Designing an Exploratory Text Analysis Tool for Humanities and Social Sciences Research

Aditi Shrikumar

Designing an Exploratory Text Analysis Tool for Humanities and Social Sciences Research

By

Aditi Shrikumar

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:
Professor Marti A. Hearst, Co-chair
Assistant Professor Bjoern Hartmann, Co-chair
Associate Professor Daniel Klein
Associate Professor Bryan Wagner

Fall 2013

Designing an Exploratory Text Analysis Tool for Humanities and Social Sciences Research

Abstract

Designing an Exploratory Text Analysis Tool for Humanities and Social Sciences Research

by

Aditi Shrikumar

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Marti A. Hearst, Co-chair
Assistant Professor Bjoern Hartmann, Co-chair

This dissertation presents a new tool for exploratory text analysis that attempts to improve the experience of navigating and exploring text and its metadata. The design of the tool was motivated by the unmet need for text analysis tools in the humanities and social sciences. In these fields, it is common for scholars to have hundreds or thousands of text-based source documents of interest from which they extract evidence for complex arguments about society and culture. These collections are difficult to make sense of and navigate. Unlike numerical data, text cannot be condensed, overviewed, and summarized in an automated fashion without losing significant information. And the metadata that accompanies the documents – often from library records – does not capture the varied content of the text within.

Furthermore, adoption of computational tools remains low among these scholars despite such tools having existed for decades. A recent study found that the main culprits were poor user interfaces and lack of communication between tool builders and tool users. We therefore took an iterative, user-centered approach to the development of the tool. From reports of classroom usage, and interviews with scholars, we developed a descriptive model of the text analysis process, and extracted design guidelines for text analysis systems. These guidelines recommend showing overviews of both the content and metadata of a collection, allowing users to separate and compare subsets of data according to combinations of searches and metadata filters, allowing users to collect phrases, sentences, and documents into custom groups for analysis, making the usage context of words easy to see without interrupting the current activity, and making it easy to switch between different visualizations of the same data.

WordSeer, the system we implemented, supports highly flexible slicing and dicing, as well as easier transitions than in other tool between visual analyses, drill-downs, lateral explorations and overviews of slices in a text collection. The tool uses techniques from computational linguistics, information retrieval and data visualization.

The contributions of this dissertation are the following. First, the design and source code of WordSeer Version 3, an exploratory text analysis system. Unlike other current systems for this audience, WordSeer 3 supports collecting evidence, isolating and analyzing sub-sets of a collection, making comparisons based on collected items, and exploring a new idea without interrupting the current task. Second, we give a descriptive model of how humanities and social science scholars undertake exploratory text analysis during the course of their work. We also identify pain points in their current workflows and give suggestions on how systems can address these problems. Third, we describe a set of design principles for text analysis systems aimed at addressing these pain points. For validation, we contribute a set of three real-world examples of scholars using WordSeer 3, which was designed according to those principles. As a measure of success, we show how the scholars were able to conduct analyses yielding otherwise inaccessible results useful to their research.

For Dad,
who put the idea in my head,
and Mom,
who thought it was cool.

Contents

List of Figures

List of Tables

Acknowledgments

To my advisor, Marti Hearst, go my first thanks. She is quite possibly the best advisor I have ever heard of, and graduate school has been exciting and enriching because of her. Thanks also to Bjoern Hartmann, Dan Klein, and Bryan Wagner, who have always had incisive, helpful suggestions. WordSeer is a much better program because of your guidance. I'm also very grateful to my other collaborators, who have often stepped in with much-needed perspective, and always been generous with their time. Michael Ullyot – thank you for taking a chance on WordSeer for English 203, and Natalia Cecire, thanks for always being willing to chat.

I would like to thank my interviewees who carefully explained how they did research, and the students of English 203 who blogged so descriptively about using WordSeer to analyze Hamlet. You helped me come to grips with the text analysis process.

I owe a huge debt to the active, friendly and open digital humanities community on twitter and at THATCamp. When I first became interested in this field, I knew hardly anything about the humanities and social sciences. I have learned so much from following my twitter `#dh` stream, attending the THATCamps, reading the popular digital humanities articles that are constantly re-tweeted, and, finally, being encouraged to start my own blog, and having my posts responded to and discussed. Included in this list are my friends Scott McGinnis and Chris Church, without whom the digital humanities community at Berkeley would never have come together.

