Artificial Empathy:
Using Vector Space Modeling and Mixed Scope Alignment to Infer Emotional States of Characters in Stories

by

Ryan Cherian Alexander

S.B., Massachusetts Institute of Technology (2015)

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

June 2016

Author . . . .
Department of Electrical Engineering and Computer Science
May 20, 2016

Certified by . . . .
Patrick Henry Winston
Ford Professor of Artificial Intelligence and Computer Science
Thesis Supervisor

Accepted by . . . .
Christopher J. Terman
Chairman, Masters of Engineering Thesis Committee

Artificial Empathy: Using Vector Space Modeling and Mixed Scope Alignment to Infer Emotional States of Characters in Stories

by

Ryan Cherian Alexander

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2016, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Emotions greatly influence human cognition. Therefore, if we are to develop artificially intelligent programs that work closely with humans, we must ensure that they are capable of empathy. In an effort to realize the goal of emotionally aware programs, I created a multi-corpus informed vector space model to determine the emotions evoked by individual terms. I then combined that information with the semantic parse trees produced by the Genesis Story Understanding System to ascertain the emotions evoked by a single sentence. Additionally, I used the story aligner within Genesis to determine the emotions evoked by stories described over multiple sentences. My program can infer characters' emotional states based on their descriptions, the situations they are involved in, and the actions they perform. For instance, it infers that Alice is joyful from the sentence *"Alice wins an award"* and that James is probably experiencing sadness from the sentence *"James is lonely."* Additionally, the program can identify that Austin is likely surprised if *"Austin has to take a test"* and *"Austin doesn't know about the test."*

Thesis Supervisor: Patrick Henry Winston
Title: Ford Professor of Artificial Intelligence and Computer Science

Acknowledgments

I would like to thank:

My family for all their love, support, and prayers.

David for more than I could ever list here. You have been an amazing friend, teacher, and sounding board for all my years at MIT.

Yajit for encouraging and inspiring me. Your wisdom and kindness have been great blessings.

The Genesis group, especially Dylan, Jessica, and Josh for all their help and advice

Professor Szpakowicz and Dr. Aman for use of their blog data corpus

I would especially like to thank Professor Winston who inspired my passion for artificial intelligence from the very first day of 6.034 and gave me an opportunity to work in this fascinating field. I will always be grateful for your patience, guidance, and compassion.

Contents

List of Figures

Chapter 1

Introduction

In order to use a tool effectively, you must have a general understanding of how it works. Similarly, in order to collaborate with someone, you must have a general understanding of how they think. The difference is that when you are working with someone, they must also have an understanding of your thought process.

We use computers effectively every day, but in order to create artificially intelligent programs that are capable of working with us, they must possess an understanding of our cognitive abilities, which are greatly influenced by our emotions. Therefore, as artificially intelligent programs become more common and as we trust them with increasingly important tasks, it becomes imperative that these programs gain an appreciation of the emotions we feel and why we feel them.

Emotion is a quintessentially human concept, one that is extremely difficult to explain. A description cannot quite capture the agony of loss or the feeling of unbridled joy. But empathy can be taught. It is much easier to explain why someone might get scared and what they might do if they were frightened than it is to explain what being gripped by terror feels like.

It follows then, that in order to develop artificial intelligence, we must first develop artificial empathy. As a step towards this goal, I have developed a program within the Genesis Story Understanding System that is capable of empathizing with characters in stories. My approach blends the large-data, shallow analysis methodologies from previous emotion extraction studies with the small-data, deep analysis design of Genesis. This approach enables my program to operate in a manner analogous to the way humans empathize.

In this thesis you will find an overview of the Genesis System, a discussion of related studies, a detailed description of the program, affectionately named *Isabella*, and an analysis of the program's abilities.

Chapter 2

The Genesis System

Genesis is a story understanding system developed by CSAIL Professor Patrick Winston and his students. The system is, in part, a realization of Winston's hypotheses on the right way to tackle the challenge of creating systems that operate in a manner analogous to the way humans think.[14] In this chapter, I will present an overview of the system's functionality, acknowledge its shortcomings, and discuss Winston's principles, which have influenced the design of *Isabella*, my empathy program.

The primary goal of Genesis is to fully understand stories. This goes beyond being able to extract the explicit elements of a story. Genesis is able to apply common sense reasoning, infer missing explanations, determine character motivations, and identify concept patterns all while being able to explain its reasoning process. Furthermore, there are modules within the system that can generate propaganda [21], answer hypothetical questions, pick out the most important sentences in order to create a summary, and, with the addition of my empathy program, infer the emotional states of characters.

2.1 The START Parser

Before Genesis can perform the analysis necessary to accomplish these tasks, it must first parse the input story. This is handled by the START Parser, developed by Boris Katz, a principal research scientist at CSAIL. Katz's parser semantically analyzes the English input and exposes the structure and meaning of the wording, producing a semantic net instead of the solely grammatical parse tree produced by most other natural language parsers. START outputs ternary statements, or triples. These generally take the form:

(subject relationship object)

For example, the sentence *"Audrey is happy"* generates the following triples:

(audrey has_property happy)
(audrey is_proper yes)
(audrey has_number singular)
(is has_tense present)
(happy has_category adj)

These triples might seem like obvious implications, but in order to pass along that information to the program, it must be made explicit. One of START's strengths is its ability to extract a great deal of information from each sentence that a parse tree might not necessarily expose. In addition to part of speech tagging and tense identification, START performs a form of lemmatization that identifies the base form of the verb. This means we can tell when an action has occurred without needed to determine which form of the verb was used in the sentence. For example, *worried*, *worrying*, and *worries* would all be lemmatized to *worry*. To be thorough, the specific form used is also recorded in a separate triple.

2.2 The Inner Language of Genesis

Winston posits in his Inner Language Hypothesis that one of the main reasons human intellect far surpasses that of other primates is because we use an inner language to construct symbolic descriptions of situations and events. These descriptions enable storytelling, which is central to education and surrogate experience, which in turn influences culture. He also states in his Directed Perception Hypothesis that it is this inner language that "enables us to direct the resources of our perceptual systems to answer common-sense questions about real and imagined events".[13]

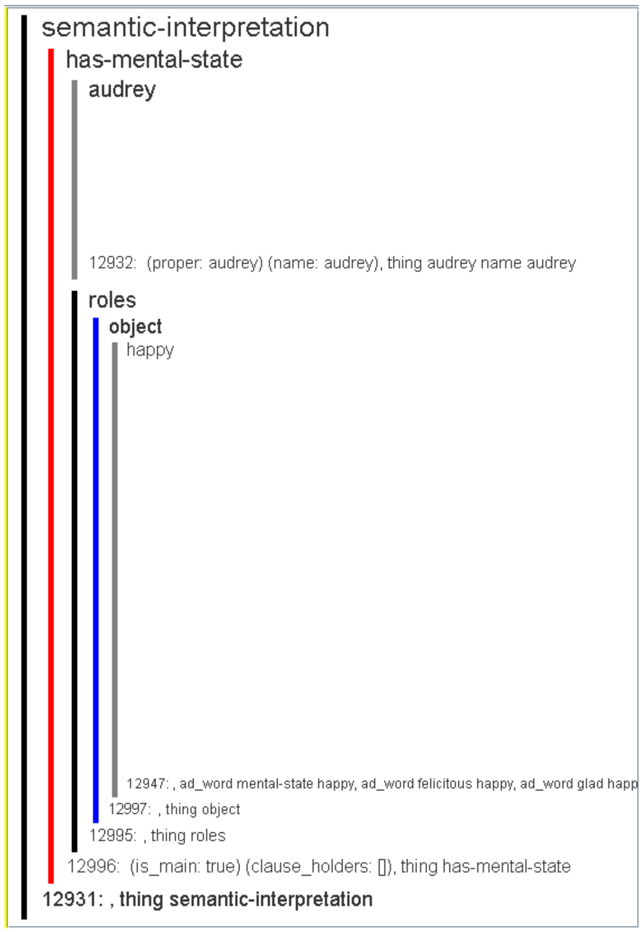
Therefore, in keeping with the overarching goal of creating computer programs that operate in a manner similar to the way humans think, the first step in the Genesis processing chain is to translate these triples into its own inner language. Most of this inner language is composed of role frames. A role frame is a representation of the information being conveyed in a sentence and typically consists of an actor, an action or property, and a set of optional entries for additional details. The role frames used in Genesis are described in terms of four classes:

Entity: A single object, place, person, or thing.
Relation: A connection between a subject and an object, representing a type of interaction between the two.
Function: A modifier that operates on a single subject.
Sequence: A collection of any number of entities, relations, functions, or other sequences.

Generally, the actor Entity will serve as the subject of the Relation, which will either be an action or some classification type. The remaining roles, including the object of the action and any modifiers, will be combined into a Sequence that serves as the object of the Relation.[15]

Figure 2-1 illustrates the Inner Language representation for the sentence *"Audrey is happy."* The black line represents a Sequence, red a Relation, gray an Entity, and blue a Function. All the information gleaned from the START parse is represented in this format.

Genesis can also hook into WordNet, a lexical database that groups words into "cognitive synonyms" which creates a network of semantic and lexical relationships. This allows us to acquire word stems, synonyms, or lemmas should we need them for semantic analysis or comparison.[7]



```
semantic-interpretation
  has-mental-state
    audrey
    12932:  (proper: audrey) (name: audrey), thing audrey name audrey
    roles
      object
        happy
        12947: , ad_word mental-state happy, ad_word felicitous happy, ad_word glad happ
      12997:  , thing object
    12995:  , thing roles
  12996:  (is_main: true) (clause_holders: []), thing has-mental-state
12931: , thing semantic-interpretation
```

Figure 2-1: An Example of the Inner Language of Genesis

2.3 Connections and Concepts

Genesis can identify various types of causal connections that are explicitly stated in a story. Some of these connections are direct links, such as *Macbeth wants to kill Duncan* ***because*** *Macbeth wants to be king.* Other connections are stated but not fully described, such as *Macbeth's murdering Duncan* ***leads to*** *Macduff fleeing.*

We then define concept patterns in terms of these connections which allows Genesis to determine when a story exhibits a particular concept. Examples of concepts the system understands include Revenge, Pyrrhic Victory, Suicide, Success, Answered Prayer, and Teaching a Lesson. Most concepts are specified using **leads to** expressions. For example, Revenge is described as *personA's harming personB leads to personB's harming personA*. Note that these concept patterns are expressed in English, not code. The system translates the rules from English into its Inner Language for use in analysis.[16] See Figure 2-2 for the Inner Language representation of the Revenge concept pattern.

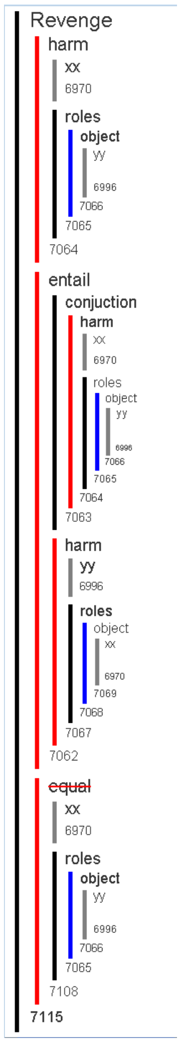


Figure 2-2: The Revenge Concept Pattern

2.4 Rules and Common Sense Reasoning

Genesis also uses a rules-based analysis system to glean additional information from the story. These rules are of the form: *IF antecedent, THEN consequent.* When the system sees a specific pattern in a story, the rule with the corresponding antecedent will fire and its consequent will be added to the list of assertions the system knows to be true, which, before any rules fire, would consist of the statements in the story.

We use these rules as an artificial commonsense apparatus. Now, human readers possess a vast set of commonsense reasoning that we automatically apply and in most cases, we do so without even thinking about it. A great deal of the additional information gained from a commonsense understanding of our world seems so basic and trivial that most authors would never bother to directly include it in the story. However, this extra information is still required for a true understanding of the story.

For example, if I tell you that *James was murdered* then you immediately know that James is dead. However, his death was never specifically mentioned and if you built a story understanding system that only uses the explicit information, it would be completely unaware of the fact that James is no longer alive.

Genesis has a set of commonsense rules that covers a wide variety of important, albeit sometimes obvious, facts including *If X is murdered, then X is dead*. Most of these rules are if-then relations, but others are more complicated. When taken as a whole, these commonsense rules represent the system's baseline understanding of the world, outside of any particular story. We can alter these prior beliefs and change the set of biases and cultural reasoning that Genesis possesses.

All of this information is combined into an Elaboration Graph, seen in Figure 2-3. The white boxes are explicit statements from the story and the elements in yellow were added by the system, extra information added as a result of the common sense rules. The connections are causal links, which are either directly stated in the story or added as the result of the rules that seek to connect related events.

2.5 Story Alignment

One of the most powerful features within the Genesis system is the story aligner, which can calculate how similar two stories are. As is the case with most Genesis functions, this alignment occurs at the concept level. Essentially, the system looks at the component elements of two stories. For every pair of elements that have the same classifications (i.e. an Object Entity or a Harm Relation), the score increases.

