user chooses the size of concept pattern to discover. Genesis finds every concept pattern in the graph of the chosen size, using the algorithm in figure 9.

Concept Pattern Discovery Algorithm

```
Find every group of events of size N
For every event a {
    For every other event after event b {
        For every other event after event c {
            Continue until the number of events in the concept pattern is N
        }
    }
}
The group of events (a, b, c) is the potential concept pattern
For every group of events in the elaboration graph of size N {
    If all events in the sub-graph are connected to at least one other event {
        Add this sub-graph as a concept unit
    }
}
```

Figure 9: The Concept Pattern Discovery Algorithm.

The algorithm works in $O(m^n)$ time, where n is the number of events in the elaboration graph, and m is the size of concept pattern being discovered. The slowest part of the algorithm is finding every possible group of events.

Genesis will not discard any concept patterns during discovery because it has no way of knowing which are relevant. However, this can create a very large number of concept patterns in a large story or a highly connected story. While doing story comparisons, Genesis will filter concept patterns in order to improve performance.

3.2.4 Concept Pattern Filtering

One issue with generated concept patterns is that a very large number of concept patterns can be found in a story. In the worst case scenario, Genesis will generate $O(n^m)$ concept patterns, where n is the number of events and m is the size of the concept patterns being generated. With large stories or highly connected stories, this number will blow up. Genesis filters out concept patterns that are considered "less useful" in order to decrease the amount of data being used. It does this by ignoring all concept patterns that only appear in one story.

This is reminiscent the approach taken by Chambers and Jurafsky in their work on unsupervised learning. (Chambers and Jurafsky, 2008) Due to their large amount of data, their system was preforming poorly. Accordingly, they eliminated rare occurrences of verb pairs, improving performance. By filtering out single-story concept patterns, story comparison can be greatly sped up.

3.2.5 Comparing Generated Concept Patterns

In order to compare stories, their individual concept patterns must be compared to each other. Two concept patterns may have different wording, but are considered the same by a reader. For example, two concept patterns may include a *harm* event, but one *harm* may be *kick* while one is *punch*. User defined concept patterns are easily compared despite differences in wording by checking if the name of the concept patterns are the same. However, generated concept do not have labels. Instead, generated patterns are compared in three steps: finding an alignment comparison, comparing the structure, and comparing word similarity.

3.2.5.1 Finding an Alignment

First, the two concept patterns are compared using an aligner. The events are compared to determine if the actors and objects line up. For examples, say two *revenge*-type concept patterns are being compared in figure 10.

Specific form of concept pattern: Concept Pattern 1: *Mark harms Jim leads to Jim harms Mark* Concept Pattern 2: *Maria harms Sally leads to Lenora harms Maria* **General form of concept pattern:** Concept Pattern 1: a *harms b leads to b harms a* Concept Pattern 2: *a harms b leads to c harms a*

Figure 10: Two Concept Patterns Being Compared. *The top examples are the specific form and contain names. The bottom examples are the more general form, and use variables to represent the entities. These concept patterns do not line up as their actors do not line up, although their actions do.*

While both contain the same actions, their actors do not line up. Genesis would not consider these the same concept pattern, as it does not align in terms of actors.

The similarity module uses an aligner written by Matthew Fay, also of the Genesis group. The aligner works by using a match tree to find the best match between two concept patterns. Take the events *John hit George on the head* and *Lisa hit Matt in the face*. The match tree would start with *John*. Then, it would try to match it to *Matt*, *Lisa*, and *face* from the second event. It chooses the most likely word, which in this case would be *Matt*, since both *John* and *Matt* are male names. However, this is not the correct matching. The match tree would be unable to find a match, or find a poor match, when it continued with matching the two events. Thus, it would go back and try to match *John* against the next most likely word *Lisa*, and it would find a high match with *John-Lisa*, *George-Matt*, and *head-face*. Once the aligner finds the best alignment, it returns it to the similarity module which uses it to determine if a good alignment has been found.

3.2.5.2 Comparing Structure

Once an alignment is found, Genesis compares the structure of the two concept patterns. For two concept patterns to be the same, they must have the same structure. For example, figure 11 shows two concept patterns that have the same events, but a different structure.

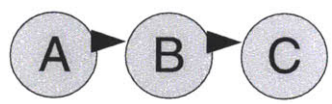
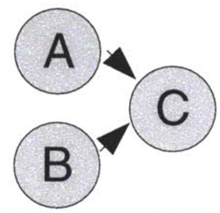
Concept Pattern	**Structure**
Mary hits Sally leads to *Sally hits Mary* leads to *Mary yells at Sally*	
Mary hits Sally and *Mary yells at Sally* leads to *Sally hits Mary.*	

Figure 11: Concept Patterns with Different Structures. *Two concept patterns, both involving events "Mary hits Sally", "Sally hits Mary", and "Mary yells at Sally". They do not have the same structure, so they are not considered the same by Genesis despite having the same events.*

These would not be considered the same, although they contain similar events. The algorithm used takes the alignment found, and uses it to compare the structure.

```
do_structures_match() {
     Find the highest scoring alignment using the aligner
     For each pair in the alignment (pair1) {
        For each other pair in the alignment (pair2) {
           If an event in pair2 is directly caused by an event in
           pair1 {
              If the other member of pair1 matches the other
              member of pair2 {
                 The structure matches so far, so continue
              }
              Else {
                 The structure does not match, so we've failed
                 matching. Return false.
              }
           }
        }
     }
     If we have made it this far without failing the structures
     match. Return true.
}
```

Figure 12: Pseudocode for the Structure Matching Algorithm.

In order to determine if two events are matched, they are compared using their WordNet threads, as explained in section 3.2.5.3 .

3.2.5.3 Comparing Word Similarity

An additional problem with comparing concept patterns is the generality of the words in the concept pattern. Generated concept patterns are always specific, as they use the words contained in the story. However, the event *a hits b* and *c kicks d* should most likely be considered the same, as they are both examples of personal physical harm. Genesis must determine whether two words are "similar enough" in order to be considered the same.

User-defined concept patterns can be generalized by default. When defining a concept pattern, the user can specify the level of generality. A *revenge* for example, could contain the action *harm*, *invade*, or *punch*. The user has complete control over the generality of the concept pattern.

Generated concept patterns cannot rely on user created names or summaries, and so instead use word threads from WordNet. WordNet is a large database of English words, from which Genesis creates objects called *threads*. For every word in WordNet, there are multiple threads where each thread corresponds to a different definition of the word. For example, the word "bag" has many threads, two of which are for the definitions "bag as in purse" and "bag as in capture". Genesis takes the first thread available, which is the most likely thread. A thread contains a sequence of words that steadily become more generalized. Figure 13 shows examples of word threads found in the conflict stories.

Object Name	***Object Thread***
Invade	{action, contend, attack, invade}
Attack	{action, contend, attack}
Coerce	{action, force, induce, compel, coerce}
Defeat	{action, get-the-better-of, defeat}
Want	{action, goal, desire, want}
Somebody	{thing, entity, physical-entity, object, whole, living-thing, organism, person, name, somebody}
PCT	{thing, entity, name, PCT}
People	{thing, entity, abstraction, group, people}
Cambodia	{thing, entity, abstraction, group, social-group, organization, unit, political-unit, state, country, name, Cambodia}

Figure 13: Examples of Word Threads from WordNet. *A thread goes from most general to most specific, and is used to determine if two words are similar. In this table,* invade *and* attack *would be considered similar as their threads are almost the same, with* attack *being the parent of* invade.

In order for two words to be "similar enough", their thread distance must be low. This means that either the words must be exactly the same, the words must share the parent word, or one of the words must be the parent of the other. For example, an *invasion* and an *attack* would be considered similar, because *attack* is the direct parent of *invasion*.