To Brett Bobley, Jason Rhody, and the rest of the staff at the NEH Office of Digital Humanities: thank you for your vision. You have always been almost unnervingly excited about WordSeer – even when it was just a search box – and more than that, you have supported it with two grants. The start-up grant (NEH HD-51244-11) gave us the confidence (and resources) to conduct the first two case studies, and the implementation grant (NEH HK-50011) has allowed the project to blossom into a full-fledged text analysis system.

And finally, a special thank you to my husband Omkar, with whom every day is new, even after ten years; and to my family, who are always proud of me, no matter what I do.

Chapter 1

Introduction

In the humanities and social sciences many scholars work with text-based primary sources. These are cultural and historical documents like museum records, papers from historical societies, papers stored in archives, historical government records, letters, newspapers, books, and magazines. Scholars look through these items for evidence that supports their characterizations, descriptions, or arguments about historical events, people, culture, or society (Palmer and Cragin 2008; Palmer *et al.* 2009; Blanke and Hedges 2013).

The increasing prevalence of online archives of documents has made access to text-based primary source materials much easier than in the past (Blanke and Hedges 2013). Now, instead of traveling to an archive (possibly located in a different city or country), and manually sifting through the papers in a box, a scholar can identify and download relevant materials over the internet. But finding and accessing source texts is just one of the many problems these scholars face.

There are certain activities that are common across all humanities and social science disciplines. These are searching for information, gathering and organizing it, skimming and re-reading it in order to gain new knowledge and assess its importance, note-taking, translating, assembling materials for writing, and disseminating a final product (Palmer and Cragin 2008). In other words, these scholars engage in *sensemaking.* The term, introduced by Weick (1979) and popularized in the computer science literature by Russell *et al.* (1993), stands for the process that people engage in while attempting to understand a topic based on a large quantity of information.

1.1 Motivation

Sensemaking is generally acknowledged to be a difficult and time-consuming activity. Fortunately, there is now a rich body of research on understanding the process itself, and on developing tools that offer assistance.

Observational studies reveal the activities that constitute the sensemaking process and the accompanying frustrations. Sensemaking is a cycle between two kinds of activities: information seeking and synthesis (Russell *et al.* 1993; Pirolli and Card 2005; O'Day and

Jeffries 1993). The first step, information seeking, is itself a cylcle consisting of searching for, identifying, and extracting information from documents that are returned in response to multiple queries. The second step, synthesis, consists of using the documents returned by a query to read, learn, and change one's internal conceptual understanding of the task. Synthesis results in new understanding, which creates new queries, and new documents to look at, so the cycle continues until the task is complete. The end goal of sensemaking is a coherent understanding – enough to support a decision, or to contribute to a written report or presentation for consumption by others.

Past work on sensemaking-support systems has mainly focused on market research, statistics, medical research, legal discovery, and government intelligence. These professionals typically work with vast collections of short documents (article-length as opposed to book-length). For instance, O'Day and Jeffries (1993) observed the information practices of financial and business-related researchers, and found that in addition to reading and annotating, the analysts spent roughly 80% of their time outside of search doing the following:

- Looking for trends or correlations
- Making comparisons of different pieces of the data set
- Experimenting with different aggregates and/or scaling
- Identifying a critical subset of relevant or unique items
- Making assessments
- Interpreting data to find meaning in terms of domain or problem concepts

And the remaining time cross-referencing, summarizing, finding visualizations, and other miscellaneous activities.

However, there are other important types of sensemaking that have not been studied. In 'Sensemaking for the Rest of Us', Russell *et al.* (2008) called attention to 'everyday sensemaking', and urged researchers to look at other types of users who have been 'left out':

> Most people don't actually need to visualize high dimensional data sets that incorporate time-varying series data. Instead, it's pretty clear that a large fraction of everyday sensemaking tasks are fairly simple: one collects a bunch of data on a topic of interest, and then needs to lightly re-organize it in order to determine the best option among many to take. This description can apply to everything from selecting a new (or used) car to buy, to deciding which grad school to attend, to understanding where to go on vacation.

Humanities and social sciences scholars are one such 'left out' group. They are usually non-technical and prefer not to use complex user interfaces (Gibbs and Owens 2012). But there are more fundamental differences that make their workflows distinct from the other types sensemaking tasks previously studied. First, the objects of study are not necessarily explicit data like numbers, facts, entities, or sequences of events, but implicit data like impressions, perceptions, allusions, influences, and themes (see Abrams and Harpham (2011) for examples). Second, the collections being studied are usually small, painstakingly assembled, and carefully-curated, and their general subject matter is already known (Watson-Boone 1994; Stone 1982). And third, the researcher does not wish to make a generalization, decision, summary, or overall characterization, but a close and detailed point about a specific topic (see, for example, Eagleton (2011)'s introduction to literary theory).