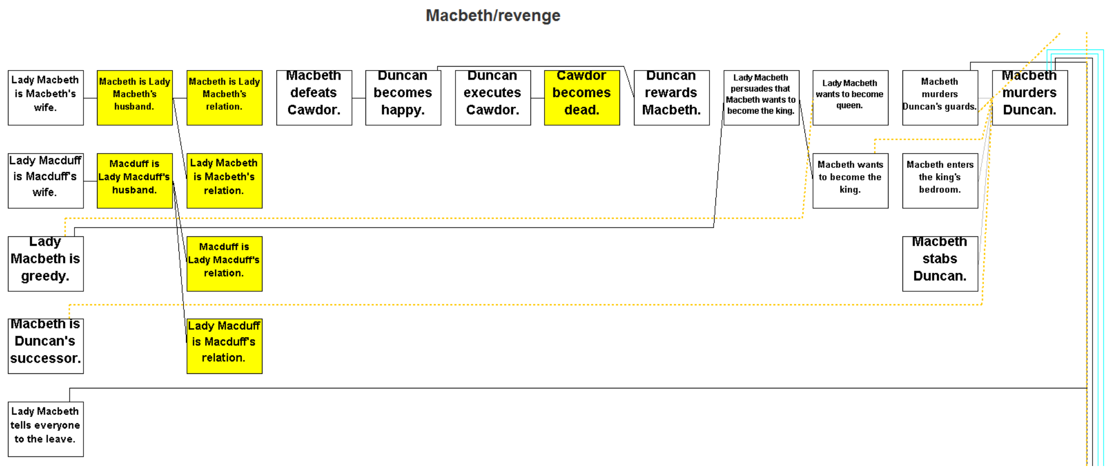
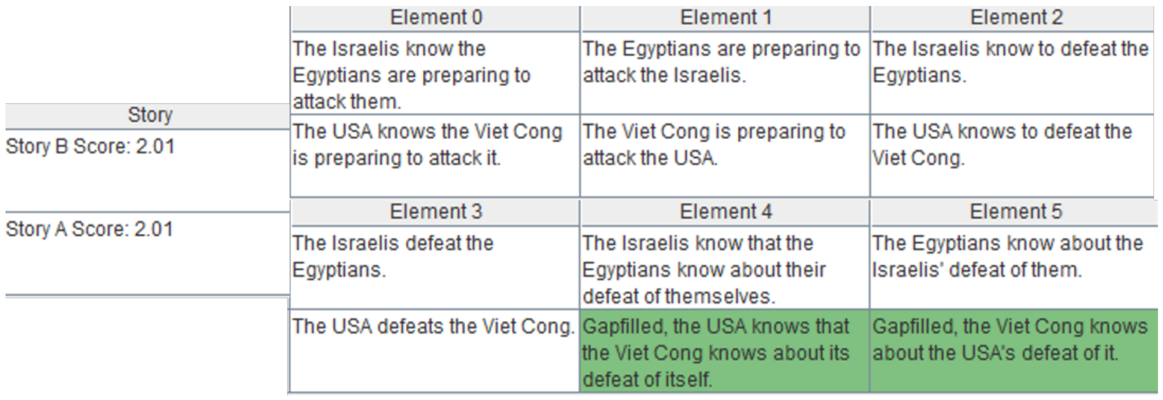

Macbeth/revenge
Lady Macbeth is Macbeth's wife.
Macbeth is Lady Macbeth's husband.
Macbeth is Lady Macbeth's relation.
Macbeth defeats Cawdor.
Duncan becomes happy.
Duncan executes Cawdor.
Cawdor becomes dead.
Duncan rewards Macbeth.
Lady Macbeth persuades that Macbeth wants to become the king.
Lady Macbeth wants to become queen.
Macbeth murders Duncan's guards.
Macbeth murders Duncan.
Lady Macduff is Macduff's wife.
Macduff is Lady Macduff's husband.
Lady Macbeth is Macbeth's relation.
Macbeth wants to become the king.
Macbeth enters the king's bedroom.
Lady Macbeth is greedy.
Macduff is Lady Macduff's relation.
Macbeth stabs Duncan.
Macbeth is Duncan's successor.
Lady Macduff is Macduff's relation.
Lady Macbeth tells everyone to the leave.

Figure 2-3: An Elaboration Graph

For every mismatch, a penalty is paid. The system can also identify gaps, which incur a lesser penalty. Gaps occur when an event or object is present in one story, but not present in the other. If the stories turn out to be very similar, the program can automatically fill those gaps by using the details from the original story and the pattern of the aligned story. See Figure 2-4 for an example of this process. The green elements are gap-filled.[11]



Story	Element 0	Element 1	Element 2	Element 3	Element 4	Element 5
Story B Score: 2.01	The Israelis know the Egyptians are preparing to attack them.	The Egyptians are preparing to attack the Israelis.	The Israelis know to defeat the Egyptians.	The Israelis defeat the Egyptians.	The Israelis know that the Egyptians know about their defeat of themselves.	The Egyptians know about the Israelis' defeat of them.
Story A Score: 2.01	The USA knows the Viet Cong is preparing to attack it.	The Viet Cong is preparing to attack the USA.	The USA knows to defeat the Viet Cong.	The USA defeats the Viet Cong.	🟩 Gapfilled, the USA knows that the Viet Cong knows about its defeat of itself.	🟩 Gapfilled, the Viet Cong knows about the USA's defeat of it.

Figure 2-4: Story Alignment

2.6 Design Philosophies

2.6.1 START vs Conventional NLP

The START parser gives Genesis a lot more information to work with than a purely grammatical parser. But there is a trade-off to consider. Because of the increased difficulty in building a semantic parse, START is unable to parse certain grammatical patterns. Most of the time, START fails because the input sentence is simply too complex, but this is not always the case. There are other patterns that might not seem like they would cause issues, but confuse START and are unable to be parsed. For example, START cannot parse sentences where two adverbs are used to describe a single verb, such as in the sentence *"She quickly and quietly ran."* You would need to split that sentence up into two separate statements. Unfortunately, this means Genesis will not be able to handle stories that have not been manually rewritten to be STARTparsable. Note that START either fails or produces a result that it is fully confident in. There are no approximations or best guess parses.

Now, conventional Natural Language Processing (NLP) relies on parsers that will make a best-estimate parse of any input, including sentences that are not grammatically valid. They need these parsers because NLP programs also rely on machine learning algorithms, virtually all of which require large amounts of training data. As long as the parser can keep up with the amount of input data, it does not matter if some of the results are incorrect. As long as the probability of error is below a tolerance factor, the results will still be useful.

The primary goal of most NLP programs is to identify patterns of word usage in large amounts of text so that groups of words can be formed. It is a large-data, shallow analysis design. Grouping can be done to identify words of similar meaning, to find words that share the same sentiment (positive or negative mood), or to organize words by topic. While these algorithms benefit from using data that has been labeled by human annotators, it is not necessary. For more on this idea, see the distinction between Supervised and Unsupervised Machine Learning.

However, the Genesis System is designed to extract as much information as possible from every sentence in a story. The goal is to be able to learn from a single example, as humans are capable of doing. In accordance with this, the main mechanism in Genesis is a rules-based system, as discussed previously. We provide the rules to the system in generic terms and expressed in English which means that this process is not unlike the way parents teach their small children. Now obviously, children also do a great deal of learning on their own, but a significant amount of information is directly given to them by their parents and teachers and it is this form of instruction that we are seeking to emulate. Because a single rule is applicable across all stories, there is no need for training data. Genesis does not require hundreds of examples of revenge stories in order to understand the concept. So we must use a parser that will generate complete, correct results, and accept the need for significant human input in our small-data, deep understanding design.

2.6.2 Why Stories?

Throughout our discussion, we have proceeded under the implicit assumption that stories are in fact worth understanding. The reasoning for this assumption comes from Winston's hypotheses. If an inner language is indeed the hallmark of human intellect, it must be tied to our problem solving abilities. It follows then that the inner language functions as the common link between our imagination, which allows us to propose possible solutions, our sensory inputs, and our understanding of the problem's constraints. Because the primary purpose of the inner language is to enable description which in turn enables story-telling, stories must also be imperative to our cognitive abilities.[14] These ideas are emphasized in the last two of Winston's four major hypotheses. The Strong Story Hypothesis states that "Our inner language enables us to tell, understand, and recombine stories, and those abilities distinguish our intelligence from that of other primates" and the Social Animal Hypothesis states that humans' "social nature amplifies the value of story understanding and directed perception."[13] It is on the basis of these hypotheses that we proceed with a focus on story understanding, believing that if we are to take artificial intelligence to the next level, we must first understand and explore the nature of human intelligence.

Chapter 3

Related Work

In this chapter I will present an overview of the methodologies and datasets that have been used in other studies for extracting emotion from text. The problem of inferring emotion is particularly challenging and so I will also discuss some of the common techniques and simplifications to make the problem tractable.

3.1 An Emotional Subset

English is an incredibly complex language with a massive vocabulary. Virtually every word has multiple meanings and readers must often rely on context clues, the grammatical structure of the surrounding sentence, and cultural connotations to disambiguate the possible definitions. Attempting to work with the set of all emotion words would be extremely difficult. It is therefore necessary to select some subset of emotions to work with. There are two subsets that are most often used in emotion extraction studies are Plutchik's Eight and Ekman's Six.

3.1.1 Plutchik's Wheel Of Emotions

In 1980, professor and psychologist Robert Plutchik created a theory of emotion centered on eight primary emotions: **Anger**, **Fear**, **Sadness**, **Disgust**, **Surprise**, **Joy**, **Anticipation**, and **Trust**. Plutchik argued that the development of these emotions is tied to the reproductive fitness of an animal and believed that they could trigger instinctual reactions. [17] He also created a Wheel of Emotions illustrating the varying intensities and nuances of each of the eight emotions, as seen in Figure 3-1. [18]

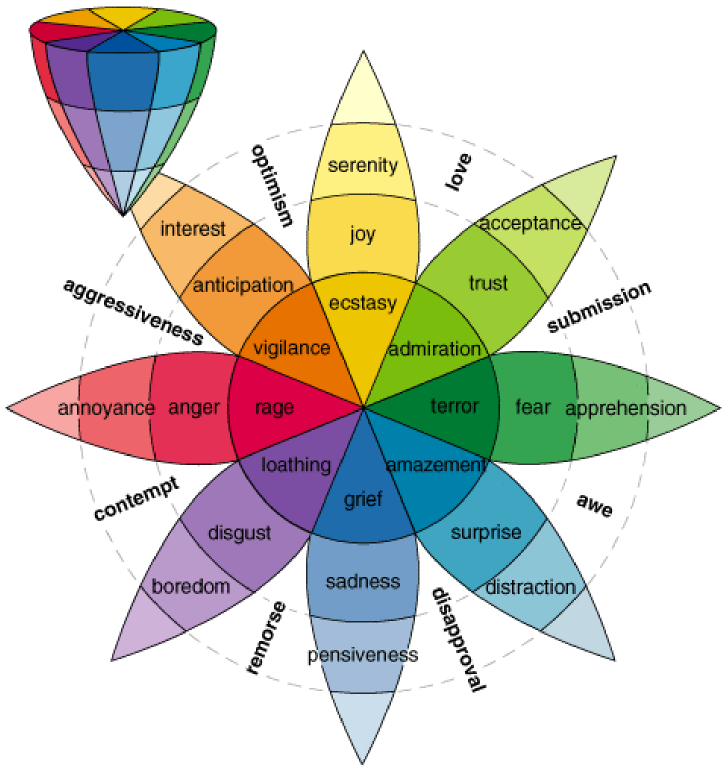


Figure 3-1: Plutchik's Wheel Of Emotions

3.1.2 Ekman's Basic Emotions

In 1992, Paul Ekman, a psychologist working at the University of California, San Francisco, developed an argument for six basic emotions: **Anger**, **Fear**, **Sadness**, **Disgust**, **Surprise**, and **Joy**. These six were a subset of Plutchik's eight, but were determined by studying human facial expressions in a wide range of cultures. Ekman was searching for a set of universal emotions that could be recognized by anyone, regardless of linguistic, cultural, or environmental differences. The primary objection to Ekman's work was that it would not be possible to confirm that any emotion was truly universal due to the widespread proliferation of media and cross-cultural communication. However, Ekman silenced these arguments when he found tribal natives of Papua, New Guinea who, despite being preliterate and isolated from the rest of the world, could still demonstrate the six emotions. See Figure 3-2 for some examples. [12]

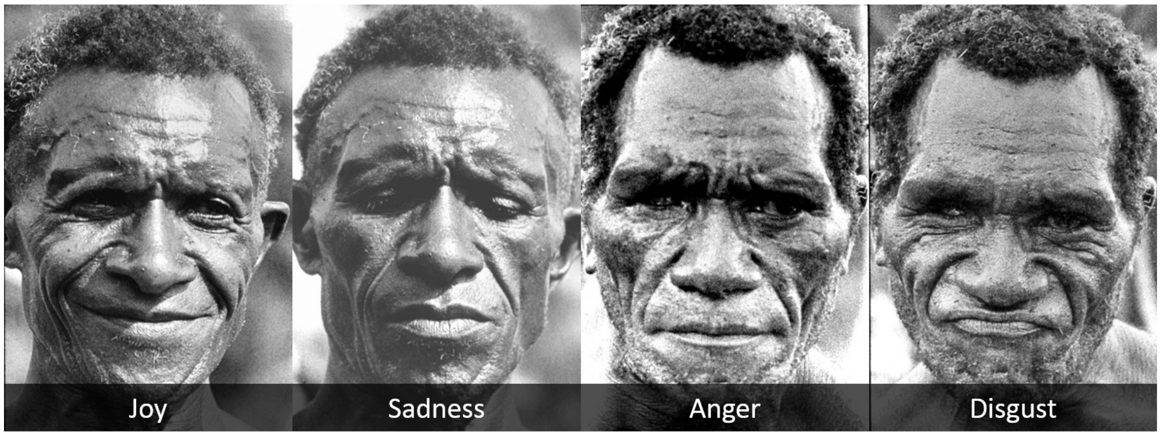


Figure 3-2: Examples of Facial Expressions Demonstrating some of Ekman's Emotions

3.2 Datasets

Several studies have been conducted in order to create annotated datasets to inform emotion extraction programs. I will discuss four of the most commonly used ones. The first two sources (WordNet Affect and emoLex) were designed to determine which emotions, if any, a single word might evoke, regardless of context. The last two sources (the Annotated Blog Corpus and the Fairy Tale Corpus) only consider how the word is used in context without directly considering its meaning.