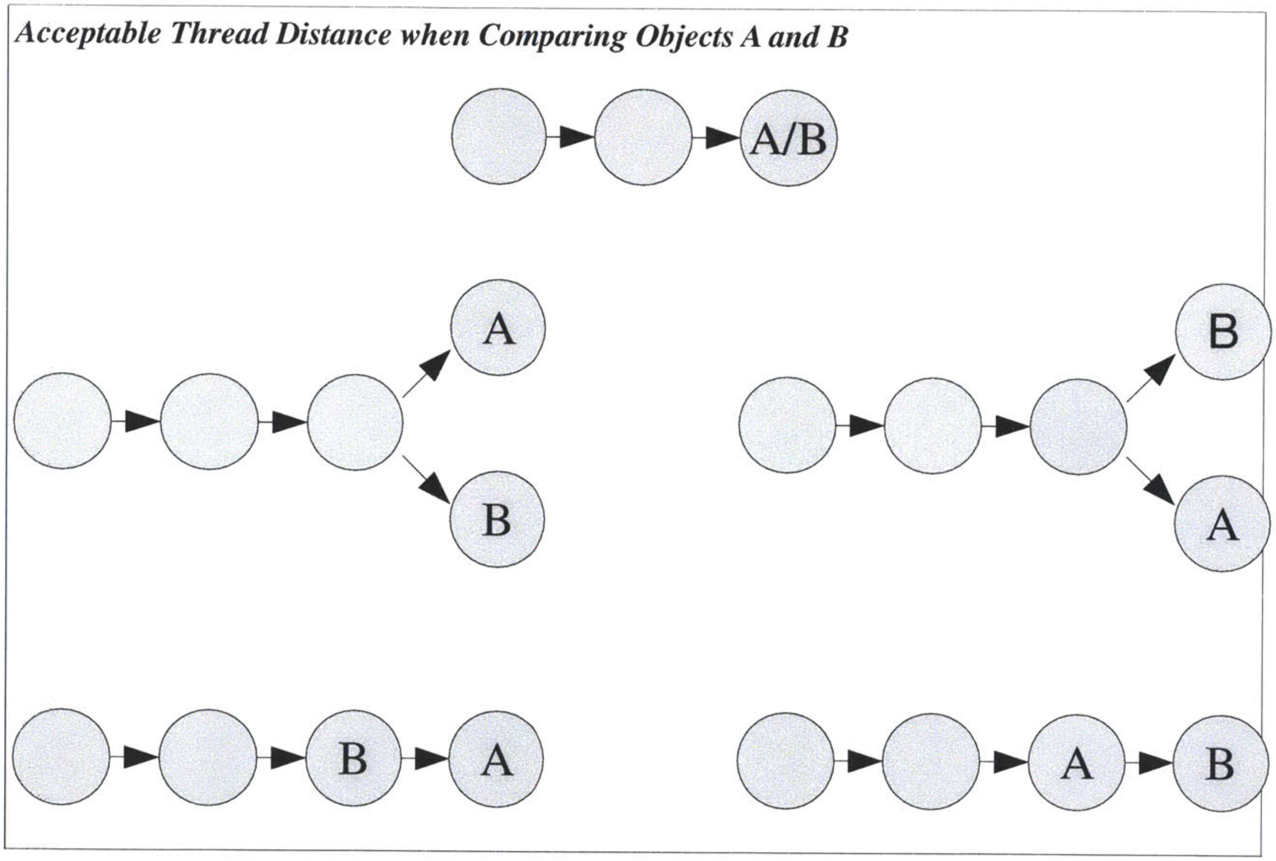


Figure 14: Samples Structures of Concept Patterns. *If objects A and B were being compared, they would have to be related in one of the ways shown in this figure in order to be considered similar enough for a match. They must either be the same, share a parent, or one must be the parent of the other.*

By using threads, Genesis can determine, automatically, if two concept patterns are similar enough to match.

3.3 Methods of Comparison

There are multiple methods of comparison for concept patterns. Each method has its advantages and disadvantages, and each are useful for different tasks.

3.3.1 Vector Angle

The first method of comparison is the vector angle of the concept patterns. This essentially compares the number of concept patterns in common for fast retrieval. The concept pattern counts for each story are saved in vectors. Then, the vector angle is calculated between the two story vectors to determine the similarity. The metric varies between 0.0 and 1.0, 1.0 being two vectors that are exactly the same.

This method is the most general method. It does not take into account ordering or importance of various concept patterns. However, it is the fastest method of comparison. It runs in O(n) time where n is the maximum number of concept patterns in a story.

3.3.2 In-Order Comparison

The second method of comparison is the ordering of concept patterns. For example, a revenge that is the result of a betrayal is different than a betrayal that is the result of a revenge. This method finds the longest common sub-string of concept patterns in two stories.

The order of concept patterns in a story is slightly difficult to determine. Concept patterns are made of multiple events, so they can take place over large parts of the story. The ordering of a concept pattern is determined by the last event in the concept pattern. This was decided because a concept pattern occurs when it is completed. For example, a *revenge* is only a revenge because the second *harm* event occurs due to the first *harm* event. Therefore, the *revenge* concept pattern does not occur until the last event. However, this is not a perfect system. For example, an *unwanted succession* is defined by a *succession* followed by an *attack* against the succeeding side. While the concept pattern is completed because the *succession* has been shown to be unwanted, it was the *succession* itself that defines the order. However, for the large majority of concept patterns the ordering can successfully be determined by the final event.

The similarity score is calculated by dividing the number of concept patterns in the longest common sub-string by the total number of concept patterns in the story with more concept patterns. A sub-string is defined by an ordered group of concept patterns. This ordered group does not have to be continuous, and can have concept patterns in between that are not included. This is so that the generality of a story does not affect story retrieval.

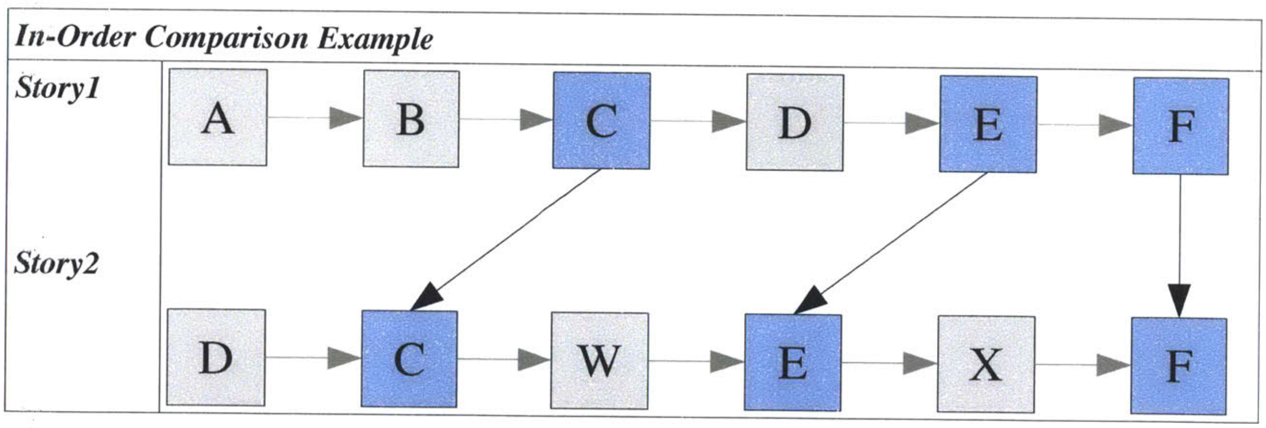


Figure 15: An Example of In-Order Concept Pattern Matching. *Story 1 and 2 contain concept patterns represented by* A, B, C, D, E, F, W, X. *With in-order matching, a sub-string of length 3 is found between the two stories. The sub-string does not have to be continuous, but does have to be in the same order. For example, even though story 2 contains the concept pattern D, it is not in the right order and thus is not included in the longest sub-string.*

The importance of ordering can also be seen in the comparison of the "China War with Vietnam" and "Congo Civil Conflict" stories, as shown in figure 16.

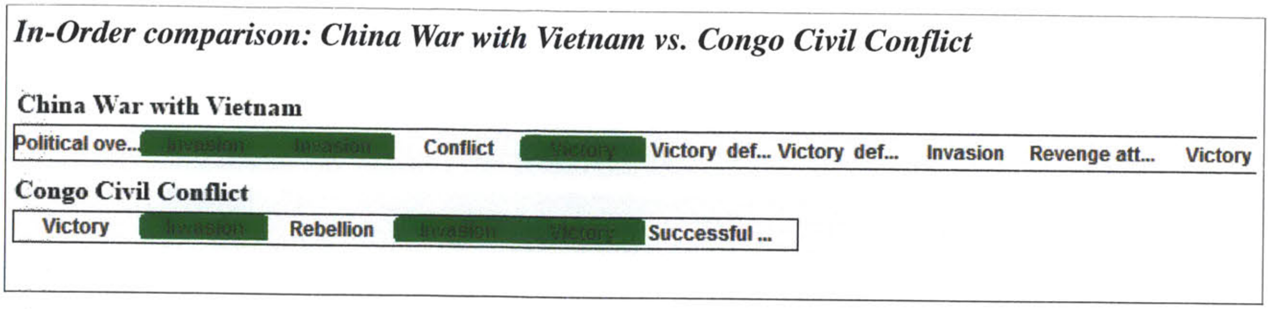


Figure 16: In-Order Matching of the "China War with Vietnam" and the "Congo Civil Conflict". *The green highlighted concept patterns are the matched longest sub-strings of the two stories. The concept patterns for the "China War with Vietnam" are {Political overthrowing, invasion, invasion, conflict, victory, victory defensive, victory defensive, invasion, revenge attack, victory}. The concept patterns for the "Congo Civil Conflict" are {Victory, invasion, rebellion, invasion, victory, successful rebellion}.*

In the stories as provided, both the "China War with Vietnam" and the "Congo Civil Conflict" contain two instances of invasion, and two instances of victory. However, one of the victory concept patterns are not matched due to in-order comparison.