These scholars are therefore in need of a good solution for making sense of, exploring, and navigating their particular types of text collections, for their particular goals. However, they have been a difficult audience to please. Frischer *et al.* (2006) found that only 6% of scholars were going beyond basic information technology and using computational tools in their research. Even then, the a common 'computational tool' was Google Books, which they used to find out the page numbers on which passages of text they wished to cite appeared. Gibbs and Owens (2012) conducted a large survey to investigate, and concluded that the main culprits were poor user experience, poor documentation, and lack of engagement between tool developers and tool users.

Another recent survey (Schreibman and Hanlon 2010) shed light on why the user experience of digital humanities tools is so poor: it has been neglected at the expense of functional features. The survey was conduced online, and targeted developers of digital humanities tools. It aimed to discover how the pursuit of tool development was percieved as a professional activity within the digital humanities. It asked them how their work fit in with the structure of academic rewards, and whether they thought it was regarded as a valuable scholarly pursuit.

It receivied 108 responses of which 54 were complete. A large majority (67%) of respondents were academic faculty of whom only 12% self-identified as programmers, but 51% collaborated extensively. Of those who collaborated, 85% reported collaborating with programmers, and 80% with other humanities scholars.

The tools seemed to have been built largely to investigate particular researchers' own projects. 96% of tool developers used the 'ability of the tool to do the job it was intended to do' to judge its performance, and 74% judged based on whether it enabled them or others to futher their research. However, usability did not seem to be a major concern:

> The vast majority, 84% (48 respondents), replied that the tools were made available to others. When asked about feedback on the usefulness of the tool as a whole, developers were less systematic. The vast majority of respondents indicated that their main mechanism for feedback was asking colleagues (86%), while 47% obtained feedback via the tool website. Less than a third of the responses (31%) indicated that usability studies were conducted, and even fewer, 14%, utilized surveys. Mailing lists, or more specifically project mailing lists, were frequently cited as a mode for obtaining feedback.

1.2 WordSeer

The work presented here is a systematic attempt at building an exploratory text analysis system for humanities and social science scholars. To avoid the usability problems that plagued previous systems, we took a user-centered design approach to development. All of the tool's analytical features and usability concerns were motivated by feedback from real scholars trying to use our tool to work on real problems.

We began with a single case study with Professor Bryan Wagner of UC Berkeley's English department. He was interested in narrative conventions (Olney 1984) – broad similarities of content and structure – in a collection of North American Slave Narratives digitized by the University of North Carolina (Andrews 2004). The tool we developed was called WordSeer, and its first analytical feature was grammatical search, a way to search over syntactic relationships between words. We built this feature to help identify instances of stereotypes that ordinary keyword search could not precisely identify. It could also be used to discover the language used around query words because it provided a bar chart of the frequency of the top 20 most-frequent words that entered into a relationship with the word. To visualize the prevalence and within-document distribution of stereotypes, we developed a newspaper-column visualization (Eick *et al.* 1992) that could be used to search for structural regularities. The experiment had limited success: we were able to identify a number of stereotypes, and found evidence to suggest that they were not as strictly ordered in the narratives as previously thought. However, the experience was ultimately unsatisfying for Professor Wagner. His feedback made it clear that scholarly work in the humanities and social sciences was much more than hypothesis testing. While such search and visualization features were important, an equal challenge was to support exploration, discovery, and 'getting a sense' of the collection.

To improve WordSeer in this direction, we drew on past work from the fields of sensemaking and exploratory search. We addressed two problems in particular: the vocabulary problem (Furnas *et al.* 1987), which is the great variety of words in which the same concepts can be expressed, and the collection and comparison problem, which is the need to isolate and compare sub-sections of a collection. For the vocabulary problem, we incorporated a way to see related words to a word by clicking on it. These were words that frequently occurred in similar contexts (computed according to the contextual-similarity algorithm in Lin (1997)), as well as the nouns, verbs, and adjectives that frequently appeared in the same sentences. For collection and comparison, we developed a 'Collections' feature that allowed a user to select documents and add them to a collection. These could be used to restrict the outputs of searches and visualizations to just that collection. With these improvements, we were able to use WordSeer to replicate portions of a master's thesis on gendered language in Shakespeare (Froehlich 2012). We found that in plays about love, the words mentioned as being 'hers' centered around body parts: eyes, lips, face, etc. Whereas in plays that were not about love, the words mentioned as being 'hers' were male relatives: father, brother, husband.