3.2.1 WordNet Affect Database

Strapparava et al. identified several seed words that evoked an emotion by definition. Using WordNet, they mapped the related words, alternate part of speech forms, and synonyms for each of the seed words and marked all of them as evoking the same emotion. The goal was to create a database of words that would evoke a certain emotion regardless of context and based purely on the meaning of the word. For example, *angry*, *anger*, *mad*, *hateful*, *hostile*, *aggravate*, and *resent* are some of the words in the **Anger** category. [4]

3.2.2 Emotional Lexicon (emoLex)

One of the issues with creating an annotated dataset, especially when attempting to classify something as subjective as emotional properties, is that several annotators are needed to classify each entry. Multiple annotators reduce the risk that the dataset will be a reflection of a single opinion.

Mohammad and Turney addressed this issue by using Amazon's Mechanical Turk to get input from a large number of human annotators via survey questions. The team got their target words from several thesauri and asked participants to determine if a target word evoked a certain emotion, and if so, with what intensity. Responses were consolidated and each target word was assigned to a subset of Plutchik's eight emotions and/or to a positive/negative tag. It is worth noting that MTurk participants are paid a reward for completing questions but are not screened beforehand. Therefore, the team specifically designed questions to identify users who were simply guessing or answering randomly so that those responses could be thrown out.[22]

3.2.3 Annotated Blog Corpus

Aman and Szpakowicz data mined publicly visible blog posts and had two annotators, working separately, tag each sentence in the dataset with an emotion from Ekman's basic six or marked the sentence as not evoking any emotion. They chose blog posts as a source of data because they wanted the emotion-rich data typically found in personal stories and descriptive commentary. [20] [19]

3.2.4 Fairy Tale Corpus

Alm et al. annotated over 1500 sentences from a collection of fairy tales written by the Grimm Brothers, H.C. Andersen, and B. Potter. Each sentence was seen by two annotators working separately to avoid bias. Sentences were tagged as evoking no emotion or with one of Ekman's basic six, with the exception of Surprise which was split into two categories - Positively Surprised or Negatively Surprised. It's interesting to note that the annotators only agreed 45-65% of the time, depending on the particular combination of annotators. This was expected, given the subjective nature of the task, and confirms the idea that multiple annotators should always be used to create such a database. [2]

3.3 Methodologies

3.3.1 Sentiment Analysis

Several studies have focused only on classifying text as positive or negative. This sentiment analysis does not provide much information about the emotional state of characters, but research in this area has identified several methodologies that have some success in identifying sets of words that invoke certain moods. The primary tools used in these studies are Naive Bayes Models or Support Vector Machines (SVMs). Studies have shown accuracy rates in the 75% - 85% range when compared to human annotations on several data sets, most notably a corpus of movie reviews. However, when these tools were used in attempts to classify text with emotional labels, their accuracy dropped significantly. These results support the idea that these models alone are good starting points, but are not complete solutions. [10]

Wu, Chuang, and Lin achieved some success by combining positive (happiness) / negative (sadness) labeling with manually created rules in order to inform a probabilistic separable mixture model. Of course, the team was limited to classifying the situations their rules could cover. [5] Furthermore, the problem with using rules to define what emotions should be evoked is that not everyone will agree on what the rules should be. After all, emotional responses, by their very definition, are subjective and vary from person to person.

3.3.2 Keyword Based Approaches

The idea behind keyword based emotional extraction programs is to identify seed words that are very likely to evoke certain emotions and then determine how frequently other terms co-occur with those seed words in order to build a model that can classify query documents. It's these modeling techniques that vary from one implementation to the next.

Danisman and Alpkocak chose a Vector Space Model (VSM) implementation in their keyword based approach. In a VSM, each document within the corpus is represented as a weighted vector with each dimension corresponding to a unique word. The weight was calculated using the Term Frequency - Inverse Document Frequency (tf-idf) scheme which essentially determines how important a word is to a particular document. If the word occurs frequently within the document, then it is considered to be more important. However, if the word occurs frequently throughout all documents, then it is not as specific to a single document and the importance factor is reduced. After the vectors for each document have been computed, all the document vectors within a single label can be averaged together in order to get a vector for the label. To classify and unknown document, the cosine difference between the query vector and the label vector can be computed to determine how similar the two are. The team used the International Survey on Emotional Antecedents and Reactions (ISEAR) database as their annotated corpus. ISEAR consists of several short stories and sentences, typically in first person, that were specifically written in response to questions about experiencing a particular emotion. The possible emotions were joy, fear, anger, sadness, disgust, shame, and guilt. [23]

3.3.3 Grammar Based Models

Neviarouskaya et al. extended a keyword based approach (informed by the WordNet Affect Database) by taking the grammatical structure of the sentence into consideration. Their program began at the word level and then moved higher in scope, considering phrases and then the entire sentence. The team developed rules for grammatical patterns and used different scoring systems for each stage of analysis, eventually generating a score for the complete sentence. [1]

Keshtkar et al. developed a sliding window algorithm designed to identify phrases that evoke emotions. Their program also used the keyword approach to get emotional quotients for seed words, but also kept track of frequently occurring words in close proximity to the seed words.[6] This group used a combination of sources including the Blog Corpus and Fairy Tale dataset discussed earlier. The team also used the Text Affect Dataset which is composed of annotated headlines developed by Strapparava et al.[3] and the LiveJournal blog dataset collected by Mishne [8], which consists of posts whose authors have tagged them with the mood or emotion expressed within.

3.4 Summary

Each of these studies attempted to solve the emotion extraction problem with a stand-alone solution. Most of them had to construct their own grammatical models or hand-craft rules before running various machine learning algorithms to compute the classifications. It stands to reason then, that an approach that utilizes an existing NLP system, complete with a host of analysis modules, could more easily achieve success. Furthermore, none of these studies attempt to tie the emotions to characters, solely focusing on extracting the overall emotional sentiment expressed in the text. I believe that my program can offer a useful and innovative solution in this space by building on these past studies and leveraging the power of the Genesis system. As Professor Winston says, “You can do it, only you can do it, but you can’t do it alone.”

Chapter 4

The *Isabella* Program

The vision for my empathy program, affectionately named *Isabella*, was to create a module within Genesis that would allow the system to infer the emotional state of characters in stories. I wanted to build on the work done by previous studies, but ultimately develop a process that is more similar to the way humans empathize. Determining what characters are likely feeling as well as why they are feeling those emotions will afford deeper insight into character motivations. The ability to emphasize can also be used to inform additional rules and recognize more concept patterns, leading to improved story understanding capabilities.

4.1 Overview

Isabella relies on a multi-corpus trained vector space model that calculates emotional alignment at the individual word scope. That information is combined with the semantic relationships produced by START and Genesis to determine the emotions invoked by a single sentence. Additionally, Isabella uses the story alignment module within Genesis to determine emotions evoked over multiple sentences. See Figure 4-1 for an outline of the process.

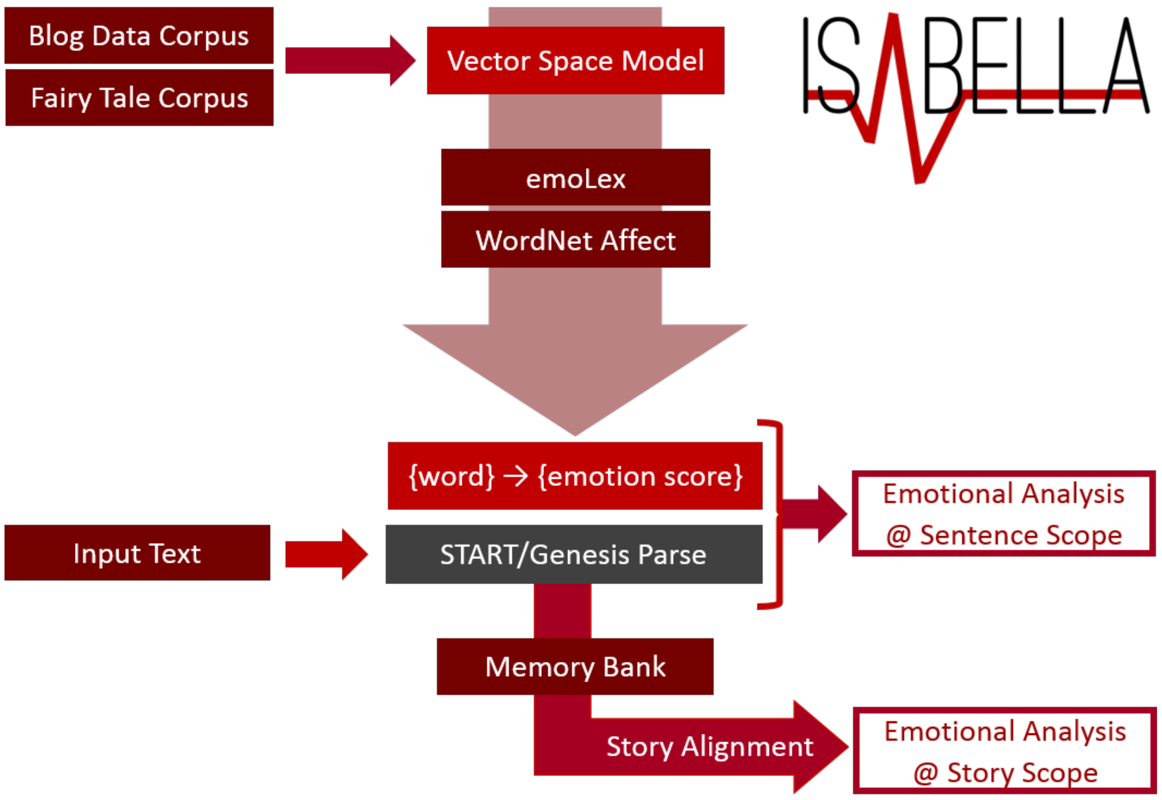


Figure 4-1: An Overview of Isabella

4.2 Datasources

My program is informed by several of the datasets discussed in the previous chapter, including WordNet Affect (WNA), emoLex, the Annotated Blog Corpus, and the Fairy Tale Corpus.[1]

I also used a database containing the 10,000 most commonly used words as curated by Josh Kaufman. Kaufman produced the dataset by abbreviating Peter Novig's analysis of Google's n-gram based database, which contains the usage frequencies for over 1 trillion English words.[9] I used WordNet to determine what the potential parts of speech were for each of the 10,000 words and created four separate lists, which I will refer to as `commonSets`, of the most frequently used Verbs, Nouns, Adjectives, and Adverbs. Note that some words are in multiple lists. For example, the word

[1]All data sources were acquired directly from their authors with the exception of the Fairy Tale Corpus which was obtained from Bogdan Neacsa's Github at https://github.com/bogdanneacsa/tts-master

review would be in both the Noun list and the Verb list. I made sure to use the lemmatizer function from the Natural Language Toolkit (NTLK) Python library to get the base form of each word for the desired part of speech. This way, I could group the different forms of a single word into one item. For example, the verbs *carried*, *carrying*, and *carry* are all lemmatized to *carry*.

4.3 Term Alignment

Words can have multiple definitions and knowing the part of speech (PoS) tells readers how the word was used and reduces the ambiguity in meaning. Therefore, I want to keep track of both the word and its part of speech as a single entity, which I will refer to as a **term**. This will allow me to keep separate weightings for the different PoS usages of a word. For example, (worried, the past tense verb) is distinct from (worried, the adjective).

The goal of this term alignment is to assign each term a six-dimensional vector of confidence scores, with each dimension representing one of Ekman's six basic emotions: *anger*, *sadness*, *joy*, *surprise*, *disgust*, and *fear*. I compute these term vectors offline for Isabella to use during her analysis.

4.3.1 WNA and emoLex

WNA and emoLex are simply lists of terms, with each term = [word, PoS] being associated with some subset of the six emotions. So the score of emotion e for term k is simply:

$S_{k,e}$ = uniform distribution over the number of emotions associated with the term

4.3.2 Fairy Tale and Blog Corpora

The corpora both consist of a set of sentences with each sentence annotated with two separate tags. In order to calculate the $S_{k,e}$ values for these datasets, I had to process the data and create a vector space model using the following steps:

Step 1. PoS tag the sentence using the natural language toolkit in Python.

$$Term\ referring\ to\ Word\ x\ as\ used\ in\ Sentence\ i = t_{xi} = [word_x, PoS_(x, i)]$$

Step 2. Remove all the terms that are not some form of Verb, Noun, Adjective, or Adverb.