In the "China War with Vietnam", the victory is a military victory at the end of the story. In the "Congo Civil Conflict", the victory is a political victory that is the cause of the rest of the events in the story. Because of their ordering, the two victories are very different, and thus makes sense that they are not matched. The ordering of the two stories makes a difference in their concept pattern similarities.

3.3.3 Rarity

The rarity of each concept pattern is also important in comparing stories. The rarity of a concept pattern is calculated by dividing the number of stories a concept pattern appears in by the total number of stories. This determines how common a specific concept pattern is in the stories read by Genesis. The rarity of a concept pattern can mean many things, three of which are:

- **Rare among a group of stories:** If a concept pattern is rare among a group of stories, it can be seen as more important when comparing similar stories. For example, when looking at a group of Disney-style fairy tales, two stories that have a princess marries a prince concept pattern, they do not seem as similar as two stories in which a concept pattern indicating princess ditches the prince and marries a poor commoner, because the ditch-the-prince concept pattern is rare.
- **Very common among a group of stories:** If a concept pattern is very common among some stories, it may be useful for grouping those stories. If a group of stories have concept patterns in common, but those concept patterns are much rarer among all stories, then that group of stories may make up a genre. For example, the "Disney-style fairy tale" genre may have concept patterns such as princess and prince fall in love, villain causes prince and princess to be kept apart, and prince and princess live happily ever after. If a new story is read with similar concept patterns, it may also be a Disney-style fairy tale.
- **Very common among most stories:** If a concept pattern is very common among most stories, then it is not particularly useful in deciding whether two stories are similar.

As an example of the influence of rarity, consider the rarities of the concept patterns in the American Revolution, with the concept rarities shown in table 2.

Concept Pattern	*Rarity*
Legal Disagreement	0.13
Invasion	0.93
Rebellion	0.20
Unwanted succession	0.20
Conflict	0.40
Allied Defense	0.07
Victory	0.80
Victory Defensive	0.27

Table 2: Concept Pattern Rarity. *The rarity of concept patterns found in the American Revolution story. An example of the invasion pattern is the most common concept pattern, while an allied defense is the most rare. Rarity is calculated by dividing the stories a concept pattern appears in by the total number of stories.*

The most common concept pattern is invasion, as it occurs in almost every story. Because of this, an invasion is a poor measure of similarity between these stories, but a very good indicator that the story is about a conflict. On the other hand, an allied defense is much more rare and therefore more important when measuring story similarity in the conflict domain. The rarity of concept patterns determine their use in story retrieval.

3.3.4 Comparing stories with generated concept patterns

One method of comparison used with generated concept patterns is the total number of concept patterns in common between two stories. The reason this may be used instead of the vector angle between two stories is that there are many generated concept patterns for each story. When doing vector comparison, the value of similarity between two stories are all very low. With so many generated concept patterns, a large difference in the number of concept patterns between stories will greatly change the results. Because the number of concept patterns is dependent on the connectivity and size of story, these aspects can affect the comparison poorly. By comparing the number in common, as opposed to the percent in common, the large number of generated concept patterns will not greatly affect story comparison.

4 Results

This section contains interesting results from story comparison. Three different methods of comparison were tested: keyword analysis, user-defined concept patterns, and Genesis generated concept patterns.

4.1.1 Keyword analysis

Keyword analysis was run on the stories in order to have a base-line of comparison techniques. Keyword comparison is done using the vector angle of words in the story.

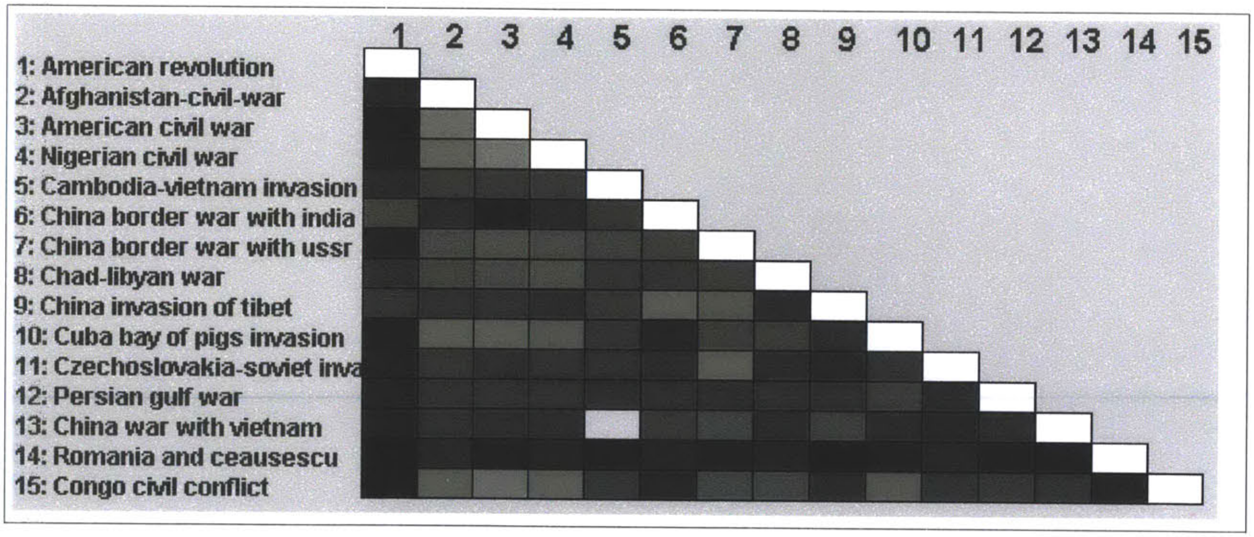


Figure 17: Keyword Analysis Results. *Lighter rectangles have higher matches, darker rectangles have lower matches.*

4.1.2 User-defined concept patterns

The first type of concept pattern is user-defined. Two methods were used for this concept pattern, vector dot product and in-order.

4.1.2.1 Vector Dot Product Comparison

The vector dot product method has a higher standard deviation than keyword comparison, meaning that user-defined concept patterns found a wider range of similarities than keywords. The mean and standard deviation for each method are shown in table 3.

Method	**Mean**	**Standard Deviation**
Keyword	0.267	0.119
User-Defined Concept Pattern	0.364	0.200

Table 3: The mean and standard deviation of similarity scores generated by each method. The standard deviation of story comparison by concept pattern is almost twice that of keyword comparison. Similarity scores are on a scale from 0.0 (not similar) to 1.0 (identical).

With keywords, stories had relatively the same level of similarity between each other. Most stories had similar wording of conflicts, and different names of actors. However, concept patterns were able to find the differences and similarities in the plots themselves.

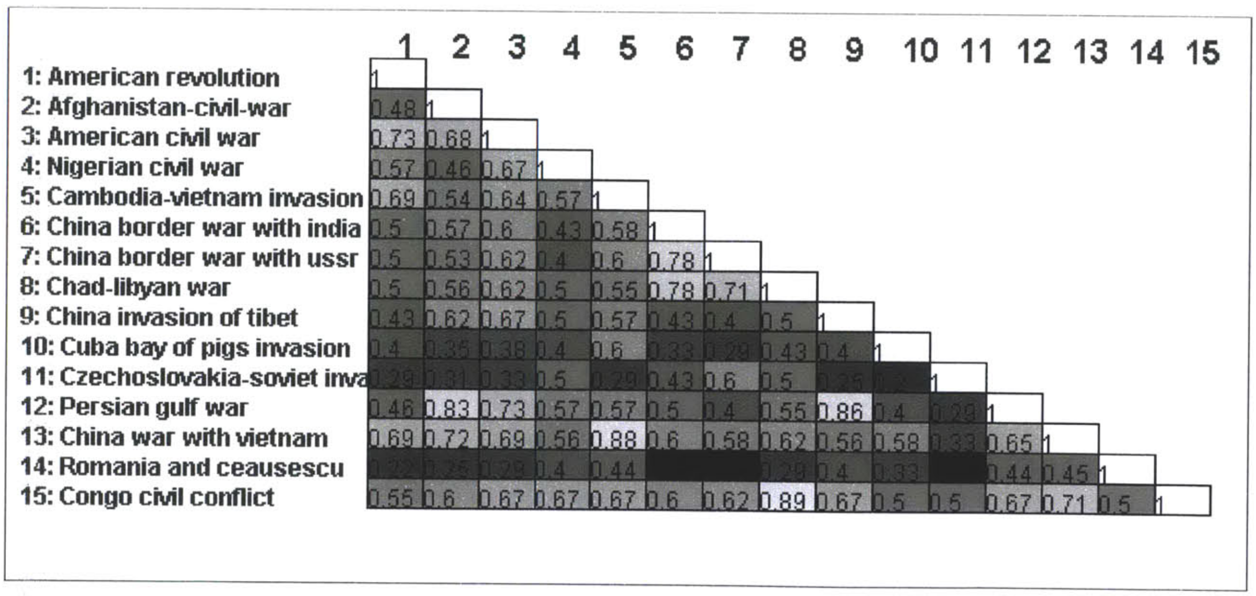


Figure 18: User-Defined Concept Pattern Results Using the Vector Dot Product. *The values in each rectangle are the match values. Darker squares are lower scores, and lighter squares are higher.*

As an example of where user-defined concept patterns are very different than keyword patterns is the comparison between the "American Civil War" and the "American Revolution". While in each war the enemy was different, they both involved a succession over legal rights (although the rights were different), and involved two parts of a country fighting each other. By using user-defined concept patterns, the two stories are rated very similar. However, they are a much lower rated comparison when using keywords.