However, we still needed to determine whether the improvements we had made were enough to support a basic level of humanities text analysis in a realistic setting. An opportunity for remote, real-world evaluation arrived when WordSeer was used along with four other similar text analysis tools in an undergraduate Shakespeare course at the University of Calgary. Twenty four undergraduate English majors spent the first few weeks of the semester learning how to use the four tools, and the remaining time using them to analyze Shakespeare's *Hamlet*. They posted about their experiences weekly on the class blog (http://engl203.ucalgaryblogs.ca). The students learned to use WordSeer based on instructional videos.

The students' feedback showed that while the new features we had added were useful, sensemaking in the humanities and social sciences had important differences from other fields. One important difference was that the students wanted to analyze text at the word, phrase, and sentence level. WordSeer only supported collecting and comparing documents. Another was that they wanted to isolate and analyze 'slices' of the collection according to combinations of searches and within-document metadata properties, such as Act, Scene, and Speaker – this was also unsupported. They also criticized the user interface. For frequently-performed sequences of action, such as comparison, collecting a set of words and then performing searches, and exploring a new train of thought, they found WordSeer's user interface flows cumbersome and disruptive. For example, in order to perform a comparison, the students needed to open up two browser windows, navigate to WordSeer in each, and type in the different queries they wanted to compare. This was a cumbersome flow, not conducive to quick, easy, exploratory comparison.

To get a better understanding of the text analysis processes of humanities scholars and social scientists, and to identify the operations we needed to support, we conducted informal open-ended conversations with English and History PhD students at UC Berkeley. We asked them to describe their work practices – the activities that constituted a day's work, how they related to each other, their reasons for doing them, and any frustrations they experienced. By summarizing their responses, we were able to devise a model of their text analysis processes, including frequently performed sequences of action, and were also able to identify specific difficulties or 'pain points' that could benefit from computational assistance.

We translated these pain points into concrete design guidelines for text analysis systems and redesigned WordSeer accordingly. The new WordSeer, version 3, supports slicing and dicing the collection according to combinations of searches, filters, and metadata properties. It also supports drilling down, making comparisons, getting overviews, and exploring word usage and context without interrupting the user's current activity.

A final set of three case studies were conducted at UC Berkeley to validate the redesign. An English PhD student analyzed a collection of *New York Times* editorials about China and Japan to find evidence for shifts in the tone of the relationship between the U.S. and China. In the second study, a PhD student in the School of Education analyzed themes and writing styles in a collection of essays written by college students describing their own experiences with literacy. In the third study, a group of one English PhD student and two undergraduates analyzed the shifting connotations of certain words in early American fiction. All users were able to conduct analyses yielding otherwise inaccessible results useful to their research.

1.3 In this Dissertation

In the next chapter, we give some background on sensemaking and describe past work most closely related to ours: previous efforts to create and evaluate sensemaking tools. Next, Chapter 3 describes the first two rounds of formative case studies with WordSeer. These studies yielded important design information, but also revealed differences between other sensemaking domains and the humanities and social sciences. In order to discover more accurate design requirements, we needed to investigate the workflows of our target audience. Chapter 4 gives the results gained from interviews with twelve humanities and social science scholars, which include a detailed model and a step-by-step description of the sequence of activities.

Chapter 5 extracts functionality requirements and design guidelines for text analysis systems based on the processes we discovered. It also describes the user interface and analytical features of the current tool, showing how they address the previously-identified difficulties. The improvements are validated with three case studies in Chapter 6. In these studies, scholars were able to conduct investigations relevant to their reseach; the tool helped them discover useful and otherwise-inaccessible information.

Chapter 7 describes controlled experiments we conducted on aspects of the user interface and the search algorithm. Finally, Chapter 8 outlines directions for future research and summarizes our contributions .

Technical implementation details and links to the open-source code repositories and documentation are in Appendix A.

Chapter 2

Background

2.1 Sensemaking

When the term sensemaking was first introduced, it was a general way to describe how humans transformed their sensory input into structured knowledge. The first model of the process was introduced by Weick (1979), who explicitly distinguished it from the related idea of *interpretation* and emphasized its cyclical nature (Weick 1995). The process consisted of three steps called 'enactment', 'selection' and 'retention' from Weick (1979) (see Figure 2.1), made concrete by fitting several real-world situations to the model (see, for an example, 'meeting a stranger' in Figure 2.2).