I don't need to consider the other parts of speech, such as prepositions and conjunctions, as they are too common to be specific to any emotion. The sentence is now a list of terms:

$$Sentence\ i = [t_1, t_2, ..., t_n]$$

Step 3. Attempt to map words to the set of the most commonly used words.

When I encounter a term that is not in the `commonSet` corresponding to its PoS, I attempt to find closely related terms that are in the `commonSet` and use those instead. WordNet allows me to do this by giving me a term's hypernyms. A hypernym is a broader category that the query term is a member of that shares the the same PoS as the target term. While synonyms are usually highly context-dependent, hypernyms are based on the word's definition alone, which is why I use it in my mapping process. For example, the term (chrysanthemum, noun) is not in the Noun `commonSet`, but its hypernym (flower, noun) is, and I can map the two without a loss in generality.

If this mapping fails, i.e. I encounter the situation where neither the term nor any of its hypernyms are in the `commonSet`, then I simply keep the term as is, acknowledging that it must be a unique word.

Now, a word can have many hypernyms that are in the `commonSet` and I really don't have a deterministic way to select which one to use. Therefore, I will use all of them, but add in a normalizing weighting factor. If the original term was in the `commonSet` to begin with or the mapping fails entirely, this factor will simply be 1. If the mapping is necessary and successful, each of the hypernyms that replace the original term will be weighted by a factor of $\frac{1}{number\ of\ hypernym\ replacements}$. If the same term appears multiple times in a sentence, its hypernym factors will be computed separately and then summed. I will refer to this hypernym factor as h and the sum of all the h weights in a sentence will be equal to the number of terms in that sentence. That is,

$$\sum_{j=1}^{t_n} h_j = n$$

where h_j is the hypernym factor of term t_j

Step 4. Create the Vector Space Model

My goal is to take every term in the corpus and represent it as a six-dimensional vector, with each dimension corresponding to an emotion. To do this, I construct a basis out of Ekman's six, and utilize a weighting scheme to place each term in this space. Once placed, a term's coordinates will essentially reflect its propensity to evoke each of the six emotions.

The general idea for the weighting scheme I desire is as follows: if I often see a term associated with emotion e, but not associated with any other emotions, then I can say that term is very specific to emotion e.

Fortunately, there is a metric that gives me precisely this - the term frequency - inverse document frequency (tf-idf) weighting scheme. As mentioned in Chapter Three, a word's tf-idf score increases with the frequency of that word within a specific document, but decreases proportionally to the rate the word appears in the entire corpus, which corrects for words that are simply very common, regardless of the document. A few notes:

- For these corpora, I treat each tagged sentence as a document
- Each sentence has two tags, one per annotator, so I represent them separately. That is, I will have (sentenceX, tag1) and (sentenceX, tag2). This has the nice effect of double weighting each sentence where there was a consensus between the two annotators.

I can now translate the sentence into a vector of weights.

$$sentence\ vector\ s_i =< w_1, w_2, ..., w_n >$$

I determine those weights by computing the tf-idf formula. However, standard tf-idf does not discriminate between parts of speech and does not involve the hypernym factor. [23] Therefore, I have altered the frequency term to correct for these extra considerations

$$w_{k,i} = \frac{f_{k,i} \cdot \log \frac{N}{n_k}}{\sqrt{\sum_{k=1}^{n} (f_{k,i})^2 (\log \frac{N}{n_k})^2}}$$

where:

$w_{k,i}$ = weight of term k in sentence i

N = total number of sentences

n_k = number of sentences that contain term k

and

$f_{k,i}$ = frequency of term k = $\frac{h_k}{\sum h_j}\ \forall\ t_j\ with\ the\ same\ PoS\ as\ t_k\ in\ sentence\ i$

I can then normalize to determine how important term k is to emotion e.

$$Score(t_k, Emotion\ e) = S_{k,e} = \frac{1}{|M_e|} \sum_{j \epsilon M_e} w_{k,j}$$

where:

$$M_e = \{all\ sentences\ tagged\ with\ Emotion\ e\}$$

Now there are some slight differences between the Fairy Tale dataset and the Blog dataset that we need to discuss.

Each sentence in the Fairy Tale corpus was tagged either with No Emotion or one of the six basic emotions, with the exception of Surprise which was split into two categories - Positively Surprised or Negatively Surprised. However, in order to maintain the use of Ekman's six as my emotional subspace, I simply combined both the surprised categories back into one tag. I computed the No Emotion category in the same way I computed the Emotion categories because I wanted tf-idf to identify the words that were mostly indicative of neutral emotion, which, by the definition of the weighting scheme, would reduce the score of those words in the emotional categories.

The Blog Data corpus was annotated with Ekman's six or No Emotion, but also contained some additional information. Along with the two emotion tags, the annotators had identified which words within the sentence had evoked the emotion. This means I could throw out the other words in the sentence because I know they did not influence the emotional tag. Because I am now only dealing with words that I know have some emotional value, I don't need to consider sentences marked No Emotion because I do not need to calculate a neutral tf-idf category.

4.3.3 Combining Scores from the Four Data Sources

Now that I have a $S_{k,e}$ for each data source z, I can multiply each score by the source's reliability factor r_z and sum the result.

$$Overall\ Score\ of\ Term\ k\ for\ Emotion\ e = E_{k,e} = \sum_{z\ \epsilon\ data\ sources} f_z \cdot S_{z,k,e}$$

I wanted to weight the context-free data sources more highly, as the actual definition is more important than occurrence correlation. Of all the sources, WNA contained only words whose dictionary definitions tied them directly to the emotional classes. emoLex was also context free, but based on survey results. Both blog corpora were annotated in virtually identical ways, so I weighted them equally. However, I did not want to stray too far from equally weighting the sources so I settled on the following:

$$\text{WNA reliability factor} = 0.35$$

$$\text{emoLex reliability factor} = 0.25$$

Blog Corpus reliability factor = 0.2

Fairy Tale Corpus reliability factor = 0.2

These values generated good experimental results, but can always be altered.

Finally, I have the six dimensional vector for term k which can be expressed as:

$$[E_{k,joy}, E_{k,anger}, E_{k,sadness}, E_{k,fear}, E_{k,disgust}, E_{k,surprise}]$$

4.4 Sentence Alignment

Many other studies treat a sentence like a bag of words, which reduces the complexity of the analysis. However, because I am attempting to achieve a relatively deep understanding and not seeking to analyze large quantities of text, I created a more computationally intensive algorithm to serve as Isabella's thought process.

4.4.1 Character Identification

As discussed previously, Isabella's goal is to empathize with characters in stories. Therefore, when performing emotional alignment at the sentence scope, she must consider what characters are involved, if any, and what their role is. In order to do this, Isabella must first know who the characters are, which means she needs some information found at the story-scope.

When Genesis reads a story, it looks for certain phrases, referred to as idioms, and treats them as special cases. The idiom `XX is a person` informs Genesis that `XX` is a character and any subsequent parses are aware of this and will PoS tag `XX` as a proper noun. Genesis will now also be aware that rules and characteristics that are exclusive to people, such as having a mental state, can be satisfied by `XX`. Isabella relies on Genesis which means that she will consider only the characters identified by this idiom to be people of interest.

4.4.2 Genesis Parse Trees

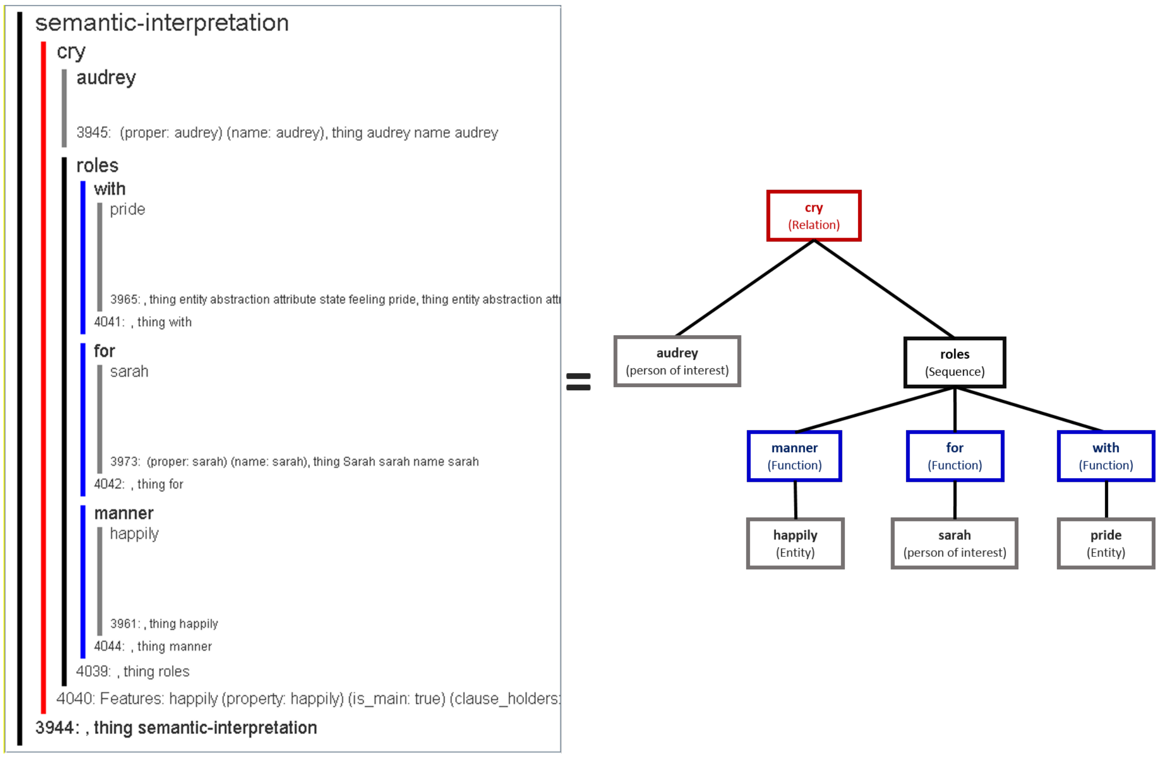
When given a sentence, Genesis will identify the subject of the sentence—be it the performer of some action, the target of a direct description, or the entity that is being something. It then ties this subject to an object via a Relation. The Relation can be a verb, a linking verb, or it can be a special type. The special cases include `has_mental_state` and `has_property` which both indicates a direct connection to a description.

The object of the Relation can be equivalent to the grammatical object of the sentence, but can also include any modifiers on the Relation. For example, when the object of a Relation is a Function `manner`, the subject of that function is an adverb that modifies the verb in the Relation. Regarding the other parts of speech, nouns are, as expected, represented as Entitles. Adjectives are expressed as properties either tied directly to Entities or as the object of the special case Relations mentioned above. This structure ensures that child nodes collectively modify their parents, a point that the scoring algorithm described in the following section addresses.

4.4.3 Design Choices

Isabella is primarily looking for the case in which a person of interest is the subject of a Relation because this means that sentence might have some information as to the emotional state of that character. You can have a person of interest serve as the object of a Relation, but Isabella knows that this generally does not say anything about that person. For example, in the sentence *"Ryan hates Tom"*, we know that Ryan is likely experiencing some amount of anger, but we have no information about Tom. Tom could be completely unaware of the situation, he could be quite fond of Ryan, he could hate Ryan back, etc. But we do not know for sure, so we cannot empathize with him and neither will Isabella.

As discussed previously, a story is a Sequence of Relations, Functions, and other Sequences and Genesis will organize this information into a semantic parse tree with each node representing a term or Genesis type. Each sentence of the story is a subtree and will be evaluated separately. See Figure 4-2 for an example of a sentence subtree.