Comparing stories using concept patterns is more congruence with my own interpretations. For example, the deviation of similarity score values is much higher than in keyword comparison on the fifteen conflict stories on which I ran experiments, just as I view story pairs as varying considerably in similarity. Following are three examples where concept pattern comparison finds similar stories but keyword comparison falls short.

- **American Revolution and the American Civil War:** Concept pattern comparison picks out the American Revolution and the American Civil War as being similar giving them a similarity score of 0.67, as they have several concept patterns in common (unwanted succession, victory, conflict, legal disagreement). This makes sense, as both stories are about a part of a country rebelling from the main country over legal disputes (taxes in one case, slaves in the other). In the word comparison, these stories have a very low similarity score of 0.1 (as shown by the red). By using concept patterns to compare stories, more meaningful story comparison is performed.
- **China border War with India and the Cambodia-Vietnam Invasion:** Another example of the concept pattern comparison succeeding while the keyword comparison fails is the comparison between the China border War with India and the Cambodia-

Vietnam Invasion. In both cases, two countries fought over an area of land (the Mekong Delta in the Cambodia-Vietnam conflict, and the Assam in the China-India conflict). The relevant concept patterns found are a land dispute along with two invasions (one by each country into the disputed region), which gives the comparison score of 0.71. The keyword comparison however, rates them as relatively dissimilar with a score of 0.26.

- **Afghanistan Civil War and the Czechoslovakia Soviet Invasion:** An example where keyword comparison has decided that two stories are similar, where in fact they are not, are the Afghanistan Civil War and the Czechoslovakia Soviet Invasion. Keyword comparison gives a score of 0.48, which is very high for keyword comparisons. However, the concept pattern comparisons give them a score of 0.0. The stories, while both involve the Soviet Union, are very different conflicts. In the Czechoslovakia Soviet Invasion, the Soviet Union invaded Czechoslovakia due to political reform. In the Afghanistan Civil War, the Soviet Union funded one side of a civil war, but did not actually attack. Thus, the two conflicts are quite different, which is shown by the concept pattern comparison.

4.1.2.2 In-Order Comparison

By finding the longest common sub-string between two stories, interesting patterns can be found. In figure 17, the "American Revolution" and "American Civil War" stories are compared.

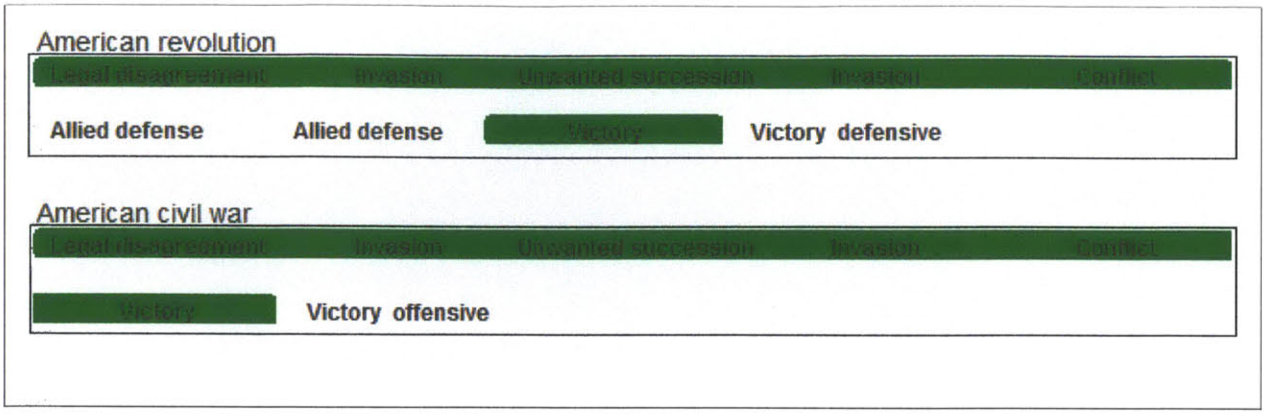


Figure 19: In-Order Comparison of "American Revolution" and "American Civil War". *In-order comparison of the "American Revolution" and the "American Civil War". Events highlighted in green are part of the longest sub-strings.*

The two stories are very similar except for two key points. Both start with a *legal disagreement* (although taxation and slavery are very different issues), and then have an *unwanted succession* and an *invasion* to stop the succession. There is then a *conflict* between the two sides, and one winner. However, the endings of each story is different. In the "American Revolution", the succession was successful, but in the "American Civil War" the succession failed. Part of the reason this occurred, was because the Americans in the "American Revolution" received help from the French in an *Allied defense*. By doing an in-order comparison, it becomes clear that the two conflicts are very much alike.

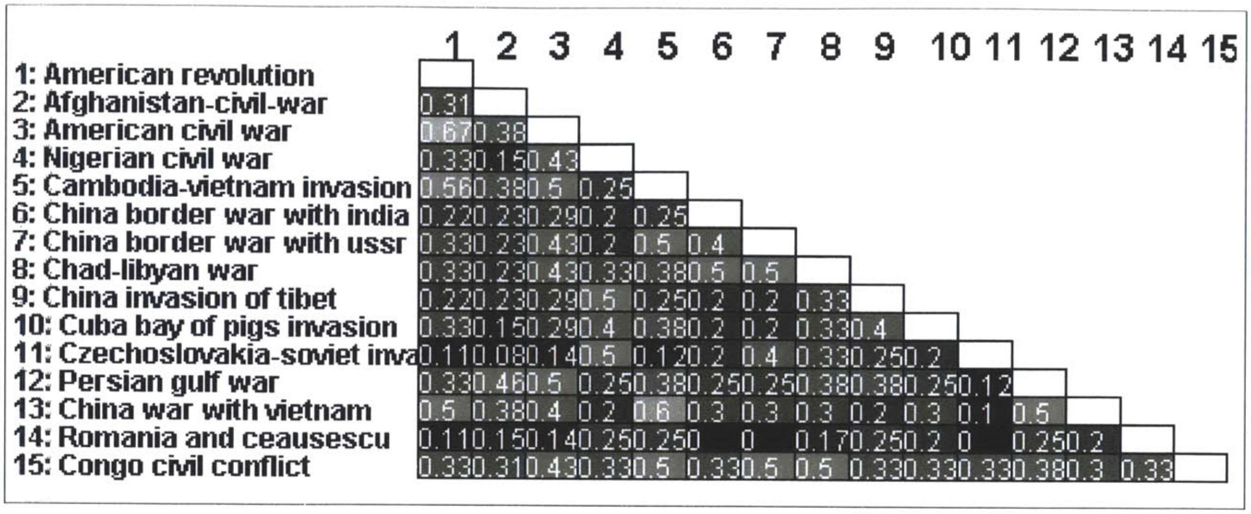


Figure 20: Results for In-Order Comparison of User-Defined Concept Patterns. *Lighter rectangles have higher matches, darker rectangles have lower matches. The numbers pictured in each square are the scores for in-order comparison.*

4.1.3 Genesis generated concept patterns

Genesis generated concept patterns act similarly to user-defined concept patterns, but there are many more generated concept patterns than user-defined concept patterns. For both user-defined and Genesis generated patterns, the "Czechoslovakia-Soviet Invasion" and the "Romania and Ceausescu" stories have zero concept patterns in common. However, while the "China Invasion of Tibet" and the "China Border War with India" have a very high similarity score of 0.89 when compared using user-defined concept patterns, they have a lower score of 0.5 when compared using generated concept patterns.