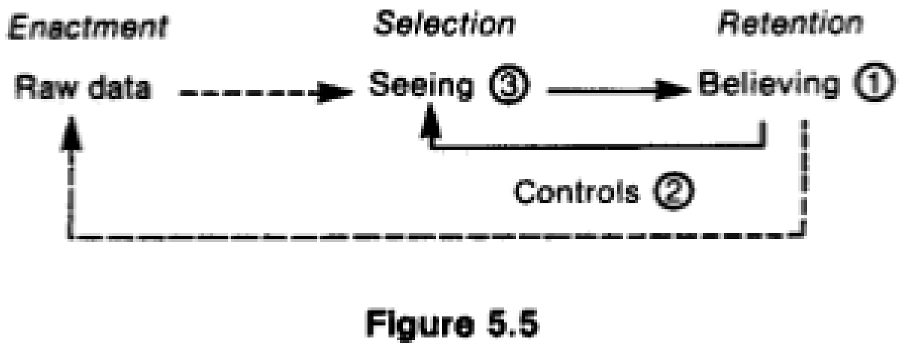


Figure 2.1: The sensemaking cycle according to Weick (1979) pp 133. The steps are illustrated with examples in Figure 2.2.

A decade later, Russell *et al.* (1993) applied this term specifically to the process of making sense of a collection of documents. Even so, their model (Figure 2.3) was equally general, consisting of the phases 'collect data', 'search for representations' and 'instantiate representations'.

Pirolli and Card were the first to map these general processes to more specific steps,

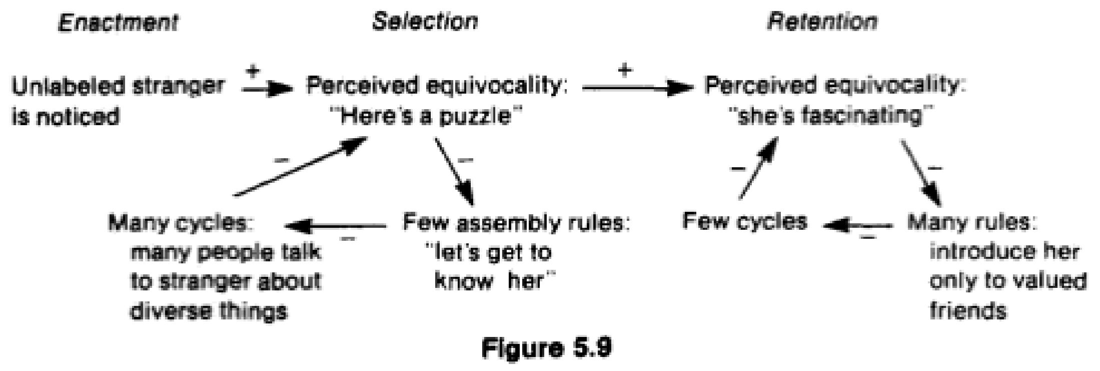

Enactment
Selection
Retention
Unlabeled stranger is noticed
+
Perceived equivocality: "Here's a puzzle"
+
Perceived equivocality: "she's fascinating"
–
–
–
–
Many cycles: many people talk to stranger about diverse things
–
Few assembly rules: "let's get to know her"
Few cycles
–
Many rules: introduce her only to valued friends
Figure 5.9

Figure 2.2: Weick's sensemaking cycle as applied to the situation of meeting a stranger.