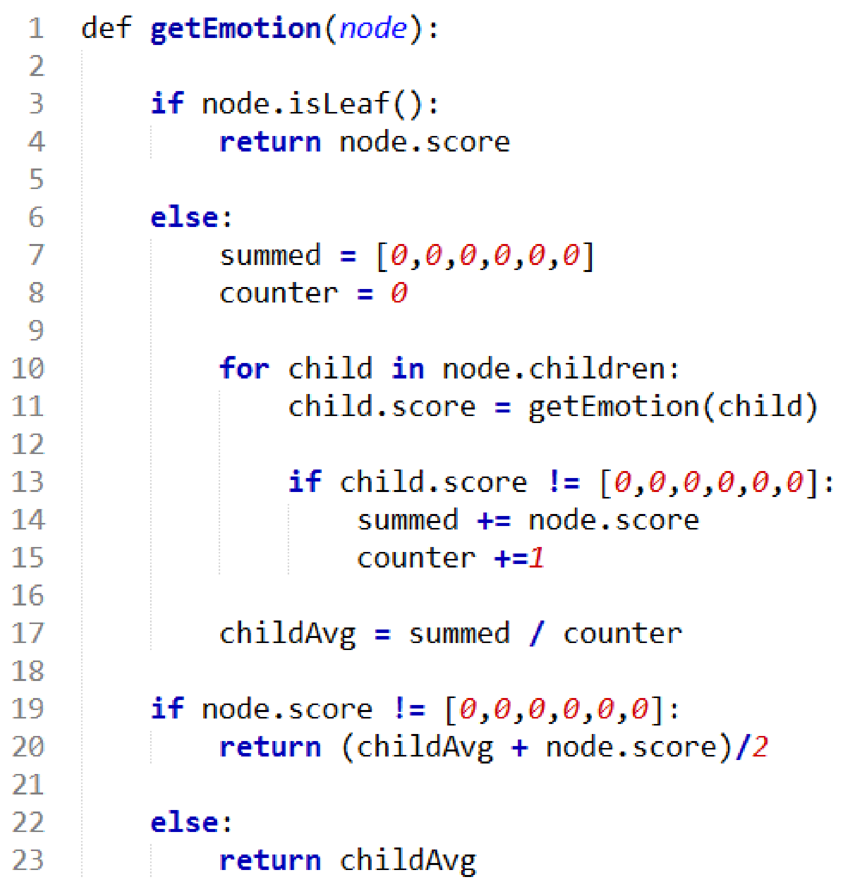


```
semantic-interpretation
  cry
    audrey
      3945:  (proper: audrey) (name: audrey), thing audrey name audrey
    roles
      with
        pride
          3965: , thing entity abstraction attribute state feeling pride, thing entity abstraction attr
        4041:  , thing with
      for
        sarah
          3973:  (proper: sarah) (name: sarah), thing Sarah sarah name sarah
        4042:  , thing for
      manner
        happily
          3961: , thing happily
        4044:  , thing manner
      4039:  , thing roles
    4040: Features: happily (property: happily) (is_main: true) (clause_holders:
  3944: , thing semantic-interpretation
```

Figure 4-2: Genesis Parse Tree for an Individual Sentence

I wanted this evaluation process to stress the importance of Relations, which represent explicit descriptions and the actions characters perform, both of which are very telling of a character's emotional state. I also wanted to ensure that terms with a very high emotional score have the ability to overpower weaker terms and influence the final outcome, but because terms can evoke multiple emotions, I did not want to disregard terms completely. Therefore, I settled on the recursive algorithm shown in Figure 4-3. Recall that the score is represented as a six dimensional vector, with each dimension corresponding to an emotion in Ekman's six.

The core objective of this evaluation is to average the emotional scores of child terms and then average that result with the score of the parent term, moving from the leaves of the tree to the root, eventually determining an emotional score for the entire sentence. Note that this function does not consider terms that evoke no emotion whatsoever to avoid diluting the scores. This algorithm captures the modifier hierarchy of the sentence and also ensures that nodes that are closer to the root of



```
def getEmotion(node):

    if node.isLeaf():
        return node.score

    else:
        summed = [0,0,0,0,0,0]
        counter = 0

        for child in node.children:
            child.score = getEmotion(child)

            if child.score != [0,0,0,0,0,0]:
                summed += node.score
                counter +=1

        childAvg = summed / counter

    if node.score != [0,0,0,0,0,0]:
        return (childAvg + node.score)/2

    else:
        return childAvg
```

Figure 4-3: Pseudocode of Emotional Analysis Algorithm for a Sentence Subtree

the sub tree have more staying power, as they will go through fewer averaging rounds before a final score for the sentence is computed. The root of the sentence subtree is almost always a Relation, so this satisfies my desire to stress their importance. Furthermore, nodes that have very high emotional scores will have a large influence on the final outcome and will overpower weaker scores. However, the average operation does not completely disregard any term or any emotion, allowing for a consensus of terms expressing one emotion to overpower a singular high scoring instance of another emotion.

4.5 Story Alignment

I have discussed how understanding the emotional alignment of individual words helps Isabella understand the emotions evoked by a single sentence, but determining the emotions evoked by a series of sentences that tell a story is an entirely different task. But before I describe how Isabella handles empathy at this scope, let me give a concrete example of the type of problem I am trying to solve. Consider the following story:

James is a person. Linda is a person. James is Linda's boyfriend. Linda wins an award.

Now we can tell that Linda is probably experiencing joy after winning that award, but what is James likely feeling?

Well, we have no explicit information concerning his feelings, so even a human can only make a guess based on past experience and personal biases. However, in the absence of any information to the contrary, a human reader would likely say that James is also feeling joy because something good happened to someone he likes and cares about. Of course, a more cynical person might say that James could be jealous and angry. Humans can hallucinate rules and reasons that might explain connections and can use their memories to roughly determine how likely each possibility is. But how can we teach a program to handle these situations?

4.5.1 Design Choices

We could provide Isabella with a set of probabilistic rules. After all, Genesis is primarily a rule-based system and other emotion extraction studies have shown that hand crafted rules can be effective. However, there are two major issues with that approach. Firstly, it would be extremely hard to create rules that were not biased. Eventually, you'd get a system that simply thought exactly like its creator. Secondly, the amount of rules required to deal with every situation would be astronomical. We could employ some form of crowd sourcing and have many people create the rules, but this tactic is inefficient at best and at worse, infeasible and inaccurate.

Therefore, my solution is to provide Isabella with her own memories and give her the means to identify which memories are applicable to the situation at hand, allowing her to make inferences based on past, albeit simulated, experiences.

Fortunately, Genesis has a module that can align two stories and determine how similar they are. The details of this program were discussed in Chapter Two, but it essentially uses the semantic parse trees to bind entities and determine similar relationships.

I created a memory bank composed of stories that are annotated with an emotional tag. For example, the James and Linda story above would be tagged with *James = Joy*. Isabella can use this aligner to compare the current story to each one in her memories and when she finds a match, she can posit that the character in the story at hand that aligned with the tagged character from the memory is likely feeling the same emotion. If multiple stories match, she can use the alignment scores to weight the possible options and present an inference with some confidence value.

Along with an annotation, each story in Isabella's memory bank must also satisfy an invariant. Consider this story:

XX is a person. YY is a person. XX likes YY. YY has red hair. YY is happy. (XX=joy).

Now a human reader understands that XX is happy because someone he likes is also happy and that YY having red hair is not what is affecting XX's emotional state. It's an irrelevant sentence. But the story aligner will not make that distinction and a story that does not include a statement about a character's red hair will have a much lower alignment score.

Therefore, in order to generate accurate results, I must ensure that every story in the memory bank contains only the information necessary to arrive at the conclusion stated in the annotation. Of course, the converse does not need to hold. All the elements in the story at hand do not have to be present in the memory for that memory to be applicable.

4.5.2 Case Analysis

Let us walk through an example to illustrate all of this. The memory story is:

XX is a person. YY is a person. XX likes YY. YY is happy. (XX=joy).

The story at hand is:

Audrey is a person. James is a person. Audrey likes James. Audrey has known James for many years. James is happy. James likes Audrey.

Every element in the memory has a match, so the memory is applicable. Isabella can insert her hypothesis that *Audrey is likely feeling joy* before the last sentence in the story, where the memory ends.

Now what if Isabella had another memory that was very similar, but had a different emotional annotation?

XX is a person. YY is a person. XX likes YY. YY is happy. (XX=anger).

Perhaps in this memory, XX became jealous. Both memories will match, which makes the need for a confidence factor apparent. When Isabella makes an inference based on memories, the inference is weighted by $\frac{\textit{score of returned alignment}}{\textit{sum of all alignment scores}}$ Obviously, the more stories Isabella has in her memory bank, the more accurate this confidence factor becomes.

There is one more case to consider. Let's say there are two stories in Isabella's memory bank.

XX is a person. YY is a person. XX likes YY. YY is happy. (XX=joy).

XX is a person. YY is a person. XX likes YY. YY hurts XX. XX hates YY. YY is happy. (XX=anger).

And we have the following story at hand:

Audrey is a person. James is a person. Audrey likes James. James hurts Audrey. Audrey hates James. James is happy.

Now, both memories will match, but the set of aligned elements to the second memory is a superset of that of the first memory. This means that the second memory is strictly a better match and will overrule the first memory and Isabella will only hypothesize that *Audrey is feeling anger.*

4.5.3 Limitations and Future Work

The major limitation of this approach is that the effectiveness of this memory bank is based on how many stories Isabella has stored. For example, if she didn't have the second memory, Isabella would assume that *"Audrey is feeling joy"* because she would not have enough information to make sense of the intermediate aspects of the story, which, unlike the red hair in the first example we discussed, did affect the end result.

I also need a method to add stories to the memory bank without violating the invariants we discussed. This feedback method has not been implemented yet, but will be in the near future. The plan is to add a GUI that would appear after Genesis processes a story. Should the user feel that the story demonstrates an emotional memory, they could remove elements from a story until only the minimal set of events remain and then tag that story with an emotional annotation. I want this process to be efficient so that all Genesis users will be able to contribute to Isabella's memories quickly and easily. This system would also allow users to correct Isabella when she makes mistakes, which will update the score values for component terms or memories.

4.6 Multi-Scope Alignment

The Isabella program operates at three scopes in order to make inferences about the emotional states of characters in stories. I have described how I computed scores for the emotions a single word can evoke using a vector space model informed by four data sources. I then illustrated how Isabella uses that information and the parse tree generated by Genesis to combine modifiers according to the structure of the sentence in order to infer the emotions of the sentence's subject. Finally, we discussed Isabella's ability to use the Genesis story alignment module to compare multi-sentence stories to a memory bank in order to determine when past experiences can provide insight about the situation at hand.

Chapter 5

Discussion of Results

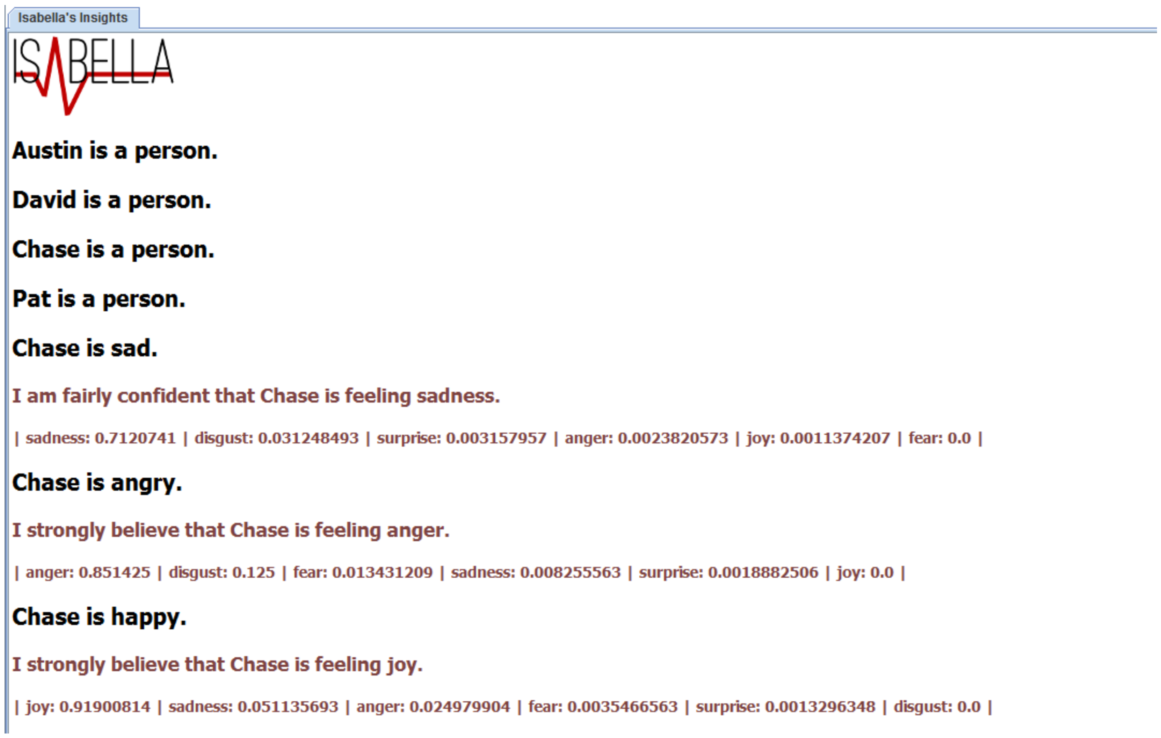
In this chapter, I will demonstrate the abilities of Isabella. Because it is difficult to ascertain ground truth when it comes to the very subjective task of empathizing, and because, even if that wasn't the case, there is no STARTparsable corpus of stories annotated with emotions to test with, I will use a case analysis in this discussion. I will also present some of the limitations of the program and what additions could be made to improve those aspects.

5.1 Isabella's Insights

The output of Isabella is shown below in Figure 5-1

The sentences in black text are the original input story's sentences.

The sentences in red are Isabella's insights on the story and are interleaved with the story text. Isabella will state her assessment and how confident she is. The complete numerical score of the alignment will also be printed. Multi-sentence insights will be printed after the last matching sentence in the original story and Isabella will indicate the relevant events that led to the assessment.



Austin is a person.

David is a person.

Chase is a person.

Pat is a person.

Chase is sad.

I am fairly confident that Chase is feeling sadness.

| sadness: 0.7120741 | disgust: 0.031248493 | surprise: 0.003157957 | anger: 0.0023820573 | joy: 0.0011374207 | fear: 0.0 |

Chase is angry.

I strongly believe that Chase is feeling anger.

| anger: 0.851425 | disgust: 0.125 | fear: 0.013431209 | sadness: 0.008255563 | surprise: 0.0018882506 | joy: 0.0 |

Chase is happy.