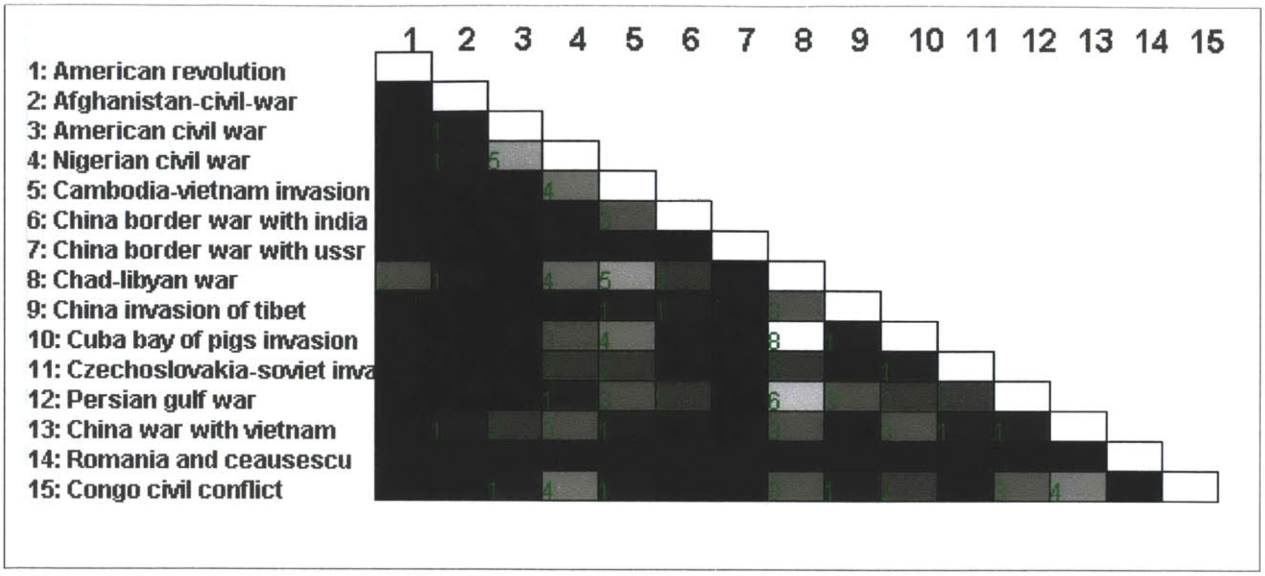


Figure 21: Results from Genesis Generated Concept Patterns Using the Vector Dot Product. *Lighter rectangles have higher matches, darker rectangles have lower matches. The numbers pictured in each square are the vector dot product of the match.*

5 Comparing Methods to Human Study

Methods of story retrieval were also compared to a human study in order to determine if they were viable methods of comparison. The method that had the closest results to the human study was the vector dot product of Genesis generated concept patterns.

5.1 Study

In order to establish a ground truth of story comparison, a human study was done to find how people would rate the stories as similar. In the study, 11 people were given the 15 conflict stories to compare. For each story, they chose 1 or 2 stories that they found to be the most similar. The results are compiled in Figure 22.

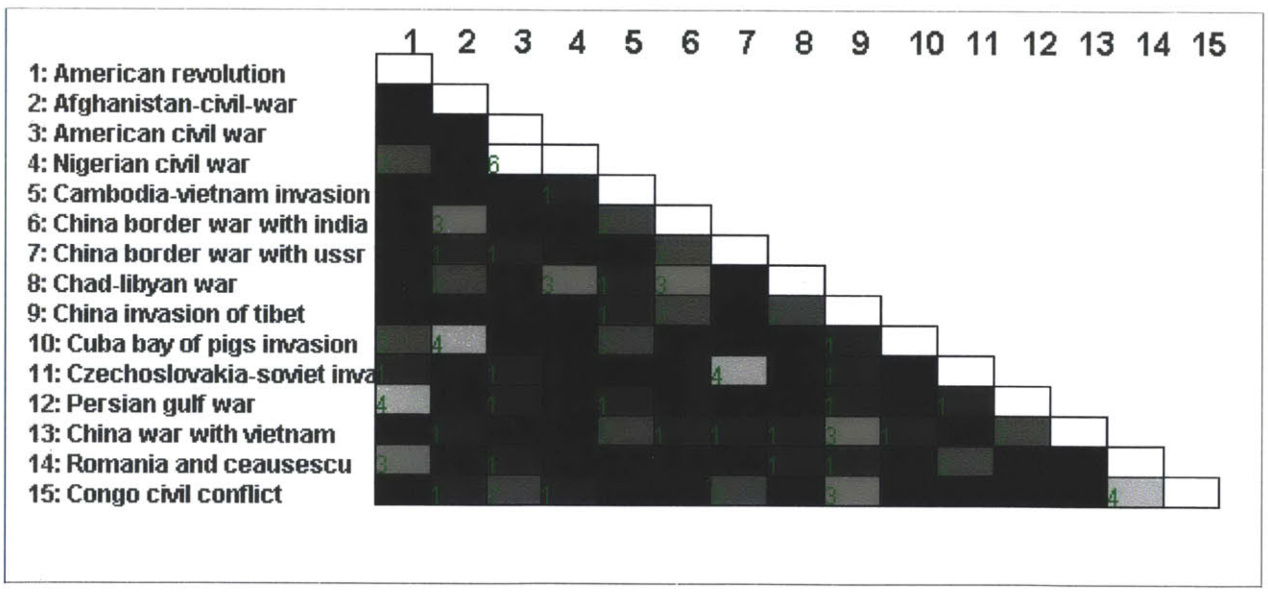


Figure 22: Human Testing Results. *Lighter rectangles have higher matches, darker rectangles have lower matches. The numbers pictured in each square are the number of people who felt that match was the best for each story.*

There was a low level of agreement among study takers. This is likely because study takers were not experts in the domain, and thus were not able to pick consistent features to for comparison. In order to quantify the results for comparison, each match was given a score on the range of 0 to 1.0. To calculate the score, the number of people finding that match to be the best was divided by the total number of participants. So a match that all 11 people thought was the best has a score of 1.0, while a match that nobody thought was the best has a score of 0.0. These scores are compared to the methods of story comparison developed for Genesis.

5.1.1 Keyword analysis

Keyword analysis is the base-line method. The standard deviation of error for the keyword analysis was 0.24.

5.1.2 User-defined concept patterns

5.1.2.1 Vector Dot Product Comparison

The user-defined concept pattern's error had a standard deviation of 0.48, which is much higher than keyword comparison. However, because the study participants were not experts, and because I created the user-defined concept pattern and am not an expert on conflicts, this is not unexpected.

5.1.2.2 In-Order Comparison

In-order comparison of user-defined concept pattern had a standard deviation of 0.28 for error. It performs better than the vector dot product method.

5.1.3 Genesis generated concept patterns

Genesis-generated concept patterns performed the best among all the methods when compared to the human study. It had an error with a standard deviation of 0.12. This is the lowest standard deviation from the human study results.

6 Conclusion

6.1 Discussion

The most successful method when compared to the human study is finding the dot product of Genesis generated concept patterns. The study that was done is not conclusive, as non-experts performed the story comparison. Genesis generated concept patterns have the least human bias, as they do not require a person to define concept patterns, possibly explaining why generated concept patterns are an effective way to retrieve stories.

Independent of the human study, I believe that the best method for story retrieval is by using generated concept patterns. A novice is unlikely to choose good user-defined concept patterns for retrieval. While a human expert would be able to define good concept patterns, this can be very impractical depending on the story domain. Generated concept patterns are the least biased method, and can be used by even a complete novice in the chosen story domain.

6.2 Applications and Future Work

The Similarity Module is a tool to compare stories and retrieve similar stories. In this section, possible applications and future work is discussed.

6.2.1 Expert Study

A study in which experts in the domains of stories being compared would be a better method of finding the ground truth. This would be possible by either finding experts in the conflict domain, or changing the story domain to something with more experts.

6.2.2 Precedent-Based Reasoning

The main purpose of the Similarity Module is to perform story retrieval for precedent-based reasoning. One application of precedent-based reasoning is understanding unexplained events in a story. For example, "Story 1" contains the event *A kills B*. However, "Story 1" offers no explanation for the killing. In order to determine why A killed B, Genesis will retrieve the most similar story to "Story 1", titled "Story 2". In "Story 2", *C kills D because D hurt C* is a concept pattern found (an instance of *revenge*). Genesis will reason that B must have hurt A sometime in the past, causing A to kill B. The reason behind A killing B is affected by the precedents set in the most similar stories. An alternate explanation may be that A is insane, or that A was paid to kill B. The story precedents are used by Genesis to reason about the events in a story.

6.2.3 Predictions

One application of the Similarity Module is predicting the end of stories. Basic story prediction is already implemented in the Similarity Module. The Similarity Module is given an story that is missing an ending, and by using story retrieval, predicts the most likely ending of the story. For example, given the very short story "A hits B leads to B hits A", the module attempts to predict what will happen next. If the most similar story contains an offensive victory by *A*, then the module will predict that *A* will defeat B in the unfinished story.