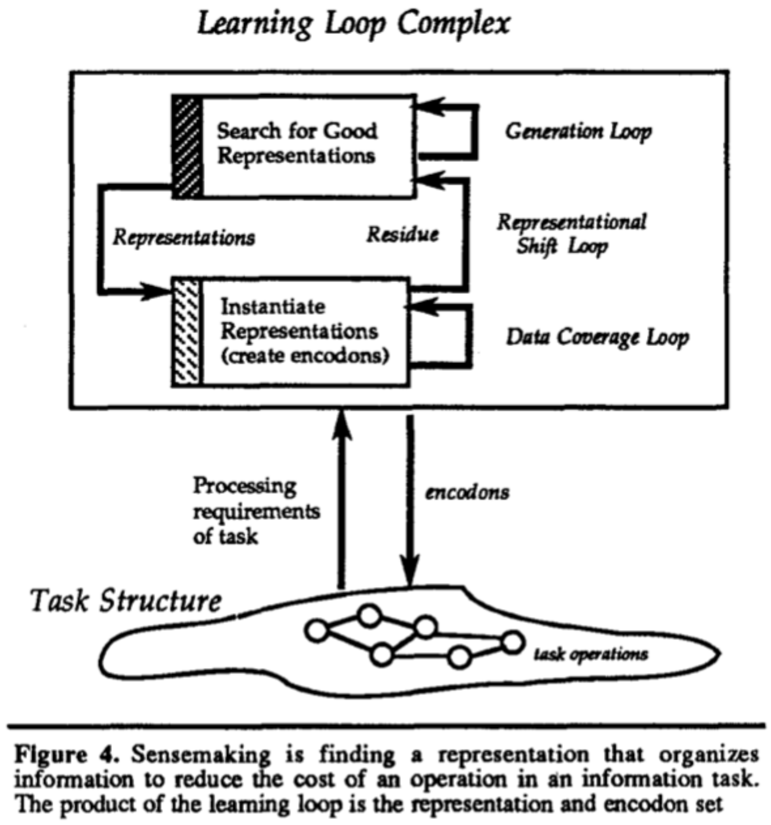

Learning Loop Complex
Search for Good Representations
Generation Loop
Representations
Residue
Representational Shift Loop
Instantiate Representations (create encodons)
Data Coverage Loop
Processing requirements of task
encodons
Task Structure
task operations
Figure 4. Sensemaking is finding a representation that organizes information to reduce the cost of an operation in an information task. The product of the learning loop is the representation and encodon set

Figure 2.3: The sensemaking cycle as it appears in Russell *et al.* (1993)

leading to concrete suggestions for tool builders. In order to do so, they specialized to a specific sensemaking activity: intelligence analysis (as conducted by government agencies and market researchers). The model they devised is shown in Figure 2.4.

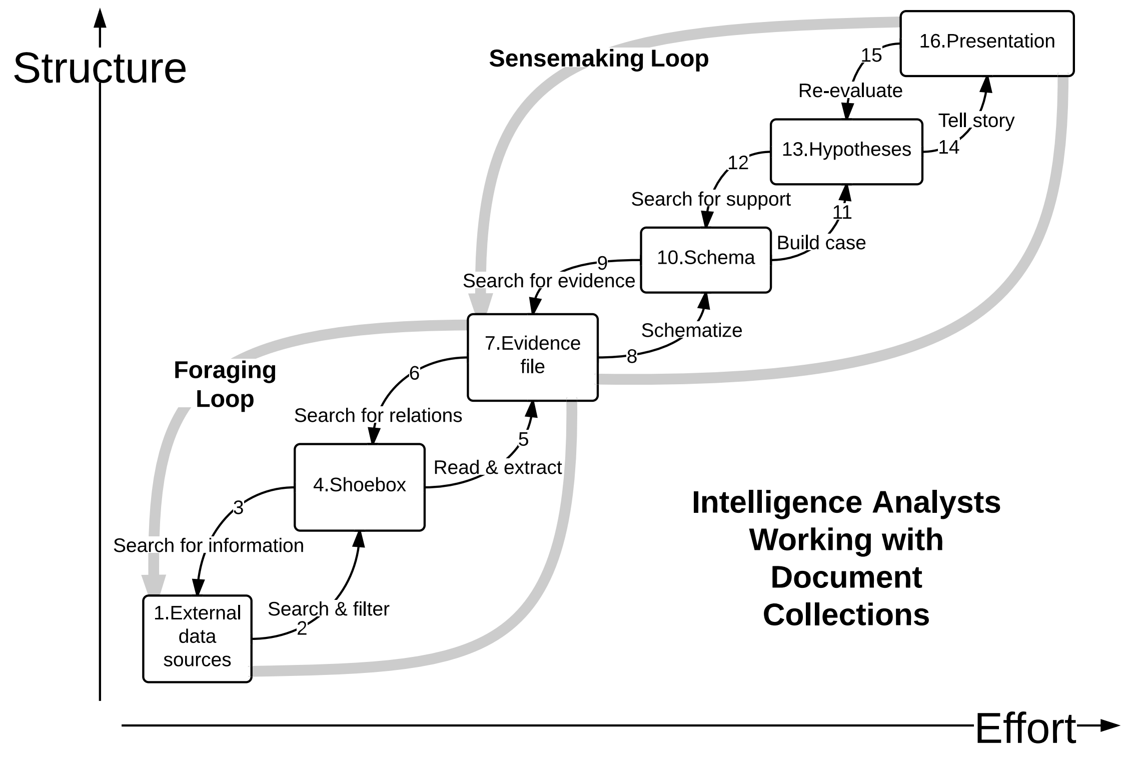


Figure 2.4: Pirolli and Card's sensemaking process for intelligence analysts

They found that analysts experienced the following difficulties:

> 'Our analysts spent considerable time scanning data seeking relevant entities (names, numbers, locations, etc.). The assessment of whether or not an item is relevant also takes time. Techniques for highlighting important information with pre-attentive codings, or re-representing documents (e.g., by summaries) appropriate to the task can improve these costs.'