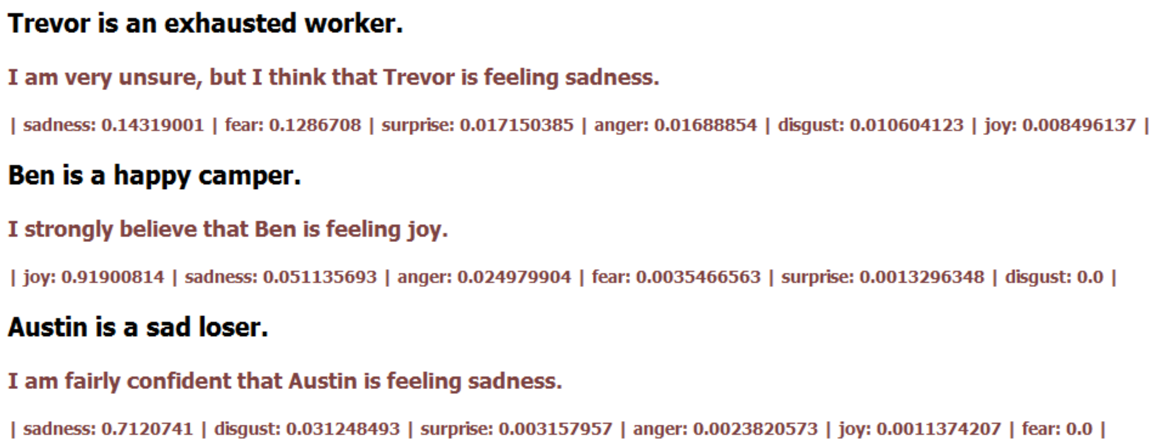
I strongly believe that Chase is feeling joy.

| joy: 0.91900814 | sadness: 0.051135693 | anger: 0.024979904 | fear: 0.0035466563 | surprise: 0.0013296348 | disgust: 0.0 |

Figure 5-1: Isabella's Insights

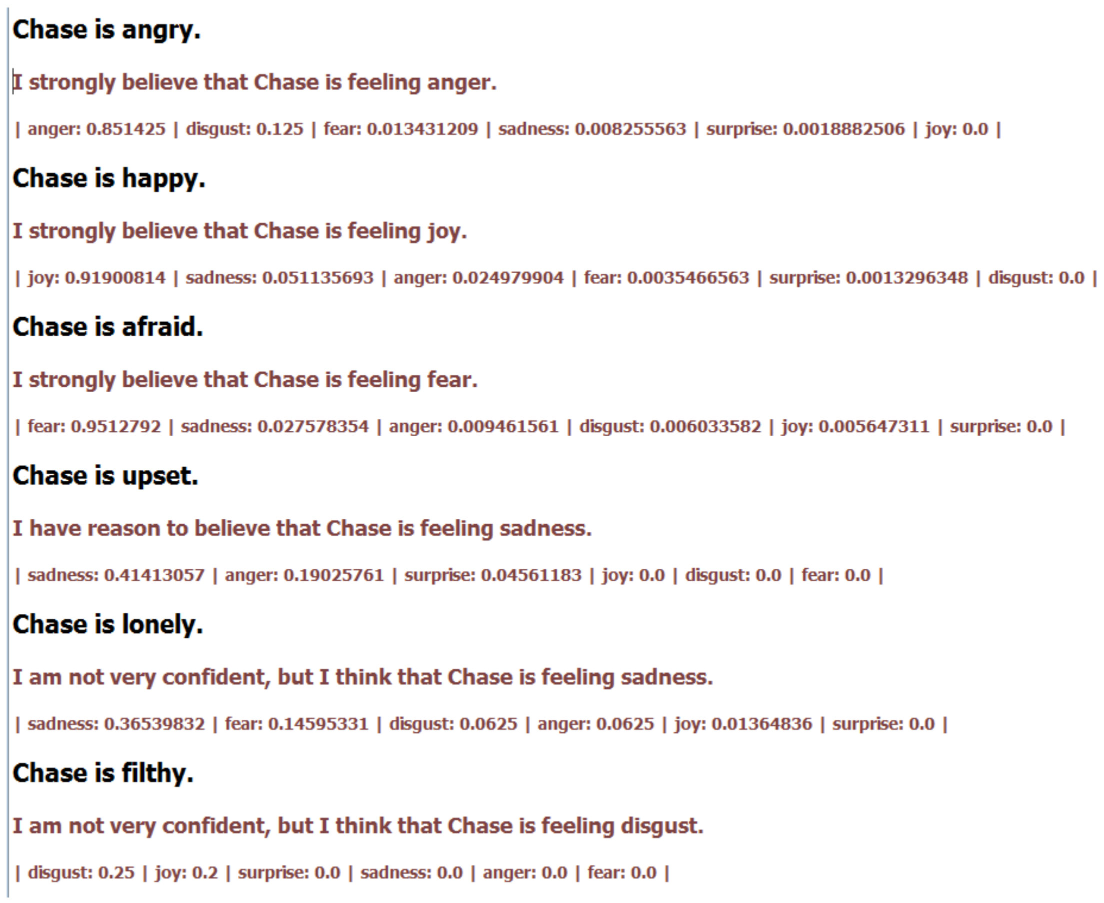
5.2 Demonstrations

The following figures demonstrate the types of situations that Isabella can successfully handle.



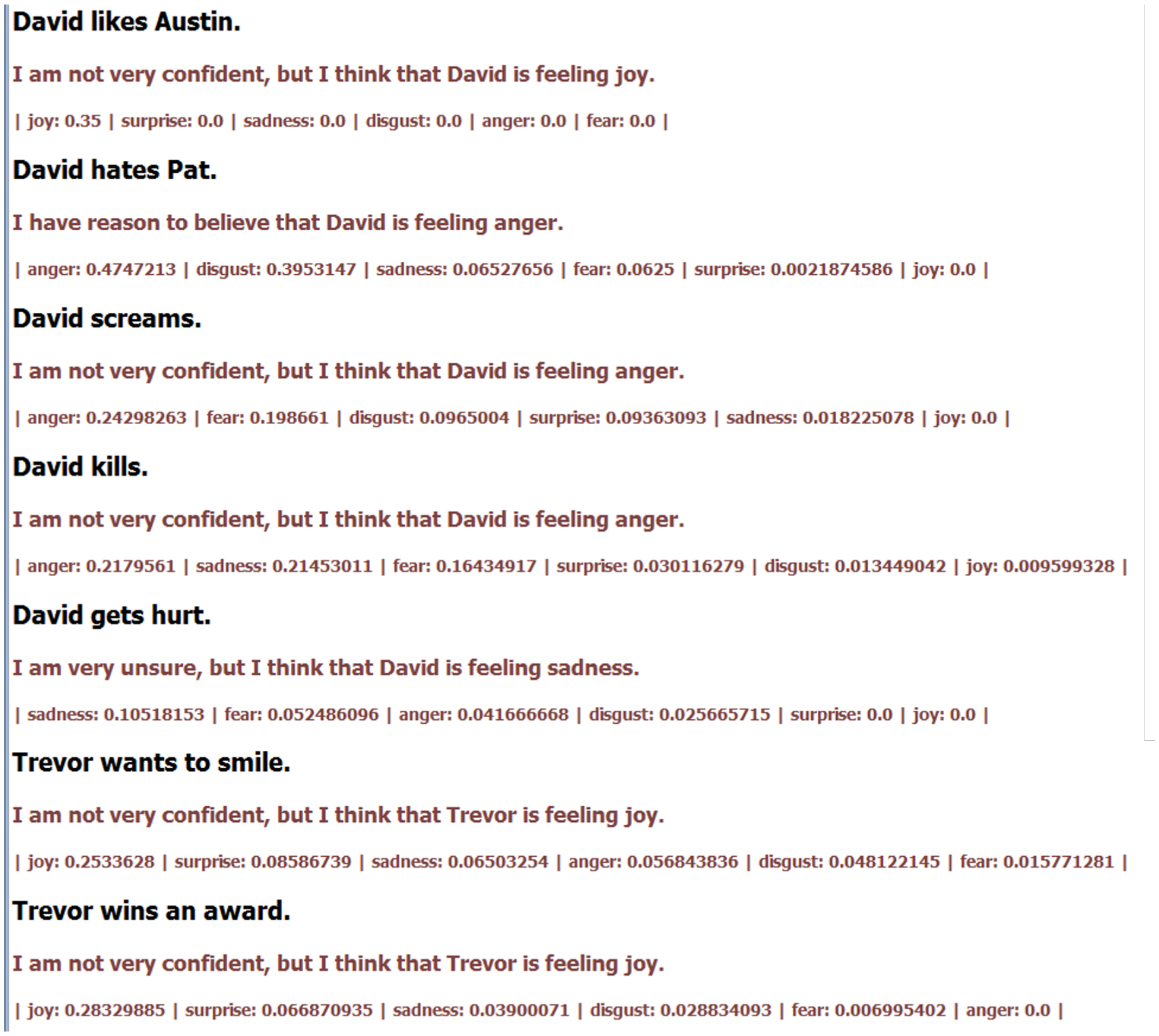
```
Trevor is an exhausted worker.
I am very unsure, but I think that Trevor is feeling sadness.
| sadness: 0.14319001 | fear: 0.1286708 | surprise: 0.017150385 | anger: 0.01688854 | disgust: 0.010604123 | joy: 0.008496137 |
Ben is a happy camper.
I strongly believe that Ben is feeling joy.
| joy: 0.91900814 | sadness: 0.051135693 | anger: 0.024979904 | fear: 0.0035466563 | surprise: 0.0013296348 | disgust: 0.0 |
Austin is a sad loser.
I am fairly confident that Austin is feeling sadness.
| sadness: 0.7120741 | disgust: 0.031248493 | surprise: 0.003157957 | anger: 0.0023820573 | joy: 0.0011374207 | fear: 0.0 |
```

Figure 5-2: Processing Noun Classifications



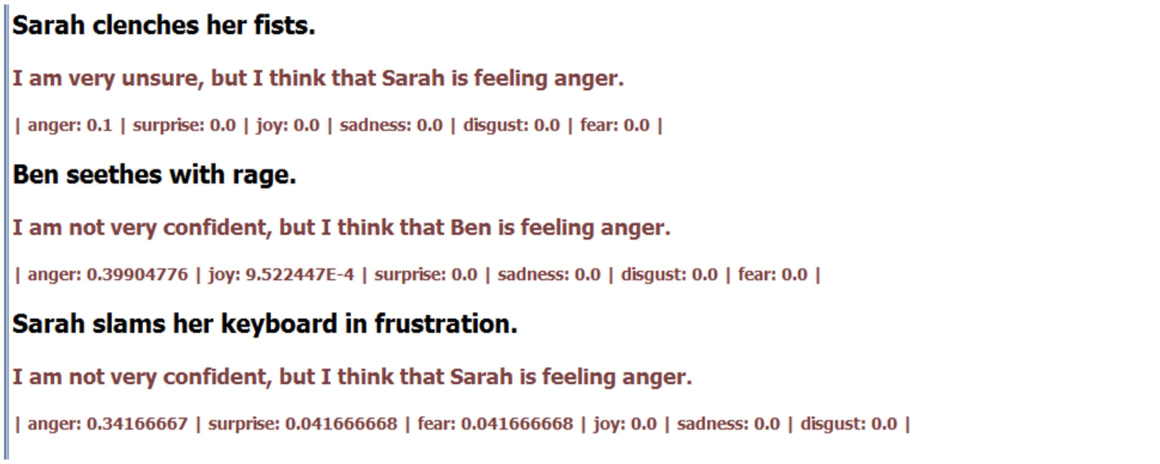
```
Chase is angry.
I strongly believe that Chase is feeling anger.
| anger: 0.851425 | disgust: 0.125 | fear: 0.013431209 | sadness: 0.008255563 | surprise: 0.0018882506 | joy: 0.0 |
Chase is happy.
I strongly believe that Chase is feeling joy.
| joy: 0.91900814 | sadness: 0.051135693 | anger: 0.024979904 | fear: 0.0035466563 | surprise: 0.0013296348 | disgust: 0.0 |
Chase is afraid.
I strongly believe that Chase is feeling fear.
| fear: 0.9512792 | sadness: 0.027578354 | anger: 0.009461561 | disgust: 0.006033582 | joy: 0.005647311 | surprise: 0.0 |
Chase is upset.
I have reason to believe that Chase is feeling sadness.
| sadness: 0.41413057 | anger: 0.19025761 | surprise: 0.04561183 | joy: 0.0 | disgust: 0.0 | fear: 0.0 |
Chase is lonely.
I am not very confident, but I think that Chase is feeling sadness.
| sadness: 0.36539832 | fear: 0.14595331 | disgust: 0.0625 | anger: 0.0625 | joy: 0.01364836 | surprise: 0.0 |
Chase is filthy.
I am not very confident, but I think that Chase is feeling disgust.
| disgust: 0.25 | joy: 0.2 | surprise: 0.0 | sadness: 0.0 | anger: 0.0 | fear: 0.0 |
```