The module finds the most similar story based on in-order comparison. The reason in-order comparison is used is to find a long pattern that leads up to an appropriate ending. Given the longest matching in-order sub-string, the module will predict the events that come after the longest sub-string as the ending.

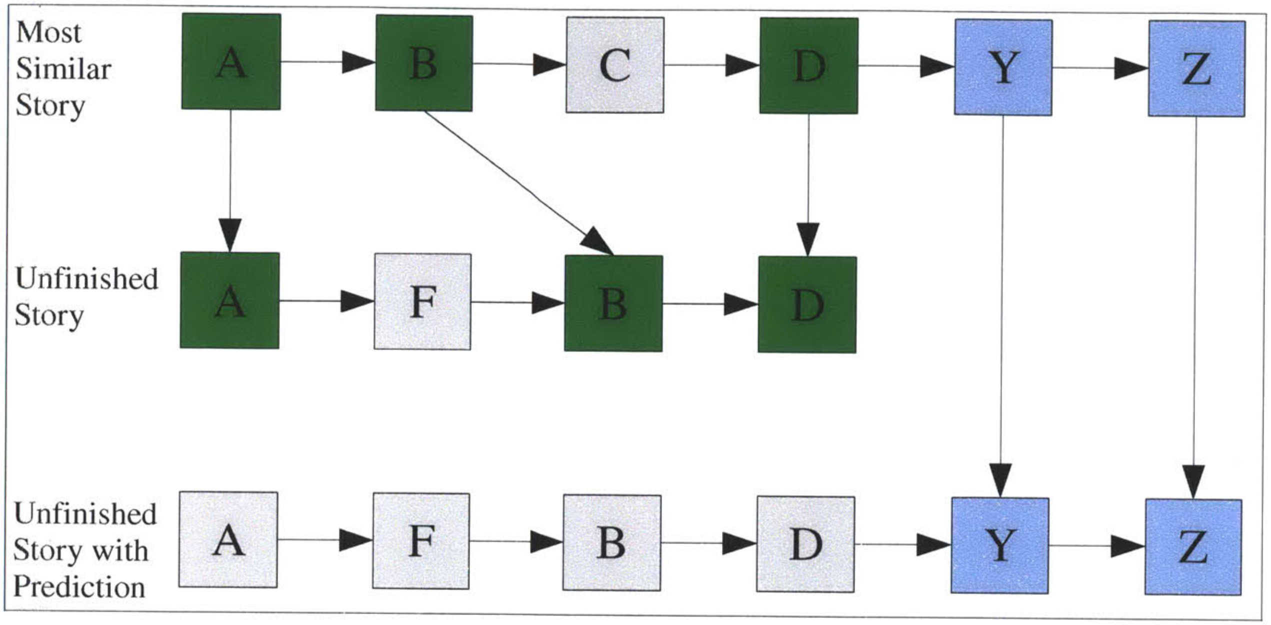


Figure 23: Story Prediction. *Story prediction using the most similar story retrieved by the similarity module.*

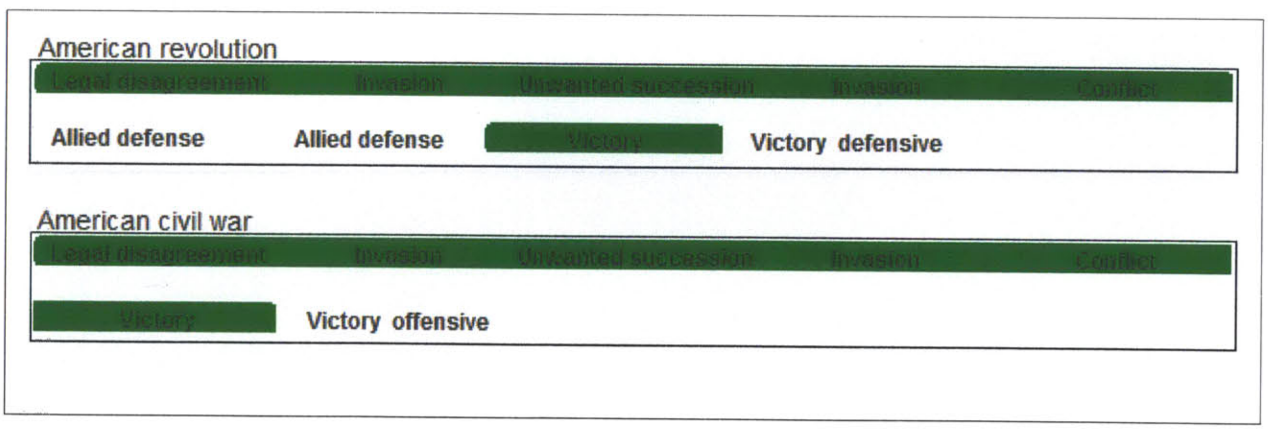


Figure 24: Comparison of the "American Revolution" and the "American Civil War" for Prediction. *With prediction, a similar story will change its ending from an offensive to defensive victory if there is an* allied defense.

As an example of prediction, in figure 24 the stories are almost exactly alike. The only attributes distinguishing them are their endings and an *allied defense* in the "American

Revolution". If a new unfinished conflict similar story is read, and compared against these two stories, then the deciding factor will be the *allied defense*. If the new story contains an *allied defense*, then Genesis will predict that the succession succeeded. Otherwise, the succession will have failed. This is a reasonable prediction, as a succession against a stronger power is much more likely to succeed if the succession has help.

The disadvantage to this story prediction is that the most similar story can change the prediction drastically, regardless if other similar stories would cause different predictions. If the most similar story ends in a bizarre manner, Genesis will not know that the ending is considered strange. To iron out these anomalies, multiple similar stories can be used. This has not been implemented, but would improve precedent-based prediction. Taking some number of the closest stories and the unfinished story, the longest in-order sub-string of concept patterns is found that is common among all the stories. A second longest sub-string is then found from the concept patterns that occur after the longest sub-string in the finished stories. This second longest sub-string is the story prediction.

One further extension of story prediction is storing common sub-strings found in stories. When an unfinished story contains the first part of a common sub-string, the remainder of the common sub-string will be the prediction. This will enable retrieval based on common story patterns as opposed to only the most similar stories.

6.2.4 Grouping

A third application of story retrieval is grouping stories into categories for faster retrieval. In this project, the category of story used was conflict. Conflict stories share similar concept patterns, such as *victory*, *invasion*, and *land dispute*. By finding stories with similar groups of concept patterns, story categories can be created.

By creating story groups, more efficient retrieval can be performed. If a user wants to retrieve the most similar story, they can specify to only look in the categories their story falls under. When looking for the most similar conflict story, Genesis should not match against stories that do not share concept patterns in common with conflict stories. By filtering out stories that will not be the most similar, computation time can be saved. By utilizing story grouping with concept patterns, more efficient retrieval can be implemented.

6.3 Contributions

My contributions to the Genesis project were the new Similarity Module, 15 new conflict stories, and an experiment on human story comparison.

6.3.1 Similarity Module

The Similarity Module is designed to retrieve similar stories for use in precedent-based reasoning. It uses concept patterns, which are structures of events that are connected by causal relations. They are the intermediate features of a story that facilitate powerful retrieval. There are two types of concept patterns that can be used for story retrieval.

- **User-Defined Concept Pattern:** This type of concept pattern is defined and labeled by the user in a separate text file. The creation and discovery of user-defined concept patterns was implemented prior to my research and the Similarity Module. User-defined concept patterns are useful when tailoring story retrieval for a specific purpose, such as comparing the romantic subplots in stories. However, they can perform poorly when poor concept patterns are chosen.

- **Genesis-Generated Concept Pattern:** This type of concept pattern is automatically discovered by Genesis from the story elaboration graph. Genesis uses all possible concept patterns that appear in more than one story. While this makes customization of story retrieval difficult, it takes out human error in defining concept patterns. Retrieval using Genesis-generated concept patterns had results more similar to the human study.

The Similarity Module contains multiple methods of story comparison and retrieval. Each method is useful for different tasks.

- **Vector Matching:** The fastest and most general story retrieval method, it uses the vector dot product to compare stories' concept patterns. It essentially retrieves the story with the most concept patterns in common.
- **In-Order Comparison:** The Similarity Module is capable of finding the longest in-order common sub-string between two stories. When using user-defined concept patterns, in-order comparison is the most similar to the results from the human study. In-order comparison is also useful for tasks such as prediction.
- **Rarity:** The Similarity Module is capable of calculating the rarity of a concept pattern among a group of stories. The rarity of a concept pattern makes a difference when performing story retrieval and comparison. This is most useful for story grouping. When used in retrieval, results vary according to rarity details:
    - If a concept pattern is rare among a group of stories, then it is useful in retrieval

among that particular group of stories.