To alleviate these difficulties, they suggested that,

> 'Techniques aimed at expanding the working memory capacity of analysts by offloading information patterns onto external memory (e.g., visual displays) may ameliorate these problems.'

These studies and others have yielded general recommendations, summarized by Hearst (2009a):

> A more supportive search tool would give an overview of the contents of the collection, would help the analysts keep track of what they had already viewed, would suggest what to look for next, would encourage analysts to try new queries, and would find additional documents similar to those already found, but would distinguish duplicates. Studies also suggest that analytical search tools should allow for aliasing of terms and concepts.

2.2 Tools that Support Sensemaking

Hearst (2009b) describes the history of efforts to create sensemaking tools. While there are none that support the entire process, individual parts have received attention, and some are now attempting to knit together the *seams of sensemaking* (Hearst and Degler 2013).

In the realm of *information triage* (Buchanan and Loizides 2007; Jonker *et al.* 2005; Badi *et al.* 2006), which is the process of prioritizing, organizing, and reviewing collected pieces of information, there have been many tools that used the canvas metaphor (Robertson *et al.* 1998; Marshall *et al.* 1994; Malone 1983). Drawing from the knowledge that people tend to use spatial layout for organizing information both in electronic spreadsheets and on paper (Nardi 1993; Malone 1983; Marshall and Shipman III 1995; Mander *et al.* 1992), these interfaces provide users a visual space on which clippings of data, intermediate sets of documents, and even the output of computational analysis can be rearranged, grouped and labeled in spatial relation to each other. In the past, these organization tools have focused on helping users tackle information that has already been collected. In a recent advance, Hearst and Degler (2013) presented an effective interface that integrated information organization with the search process, allowing users transition seamlessly between gathering documents and organizing them to reflect emerging structures. They also proposed general design guidelines for future systems.

Another process that has received attention is hypothesis generation. The Sandbox (Wright *et al.* 2006) and its companion information retrieval system, TRIST (Jonker *et al.* 2005), lets users see how a certain concept had been reasoned about in the past and provides templates that the users can fill in with evidence relevant their particular case. TRIST also provides compelling interfaces to help analysts select a subset of documents from a large collection for further scrutiny, including extracting important entities and organizing retrieved documents into topics, but with less emphasis on analyzing text at the word level.

Exploratory data analysis (EDA) is an approach popularized by the statistician John Tukey whose goals are to maximize insight into a data set, uncover underlying structure, test underlying assumptions, and find bases from which to formulate hypotheses to test with confirmatory approaches (Tukey 1977, 1980). Many of the tools developed for EDA such as Polaris (which because the product Tableau) (Stolte *et al.* 2002), and Spotfire (Ahlberg 1996) allow users to view multidimensional data from different angles by selecting subsets of data, viewing those as visualizations, moving laterally to view other subsets of data, moving into another view, expanding the viewed data by relaxing constraints, and so on. However, these tools operate over numerical and categorical data, but do not seamlessly operate over raw textual information with the same flexibility.

Early tools for exploratory data analysis such as Protofoil (Rao *et al.* 1994) made first steps into linking text search, category search, grouping, and clustering. More recent tools such as Jigsaw (Gorg *et al.* 2012), Takmi (made into the product IBM TextMiner) (Uramoto *et al.* 2004), QDA Miner Lewis and Maas (2007) and SAS Text Analytics (Matignon 2007) integrate analysis techniques like text classification, named-entity recognition, sentiment analysis and summarization into one interface. To make the results of these computational analyses interpretable, they display the outputs with interactive data visualization. These systems have made advanced text mining and visualization algorithms available to users without expertise in those areas.

There is active research around the problem of mining, and creating visual interfaces to interact with text-based news. Some recent visual interfaces include those by Krstajic *et al.* (2010b) and Lloyd *et al.* (2005), who gave ways to mine and visually explore entities and their mentions over time, and Leskovec *et al.* (2009); Schneider *et al.* (2010) and Krstajic *et al.* (2010a) who give ways to visualize and explore topics over time, and Bernstein *et al.* (2010) who created Eddi, a comprehensive visual interface for managing one's twitter stream.

There are also a number of exploratory tools for bibliographic citations, including Paperlens (Lee *et al.* 2005) and Apolo (Chau *et al.* 2011). These provide access to networks of co-authorship, and relations among metadata categories, but do not provide rich access to the text itself.