Figure 5-3: Handling Explicit Descriptions with Adjectives



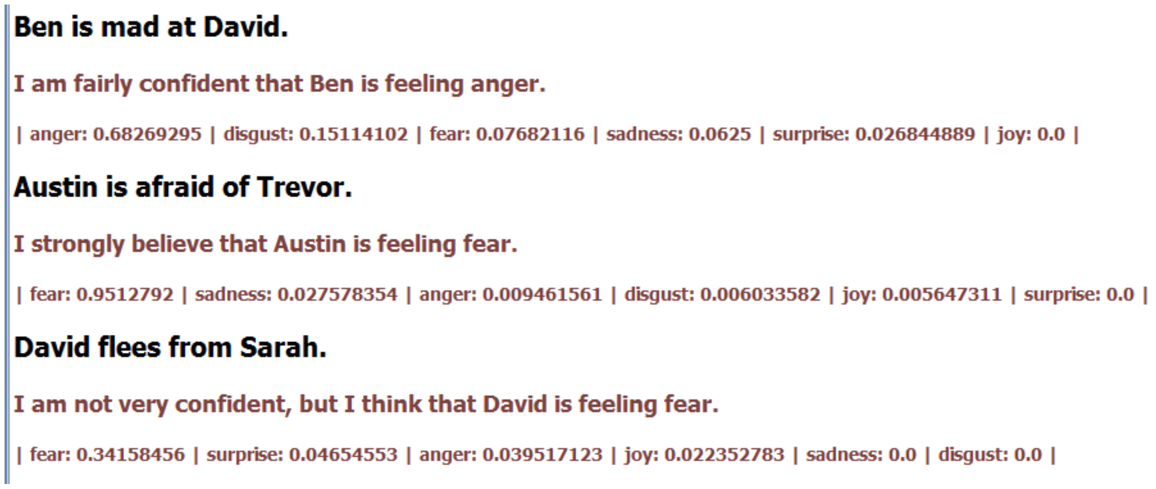
```
David likes Austin.
I am not very confident, but I think that David is feeling joy.
| joy: 0.35 | surprise: 0.0 | sadness: 0.0 | disgust: 0.0 | anger: 0.0 | fear: 0.0 |
David hates Pat.
I have reason to believe that David is feeling anger.
| anger: 0.4747213 | disgust: 0.3953147 | sadness: 0.06527656 | fear: 0.0625 | surprise: 0.0021874586 | joy: 0.0 |
David screams.
I am not very confident, but I think that David is feeling anger.
| anger: 0.24298263 | fear: 0.198661 | disgust: 0.0965004 | surprise: 0.09363093 | sadness: 0.018225078 | joy: 0.0 |
David kills.
I am not very confident, but I think that David is feeling anger.
| anger: 0.2179561 | sadness: 0.21453011 | fear: 0.16434917 | surprise: 0.030116279 | disgust: 0.013449042 | joy: 0.009599328 |
David gets hurt.
I am very unsure, but I think that David is feeling sadness.
| sadness: 0.10518153 | fear: 0.052486096 | anger: 0.041666668 | disgust: 0.025665715 | surprise: 0.0 | joy: 0.0 |
Trevor wants to smile.
I am not very confident, but I think that Trevor is feeling joy.
| joy: 0.2533628 | surprise: 0.08586739 | sadness: 0.06503254 | anger: 0.056843836 | disgust: 0.048122145 | fear: 0.015771281 |
Trevor wins an award.
I am not very confident, but I think that Trevor is feeling joy.
| joy: 0.28329885 | surprise: 0.066870935 | sadness: 0.03900071 | disgust: 0.028834093 | fear: 0.006995402 | anger: 0.0 |
```

Figure 5-4: Identifying Characters Performing Actions



```
Sarah clenches her fists.
I am very unsure, but I think that Sarah is feeling anger.
| anger: 0.1 | surprise: 0.0 | joy: 0.0 | sadness: 0.0 | disgust: 0.0 | fear: 0.0 |
Ben seethes with rage.
I am not very confident, but I think that Ben is feeling anger.
| anger: 0.39904776 | joy: 9.522447E-4 | surprise: 0.0 | sadness: 0.0 | disgust: 0.0 | fear: 0.0 |
Sarah slams her keyboard in frustration.
I am not very confident, but I think that Sarah is feeling anger.
| anger: 0.34166667 | surprise: 0.041666668 | fear: 0.041666668 | joy: 0.0 | sadness: 0.0 | disgust: 0.0 |
```

Figure 5-5: Handling more Complex Sentence Structures.


Ben is mad at David.

I am fairly confident that Ben is feeling anger.

```
| anger: 0.68269295 | disgust: 0.15114102 | fear: 0.07682116 | sadness: 0.0625 | surprise: 0.026844889 | joy: 0.0 |
```

Austin is afraid of Trevor.

I strongly believe that Austin is feeling fear.

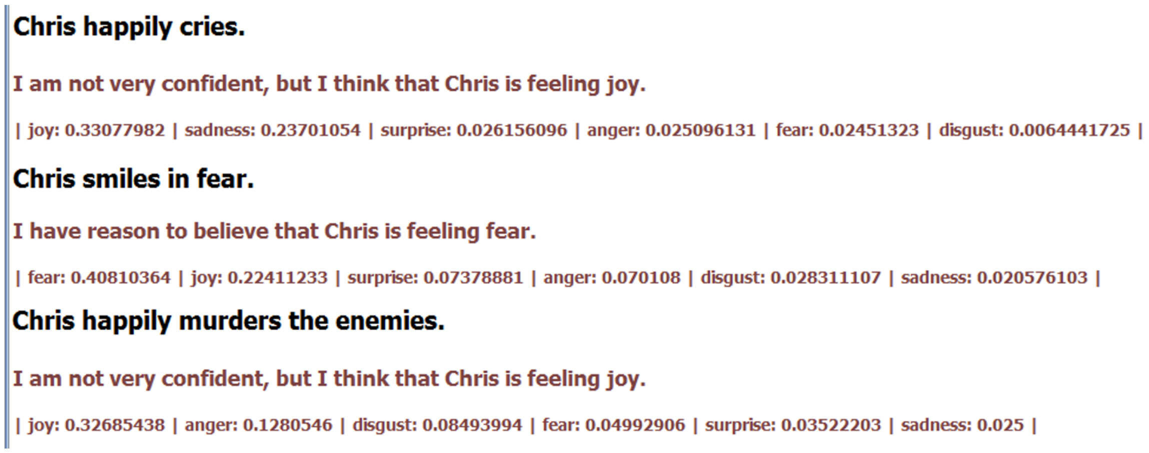
```
| fear: 0.9512792 | sadness: 0.027578354 | anger: 0.009461561 | disgust: 0.006033582 | joy: 0.005647311 | surprise: 0.0 |
```

David flees from Sarah.

I am not very confident, but I think that David is feeling fear.

```
| fear: 0.34158456 | surprise: 0.04654553 | anger: 0.039517123 | joy: 0.022352783 | sadness: 0.0 | disgust: 0.0 |
```

Figure 5-6: Targeted Empathy with Multiple Characters Present.


Chris happily cries.

I am not very confident, but I think that Chris is feeling joy.

```
| joy: 0.33077982 | sadness: 0.23701054 | surprise: 0.026156096 | anger: 0.025096131 | fear: 0.02451323 | disgust: 0.0064441725 |
```

Chris smiles in fear.

I have reason to believe that Chris is feeling fear.

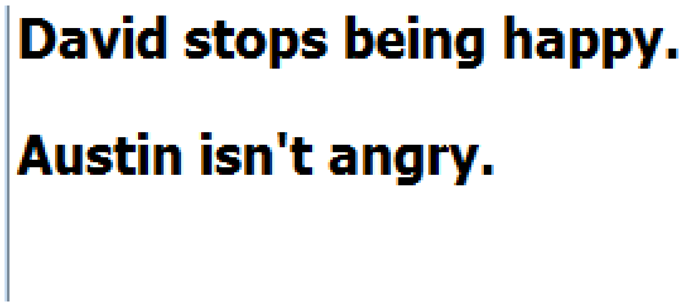
```
| fear: 0.40810364 | joy: 0.22411233 | surprise: 0.07378881 | anger: 0.070108 | disgust: 0.028311107 | sadness: 0.020576103 |
```

Chris happily murders the enemies.

I am not very confident, but I think that Chris is feeling joy.

```
| joy: 0.32685438 | anger: 0.1280546 | disgust: 0.08493994 | fear: 0.04992906 | surprise: 0.03522203 | sadness: 0.025 |
```

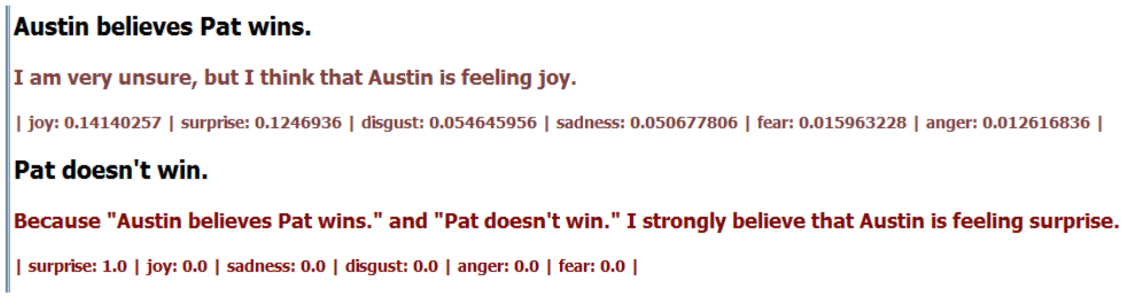
Figure 5-7: Handling Sentences with Mixed Emotions.


David stops being happy.

Austin isn't angry.

Figure 5-8: Correctly Handling the Lack of Any Emotion.

5.3 Multi-Sentence Alignment



Austin believes Pat wins.

I am very unsure, but I think that Austin is feeling joy.

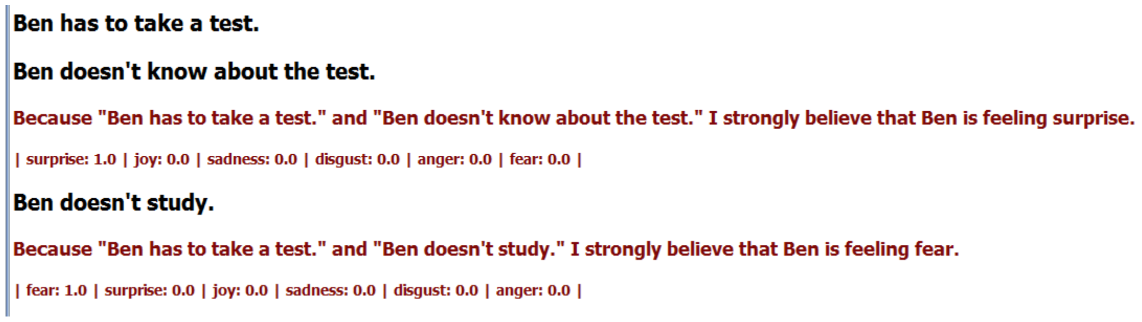
```
| joy: 0.14140257 | surprise: 0.1246936 | disgust: 0.054645956 | sadness: 0.050677806 | fear: 0.015963228 | anger: 0.012616836 |
```

Pat doesn't win.

Because "Austin believes Pat wins." and "Pat doesn't win." I strongly believe that Austin is feeling surprise.

```
| surprise: 1.0 | joy: 0.0 | sadness: 0.0 | disgust: 0.0 | anger: 0.0 | fear: 0.0 |
```

Figure 5-9: Simple Example of Multi-Sentence Alignment



Ben has to take a test.

Ben doesn't know about the test.

Because "Ben has to take a test." and "Ben doesn't know about the test." I strongly believe that Ben is feeling surprise.

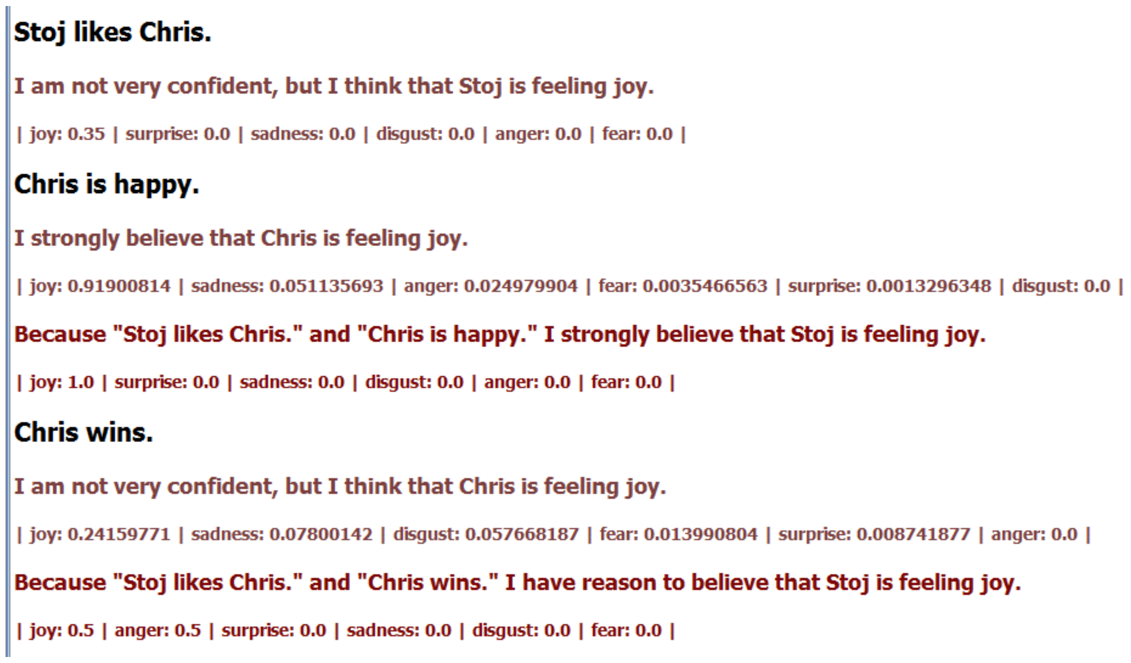
```
| surprise: 1.0 | joy: 0.0 | sadness: 0.0 | disgust: 0.0 | anger: 0.0 | fear: 0.0 |
```

Ben doesn't study.