- If a concept pattern is common among a group of stories, but rare otherwise, then it is not as useful for retrieval among that group, but may be a good indicator that a story is a member of the group.
- If a concept pattern is common among most stories, then it is unlikely to be useful when doing story retrieval.

Finally, the Similarity Module contains a story prediction system. The prediction routine predicts the ending of an unfinished story by retrieving the most common story using in-order comparison.

6.3.2 Conflict Stories

15 conflict stories were added to Genesis's story database. In addition, 39 common sense rules and 37 user-defined concept patterns were defined. These conflict stories and rules formed the basis of the experiments done on story retrieval.

6.3.3 Human Study

A human study involving 11 participants set a ground truth for the various methods used in this thesis. Humans in the study performed the same comparison task as the Similarity Module, by retrieving the most similar story to each story in the conflict corpus. Genesis generated concept patterns, with vector matching, were shown to be the most similar to human results, with an error with a standard deviation of only 0.12. In contrast, keyword comparison had an error of 0.24, and user-defined concept patterns had an error of 0.48.

Bibliography

Morteza Dehghani, Dedre Gentner, Kenneth D Forbus, Hamed Ekhtiari, and Sonya Sachdeva. 2009. *Analogy and moral decision making.* New Frontiers in Analogy Research Proceedings of the Second International Conference on Analogy.

Brian Falkenhainer, Kenneth D. Forbus, and Dedre Gentner. 1989. *The structure mapping engine: Algorithm and examples.* Artificial Intelligence, 41(1):1–63.

Mark A. Finlayson and Patrick Henry Winston. 2005. *Intermediate features and informational-level constraint on analogical retrieval.* In Proceedings of the 27th Annual Meeting of the Cognitive Science Society, pages 666–671.

Finlayson, M.A. & Winston, P.H., Analogical Retrieval via Intermediate Features: The Goldilocks Hypothesis. MIT CSAIL Technical Report No. 2006-071

Mark A. Finlayson. 2012. *Learning Narrative Structure from Annotated Folktales.* Ph.D. thesis, MIT.

D Gentner and K Forbus, 1991. *MAC/FAC: A model of similarity-based access and mapping.*

Caryn E. Krakauer and Patrick H. Winston. 2012. *Story Retrieval and Comparison Using Concept Patterns.* Unpublished paper presented at the Workshop on Computational Models of Narrative, Istanbul, Turkey.

Wendy Lehnert. 1981. *Plot units and narrative summarization.* Cognitive Science, 5(4):293–331.

Shimon Ullman, Michel Vidal-Naquet, and Erez Sali. 2002. *Visual features of intermediate complexity and their use in classification.* Nature Neuroscience, 5(7).

Patrick Henry Winston. 2011. *The strong story hypothesis and the directed perception hypothesis.* In Pat Langley, editor, Technical Report FS-11-01, Papers from the AAAI Fall Symposium, pages 345–352. AAAI Press.

Appendix A: Conflict Stories

American Revolution

Britain is a country.
America is a country.
France is a country.
America is a country.

America was a colony of Britain.
Britain wanted to tax America.
Britain taxed America.

America resisted Britain because Britain taxed America.
America left Britain.
Britain invaded America.
America attacked Britain's troops because Britain invaded America.
France wanted to help America because Britain invaded America.
France gave money to America because France wanted to help America.
France gave weapons to America because France wanted to help America.
France gave ships to America because France wanted to help America.
America defeated Britain because France helped America.

American Civil War

The United States is a country.
The Confederacy is an entity.
The Union is an entity.
The Confederacy was a region of the United States.
The Union was a region of the United States.

The Union disliked the Confederacy because the Confederacy possessed slaves.
The Confederacy left the United States because the Confederacy disliked the Union.
The Union wanted the Confederacy to stay in the United States.

The Union attacked the Confederacy.
The Confederacy attacked the Union because the Union attacked the Confederacy.
The Union was stronger than the Confederacy.
The Union defeated the Confederacy.
The Union controlled the Confederacy because the Union defeated the Confederacy.
The Union forced the Confederacy to return to the United States.

Afghanistan Civil War

Najibullah was a man.
Afghanistan is a country.
The Soviets are an entity.
The Mujahideen were a men.
The Mujahideen were muslim.
The Jamiat-i-Islam was a faction of the Mujahideen.
Massoud was a man.
Massoud led the Jamiat-i-Islam.
The Hizb-i-Islam was a faction of the Majahideen.
Hekmatyar was a man.
Hekmatyar led the Hizb-i-Islam.

Najibullah controlled the cities of Afghanistan.

The Soviets liked Najibullah because Najibullah controlled the cities of Afghanistan.
The Soviets liked Najibullah because Najibullah liked the Soviets.
The Soviets gave weapons to Najibullah because the Soviets supported Najibullah.

The Mujahideen were confident because the Mujahideen controlled the countryside of Afghanistan.
The Mujahideen opposed Najibullah because Najibullah liked the Soviets.
The Mujahideen opposed Najibullah because the Soviets supported Najibullah.
The Mujahideen opposed Najibullah because the Mujahideen were confident.

The Mujahideen attacked Najibullah because the Mujahideen opposed Najibullah.
Najibullah attacked the Mujahideen because the Mujahideen attacked Najibullah.

The Soviets stopped giving weapons to Najibullah because the USSR collapsed.

The Mujahideen were stronger than Najibullah because the Soviets stopped giving weapons to Najibullah.

The Mujahideen defeated Najibullah.

Massoud wanted to control Afghanistan.
Hekmatyar wanted to control Afghanistan.
Hekmatyar disliked Massoud.
Massoud disliked Hekmatyar.

Hekmatyar's faction attacked Massoud's faction because Hekmatyar wanted Hekmatyar to control Afghanistan.

Cambodia-Vietnam Invasion

Vietnam is a country.
Cambodia is a country.
Vietnam was stronger than Cambodia because Vietnam was bigger than Cambodia.

The KhmerRouge was a movement.
The KhmerRouge were Khmer.
The KhmerRouge ruled Cambodia.
The MekongDelta is a region.
Vietnam possessed the MekongDelta because the MekongDelta is in Vietnam.

Cambodia thought Vietnam wanted to control Cambodia.
Cambodia did not trust Vietnam because Vietnam wanted to control Cambodia.
Cambodia wanted to possess the MekongDelta.
Cambodia disliked Vietnam.

Vietnam did not want Cambodia to possess the MekongDelta.
Cambodia invaded Vietnam because it wanted to possess the MekongDelta.
Vietnam disliked Cambodia because Cambodia invaded Vietnam.

The USSR liked Vietnam.
China disliked the USSR.
China supported Cambodia because China disliked the USSR.
China gave weapons to Cambodia.

Vietnam invaded Cambodia because Cambodia invaded Vietnam.
Vietnam defeated Cambodia because Vietnam was stronger than Cambodia.
Vietnam controlled Cambodia.

Chad-Libyan War

Chad is a country.
Libya is a country.
France is a country.
Italy is a country.
France possessed Chad.
Italy possessed Libya.

The NorthernStrip is a region.
Chad possessed the NorthernStrip.
The NorthernStrips's people were Muslim.
Libya's people were Muslim.
Habre is a person.
FrolinatRebels is an entity.

LibyaPeople were people.
LibyaPeople lived in Libya.
NorthernStripPeople were people.
NorthernStripPeople lived in the NorthernStrip.

Muslim is an religion.
LibyaPeople were Muslim.
Libya liked the NorthernStrip's people because the NorthernStrip's people were Muslim.
Libya wanted to possess the NorthernStrip because Libya liked the NorthernStrip's people.

France and Italy agreed to give the Northern Strip to Libya.

Libya helped the FrolinatRebels because Libya liked the NorthernStrip.
Libya helped the FrolinatRebels because the FrolinatRebels lived in NorthernStrip.
Libya invaded the NorthernStrip because Libya wanted to possess the NorthernStrip.

France left Chad.
Italy left Libya.
Libya invaded the NorthernStrip because France left Chad.
Libya invaded the NorthernStrip because Italy left Libya.

Habre lived in the NorthernStrip.
Habre was a leader of the FrolinatRebels.
Habre disliked Libya because Libya invaded the NorthernStrip.
Habre disliked Libya because Habre lived in the NorthernStrip.
Habre helped Chad because Habre disliked Libya.
Habre attacked Libya because Habre disliked Libya.
Habre defeated Libya.

China Border War With India

India is a country.
China is a country.
Britain is a country.
IndiaTroops is an entity.
ChinaTroops is an entity.
Assam is a region.
IndiaBorders are a thing.

Britain defined IndiaBorders.
IndiaBorders were vague because Britain defined India's borders.
Britain left India.

Assam was disputed because IndiaBorders were vague and because Britain left India.

India wanted to possess Assam because Assam was disputed.
China wanted to possess Assam because Assam was disputed.

India invaded Assam because India wanted to possess Assam.
China invaded Assam because India invaded Assam.
China invaded Assam because China wanted to possess Assam.
India attacked China because India invaded Assam and China invaded Assam.

China Border War with USSR

'China is a country.
USSR is a country.

Mao was a person.
Mao was the leader of China.

Communist is a system.

China was communist.
The USSR was communist.
China obeyed the USSR because the USSR was communist and China was communist.

The USSR disrespected Mao.
China disliked the USSR because the USSR disrespected Mao.

China wanted to alter China's border.
The USSR disrespected Mao because the USSR refused to alter China's border.

China disobeyed the USSR because China wanted to alter China's border and the USSR refused to alter China's border.
The USSR attacked China because China disobeyed the USSR.
China attacked the USSR because China disliked the USSR and the USSR attacked China.

China Invasion of Tibet

The United States is a country.
China is a country.
Tibet is a country.
The DalaiLama is a person.
The DalaiLama was the ruler of Tibet.
India is a country.

China wanted to possess Tibet.
Tibet did not want China to possess Tibet.
Tibet asked India to help Tibet.
India did not want to offend China.
India refused to help Tibet because China wanted to possess Tibet and India did not want to offend China.
The United States wanted to help Tibet.
India stopped The United States helping Tibet because the United States wanted to help Tibet and India did not want to offend China.

China invaded Tibet because China wanted to possess Tibet.

Tibet's soldiers were less experienced than China's soldiers.
Tibet's army was smaller than China's army.
China defeated Tibet because China invaded Tibet and Tibet's soldiers were less experienced than China's soldiers and Tibet's army was smaller than China's army.
China possessed Tibet because China defeated Tibet.

China War with Vietnam

The KhmerRogue is an entity.
Cambodia is a country.
Vietnam is a country.
China is a country.

The KhmerRouge controlled Cambodia.
Cambodia attacked Vietnam because the KhmerRouge disliked Vietnam and The KhmerRouge controlled Cambodia.
Vietnam disliked the KhmerRouge because Cambodia attacked Vietnam.
Vietnam attacked Cambodia because Cambodia attacked Vietnam.

Vietnam's army was larger than Cambodia's army.
Vietnam defeated Cambodia because Vietnam attacked Cambodia and Vietnam's army was larger than Cambodia's army.
Vietnam ousted the KhmerRouge because Vietnam defeated Cambodia.

China invaded Vietnam because Vietnam ousted the KhmerRouge.
Vietnam did not want China to invade Vietnam.
Vietnam's army impeded China because Vietnam did not want China to invade Vietnam and China invaded Vietnam.
China left Vietnam because Vietnam's army impeded China.
Vietnam defeated China because China left Vietnam.

Congo-Civil Conglict

Congo-Brazzaville is a country.
UPADS is an entity.
PCT is an entity.

Sassou-Nguesso was a person.
Sassou-Nguesso led the PCT.

Lissoube was a person.
Lissoube led UPADS.

The PCT possessed Congo-Brazzaville.
The people of Congo-Brazzaville disliked the PCT.

The PCT lost the election because the people of Congo-Brazzaville disliked the PCT.
The UPADS defeated the PCT because the PCT lost the election.

The PCT disliked UPADS because the UPADS defeated the PCT.
The PCT was corrupt.
The PCT wanted to possess Congo-Brazzaville.

The PCT created rebellion because the PCT disliked UPADS and the PCT was corrupt and the PCT wanted to possess Congo-Brazzaville.
The PCT attacked UPADS because the PCT created rebellion.
The PCT defeated UPADS because the PCT attacked UPADS.
The PCT possessed Congo-Brazzaville because the PCT defeated UPADS.

Cuba Bay of Pigs Invasion

Cuba is a country.
The Soviets are an entity.
The UnitedStates is a country.
The exiles of Cuba is a group.
Castro is a person.
Castro led Cuba.

The Soviets liked Castro.
Castro liked the Soviets.

The UnitedStates disliked Castro because Castro liked the Soviets.
The UnitedStates disliked Castro because the Soviets liked Castro.
The UnitedStates wanted to overthrow Castro because the UnitedStates disliked Castro.

The UnitedStates wanted the exiles of Cuba to invade Cuba because the UnitedStates disliked Castro.
The UnitedStates helped the exiles of Cuba because the UnitedStates wanted the exiles of Cuba to invade Cuba.
The exiles of Cuba disliked Castro.
The exiles of Cuba invaded Cuba because the UnitedStates wanted the exiles of Cuba to invaded Cuba and the exiles of Cuba disliked Castro.
The Cuba was stronger than the exiles of Cuba.

The Cuba defeated the exiles of Cuba because the exiles of Cuba invaded Cuba and the Cuba was stronger than the exiles of Cuba.

Czechoslovakia-Soviet Invasion

Czechoslovakia is a country.
USSR is a country.
The Slovaks are a people of Czechoslovakia.

Czechoslovakia was part of the SovietBloc.
The SovietBloc contained Czechoslovakia.
Czechoslovakia was communist.

USSR liked Czechoslovakia because Czechoslovakia was communist and Czechoslovakia was part of the SovietBloc.
The Slovaks were unhappy.

The Slovaks wanted reform because the Slovaks were unhappy.
Czechoslovakia began reform because the Slovaks wanted reform.
Czechoslovakia began to move from communism because Czechoslovakia began reform.
Czechoslovakia disobeyed USSR because Czechoslovakia began to move from communism.

USSR disliked Czechoslovakia because Czechoslovakia disobeyed USSR.
USSR was stronger than Czechoslovakia.
USSR invaded Czechoslovakia because USSR disliked Czechoslovakia beginning to move from communism.

USSR controlled Czechoslovakia because USSR invaded Czechoslovakia.
Czechoslovakia was communist because USSR controlled Czechoslovakia and USSR was communist.

Nigerian Civil War

Nigeria is a country.

The NigerianEast is a region.
The NigerianWest is a region.
The NigerianNorth is a region.

The NigerianEast was a region of Nigeria.
The NigerianWest was a region of Nigeria.
The NigerianNorth was a region of Nigeria.

Hausa-Fulani is an adjective.
Yoruba is an adjective.
Igbo is an adjective.

The NigerianEast's people were Igbo.
The NigerianWest's people were Yoruba.
The NigerianNorth's people were Hausa-Fulani.

The NigerianWest was a part of Nigeria.
The NigerianNorth was a part of Nigeria.

The NigerianNorth disliked the NigerianEast because the NigerianEast's people were Igbo.
The NigerianEast disliked the NigerianNorth because the NigerianNorth disliked the NigerianEast.
The NigerianWest liked the NigerianNorth.
The NigerianEast disliked the NigerianWest because the NigerianWest liked the NigerianNorth.
The NigerianEast left Nigeria because the NigerianEast disliked the NigerianNorth.
The NigerianEast left Nigeria because the NigerianEast disliked the NigerianWest.

Nigeria wanted the NigerianEast to stay at Nigeria.
Nigeria was stronger than the NigerianEast.

Nigeria attacked the NigerianEast because Nigeria wanted the NigerianEast to stay at Nigeria and the NigerianEast left Nigeria.
Nigeria defeated the NigerianEast because Nigeria attacked the NigerianEast and Nigeria was stronger than the NigerianEast.
Nigeria controlled the NigerianEast because Nigeria defeated the NigerianEast.
Nigeria forced the NigerianEast to join Nigeria because Nigeria wanted the NigerianEast to stay at Nigeria.
Nigeria forced the NigerianEast to join Nigeria because Nigeria controlled the NigerianEast.

Persian Gulf War

Iraq is a country.
Kuwait is a country.
UnitedStates is a country.
KuwaitsOil is a resource.

Iraq invaded Kuwait because Iraq wanted KuwaitsOil and Kuwait possesses KuwaitsOil.
Iraq was stronger than Kuwait.
Iraq defeated Kuwait because Iraq invaded Kuwait and Iraq was stronger than Kuwait.
Iraq possessed Kuwait because Iraq defeated Kuwait.

The UnitedStates invaded Iraq because the UnitedStates did not want Iraq to possess Kuwait and Iraq possessed Kuwait.
The UnitedStates defeated Iraq because the UnitedStates invaded Iraq and the UnitedStates was stronger than Iraq.
The UnitedStates possessed Iraq because the UnitedStates defeated Iraq.
The UnitedStates forced Iraq to leave from Kuwait because the UnitedStates possessed Iraq.

Romania and Ceausescu

Romania is a country.
Ceausescu was a person.
Ceausescu was the leader of Romania.
Ceausescu controlled Romania.
Romanians are people.

Ceausescu killed Romanians because Ceausescu was corrupt.
Romanians were angry because Ceausescu controlled Romania.

Romanians overthrew Ceausescu because Romanians defeated Ceausescu.
Romanians executed Ceausescu.
Romanians were happy.
Romanians were happy because Romanians executed Ceausescu.