Because "Ben has to take a test." and "Ben doesn't study." I strongly believe that Ben is feeling fear.

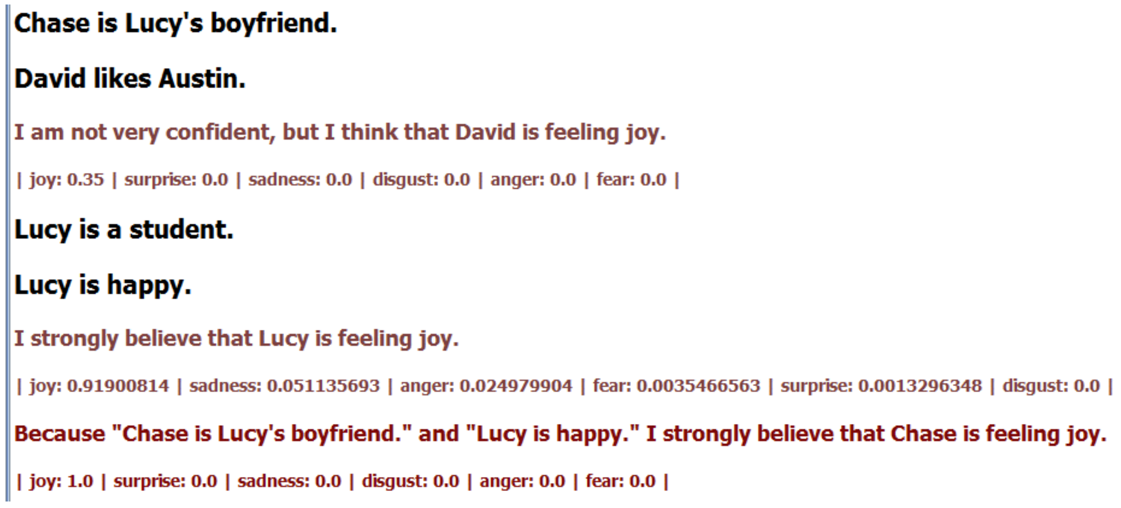
```
| fear: 1.0 | surprise: 0.0 | joy: 0.0 | sadness: 0.0 | disgust: 0.0 | anger: 0.0 |
```

Figure 5-10: Interleaved Memory Alignment



```
Stoj likes Chris.
I am not very confident, but I think that Stoj is feeling joy.
| joy: 0.35 | surprise: 0.0 | sadness: 0.0 | disgust: 0.0 | anger: 0.0 | fear: 0.0 |
Chris is happy.
I strongly believe that Chris is feeling joy.
| joy: 0.91900814 | sadness: 0.051135693 | anger: 0.024979904 | fear: 0.0035466563 | surprise: 0.0013296348 | disgust: 0.0 |
Because "Stoj likes Chris." and "Chris is happy." I strongly believe that Stoj is feeling joy.
| joy: 1.0 | surprise: 0.0 | sadness: 0.0 | disgust: 0.0 | anger: 0.0 | fear: 0.0 |
Chris wins.
I am not very confident, but I think that Chris is feeling joy.
| joy: 0.24159771 | sadness: 0.07800142 | disgust: 0.057668187 | fear: 0.013990804 | surprise: 0.008741877 | anger: 0.0 |
Because "Stoj likes Chris." and "Chris wins." I have reason to believe that Stoj is feeling joy.
| joy: 0.5 | anger: 0.5 | surprise: 0.0 | sadness: 0.0 | disgust: 0.0 | fear: 0.0 |
```

Figure 5-11: Interleaved Memories with Multiple Characters



```
Chase is Lucy's boyfriend.
David likes Austin.
I am not very confident, but I think that David is feeling joy.
| joy: 0.35 | surprise: 0.0 | sadness: 0.0 | disgust: 0.0 | anger: 0.0 | fear: 0.0 |
Lucy is a student.
Lucy is happy.
I strongly believe that Lucy is feeling joy.
| joy: 0.91900814 | sadness: 0.051135693 | anger: 0.024979904 | fear: 0.0035466563 | surprise: 0.0013296348 | disgust: 0.0 |
Because "Chase is Lucy's boyfriend." and "Lucy is happy." I strongly believe that Chase is feeling joy.
| joy: 1.0 | surprise: 0.0 | sadness: 0.0 | disgust: 0.0 | anger: 0.0 | fear: 0.0 |
```

Figure 5-12: Memory Alignment Despite Irrelevant Information

5.4 Analysis of Successes and Limitations

As illustrated by the preceding figures, Isabella can correctly handle a wide range of situations. However, there are also several limitations and potential future improvements to discuss.

5.4.1 START/Genesis and Isabella

The process for handling explicit descriptions and characters performing actions is relatively simple and very accurate. However, if Isabella encounters a term that was not in any of the datasets, she simply has no idea what emotions it could evoke, and will not be able to make an assessment. Similarly, if a term has a very low confidence score or an incorrect association due to inaccuracies in the dataset, then Isabella will likely make a wrong inference. The solution to these types of limitations is to increase the number of datasets Isabella is learning from. This reduces the chance that she encounters a term she has never seen before and the use of multiple sources reduces the chance that a word is misused or misclassified. The feedback system described at the end of Chapter Four would also help Isabella learn from her mistakes, augmenting the set of terms she is familiar with.

Additinoally, as the sentence structure gets more complicated, the probability that Isabella fails increases, mostly due to the START/Genesis parse. In some cases, START or Genesis simply produce an incorrect parse. For example, in the sentence *"He was scared"*, START identifies *scared* as a past tense verb and not an adjective. The resulting tree is therefore invalid for the explicit description pattern that Isabella can deal with, and she is not able to make an assessment.

In other cases, the fault lies with Isabella. Some of the parse trees produced by Genesis are very complicated and Isabella cannot determine if a person of interest is indeed the subject of the sentence. In order to resolve these shortcomings, I need to give Isabella a more complete set of instructions on how to analyze the START/Genesis parse tree.

It should be noted that in a lot of situations, the Genesis parse makes things easier for Isabella. For example, in the sentence *"Tom and Bob run to the river"*, Genesis splits the sentence up into two separate sentences - one concerning Tom and the other, Bob. This simplifies the situation by ensuring that Isabella will make two distinct assessments.

5.4.2 Nuanced Situations

Isabella does a great job of handling sentences with multiple people, only empathizing with the subject of the sentence. However, there are times when the sentence does in fact concern the emotional state of the object. For instance, consider the sentence *Tom broke Audrey's heart.* Tom performed the action, but Audrey is the one who is likely now saddened. Isabella, in her current state, is not able to correctly make that assessment. Even though in most cases, a sentence allows you to empathize with the subject, Isabella will need to be taught to recognize cases such as these.

Sentences with multiple, often conflicting, emotions can be difficult for humans to parse. Isabella handles these relatively well, but the end result is extremely dependent on the weighting of individual words. In some cases, the numeric score of a word is not quite what you'd expect to be, which can result in some unanticipated insights from Isabella. As discussed before, additional data upon which to form the word alignments would improve the situation. Furthermore, as a point of future study, I would like to explore the possibility of other scoring amalgamation techniques to see if comparison based approaches or exponentiation of the scores would provide more accurate results than the mean based algorithm I am currently using.

It should also be noted that Isabella is aware that when a character is experiencing the lack of an emotion, it does not mean they are necessarily experiencing another emotion. Furthermore, when a character stops feeling an emotion, Isabella recognizes that they do not have to start feeling something else. In these cases, Isabella's lack of an assessment as seen in Figure 5-8 is precisely the behavior I desired.

5.4.3 Memory Banks and Story Alignment

Isabella can handle several memories being aligned to the story at hand simultaneously, interleaved memories, and irrelevant information with ease. However, the aligner itself is fairly fragile and specifics in the story need to be almost identical to those of a memory in order to match. Obviously, the aligner can match people with different names and general types of relationships, but additional flexibility would greatly improve the system. I do plan to extend the aligner's functionality to consider synonyms and hypernyms of terms to be valid matches. The difficulty comes in determining how much flexibility I can introduce without generating invalid pairings.

However, the most important improvement I can make in this area is the creating of the feedback system as described in Chapter Four. Currently, Isabella's memory bank is static and there is no way to add additional memories without writing and inserting the stories directly into the code, a process that would not encourage other users to contribute. When the feedback system is in place, the hope is that eventually, Isabella will have so many past experiences to draw on that her reliance on individual word scores is greatly lessened, alleviating some of the concerns expressed earlier.

5.4.4 Summary

All things considered, I feel that Isabella is able to handle most of the situations I originally wanted to deal with and she certainly will be a nice addition to the many analysis modules of Genesis. Hopefully, I will be able to make some of the improvements discussed and integrate her module completely, allowing Genesis to seamlessly empathize with all the characters it encounters.

Chapter 6

Contributions

In this thesis, I described an artificial empathy program that I developed to be part of the Genesis story understanding system. I provided an overview of the Genesis system along with a brief discussion about its design principles, which have also influenced my program. I also examined several past studies that sought to extract emotion from text and described how my program leveraged their insights. Finally, I presented the results of my program, analyzed its abilities, acknowledged its limitations, and addressed future plans for improvement.

My program has:

- Addressed some of the limitations of previous studies by combining large amounts of data and shallow processing with small data, deep understanding paradigms.

- Illustrated the benefits of a mixed scope approach by combining the emotional scores of individual words with the semantic structure of sentences as well as employing story alignment in order to produce results.

-Demonstrated the ability to specifically infer the emotions of characters as opposed to identifying the emotions evoked by the text as a whole.

- Added another analysis module to the Genesis toolkit, improving its story understanding capabilities

Bibliography

[1] M.Ishizuka A.Neviarouskaya, H.Prendinger. Compositionality principle in recognition of fine-grained emotions from text. *Proceedings of the Third International ICWSM Conference*, 2009.

[2] R.Sproat C.O.Alm, D.Roth. Emotions from text: machine learning for text-based emotion prediction. *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, 2005.

[3] R.Mihalcea C.Strapparava. Semeval-2007 task 14: Affective text. *Proceedings of the 4th International Workshop on the Semantic Evaluations*, 2007.

[4] V.Valitutti C.Strapparava. Wordnet-affect: an affective extension of wordnet. *Proceedings of the 4th international conference on language resources and evaluation*, 2004.

[5] Y.Lin C.Wu, Z.Chuang. Emotion recognition from text using semantic labels and separable mixture models. *ACM Transactions on Asian Language Information Processing TALIP*, 2006.

[6] D.Inkpen F.Keshtkar. A corpus-based method for extracting paraphrases of emotion terms. *Computational Intelligence*, 2012.

[7] G.Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[8] G.Mishne. Experiments with modd classification in blog posts. *ACM SIGIR*, 2005.

[9] J.Kaufman. https://github.com/first20hours/google-10000-english.

[10] J.Read. Recognising affect in text using pointwise-mutual information. 2004.

[11] M.Fay. Enabling imagination through story alignment. 2011.

[12] P.Ekman. An argument for basic emotions. *Cognition and Emotion*, 1992.

[13] P.H.Winston. The strong story hypothesis and the directed perception hypothesis. *Proceedings of the AAAI Fall Symposium on Advances in Cognitive Systems*, 2011.

[14] P.H.Winston. The right way. *Advances in Cognitive Systems*, 2012.

[15] P.H.Winston. Genesiss implementation substrate. `http://groups.csail.mit.edu/genesis/Documentation/frames.pdf`, 2015.

[16] P.H.Winston. How to speak genesese. `http://groups.csail.mit.edu/genesis/Documentation/Genesese.pdf`, 2016.

[17] R.Plutchik. A general psychoevolutionary theory of emotion. *Emotion Theory, Research and Experience*, 1, 1980.

[18] R.Plutchik. The nature of emotions. *American Scientist*, 2001.

[19] S.Szpakowicz S.Aman. Identifying expressions of emotion in text. *Text, Speech and Dialogue*, 2007.

[20] S.Szpakowicz S.Aman. Using roget's thesaurus for fine-grained emotion recognition. *IJCNLP*, 2008.

[21] E.S. Sayan. Audience aware computational discourse generation for instruction and persuasion. 2014.

[22] P. Turney S.Mohammad. Emotions evoked by common words and phrases: using mechanical turk to create an emotion lexicon. *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, 2010.

[23] A. Alpkocak T.Danisman. Feeler: Emotion classification of text using vector space model. *ISB 2008 Convention Communication, Interaction and Social Intelligence*, 2008.