2.3 Text Analysis Tools for the Humanities and Social Sciences

As we discuss in Chapters 3 and 4, there are differences between the workflows of intelligence analysts and scholars in the humanities and social sciences. These differences are driven by the fact that arguments in the humanities and social sciences are more qualitative, and based on different kinds of evidence. In addition to chaining together sequences of events, or facts and figures, these scholars chain together interpretations. Their arguments rest on what was said, as well as the language used to say it. Intelligence analysts work from large collections of short documents, whereas in the humanities and social sciences, small collections of longer documents also come into play.

Currently, we know of no other sensemaking tools for this domain. The closest are visual text analysis tools, aimed at answering a specific question with a visualization. These usually have two parts. First, they apply some form of natural language processing to extract aggregate statistics about word usage, named entities, and parts of speech. Second, they display the extracted information with visualizations such as word clouds, node-and-link diagrams, and concordances.

One of the first visual text analysis systems in this area was Compus (Fekete and Dufournaud 2000), which allowed users to visually explore a collection of 16th-centry legal documents. Each document was represented as a long rectangular column, side-by-side with all the others. The user could search for items of interest, and have them highlighted as smaller sub-rectangles in the long document columns. This created a visualization showing the prevalence of that item in the entire collection. Users could filter documents and sort them by date, title, and other metadata.

At present, there are three well-known text analytics efforts for the humanities and social sciences. However, none support the essential sensemaking functions of browsing and filtering a collection, exploring related items, and isolating subsets for analysis. They do not perform computational linguistic analysis of sentence structure and therefore do not offer related word exploration or grammatical search. None were evaluated with usability studies.

The first is the now-inactive MONK project (http://monkproject.org) incorporating the SEASR analysis toolkit (Llora *et al.* 2008). This project offered two computational linguistics tools in addition to word distribution and frequency statistics: tagging words with their parts of speech and extracting named entities. Users could visualize occurrence patterns of word sequences within a chosen text, and plot networks of how often named entities occur near each other. This research led to visual text-mining analyses of Emily Dickinson's correspondence (Plaisant *et al.* 2006), and of Gertrude Stein's "The Making of Americans" (Clement 2008) and an interface for exploring the parts of speech used near query words of interest (Vuillemot *et al.* 2009).

The second is Voyant (Rockwell *et al.* 2010; Rockwell 2003a), which operates entirely at the word level. It allows users to plot word frequencies, see concordances (contexts in which words occur) and create tag clouds. For visualization of word prevalence, there is a single scatter plot. It shows users whether query words occur towards the beginnings or the ends of documents, but not how common the words are overall, or whether they are more prevalent in certain documents than others.

Third, WordHoard (Mueller 2008), and the associated MorphAdorner program (Mueller 2009) focus on morphology and word forms. Using side-by-side tables, but no visualizations, WordHoard allows users to compare the frequencies and contexts of words across parts of speech, and various different categories of text. It relies on rich metadata, such as (in Shakespeare) speaker gender and sentence type.

More recently, there is the Subjunctive interface (Bron *et al.* 2012), which is designed for Media Studies researchers and allows for two side-by-side comparisons of text content, but only with a limited set of views (bar charts and line graphs of frequencies along with word clouds).

Other projects have used more advanced language processing, but have not developed them into user interfaces or combined them with visualizations. Topic modeling is being applied to 19th Century British and American novels (Jockers 2011). These novels were also the subjects of recent research that showed how to automatically extract social networks from free text (Elson *et al.* 2010).

2.4 Evaluating Sensemaking Systems

Evaluating sensemaking systems is an active research area, because sensemaking is an unpredictable task with subjective outcomes. This is for several reasons. First, sensemaking workflows have mainly been studied in the context of intelligence analysis, and are poorly understood in other disciplines, making user behavior difficult to understand. Second, success is hard to measure because the goal is to 'get a sense' of a collection of data, or to discover 'something new' about it. Because of this, there may be multiple ways to achieve success, and the path from start to finish may look very different from person to person. Finally, these systems tend to be highly interactive and incorporate many different analysis tools, so it may be difficult to disentagle the ways in which the different components of a system affect the final outcome (White *et al.* 2008).

Since 2008, when White *et al.* (2008) organized the first workshop on evaluating exploratory search systems, some consensus has grown. As in other fields of human-computer interaction, evaluation methods fall into two categories, depending on whether the goal is summative or formative. Summative evaluations are intended to summarize the effectiveness of an approach. In other areas, highly-controlled, comparative experiments that measure user behavior in quantitative ways are the norm. Summative evaluations are typically useful in the later stages of product design, when they can help distinguish between fine-grained design differences, or help make purchasing decisions easier. By contrast, formative evaluations are conducted in the early stages of development. Instead of yielding quantitative measurements for performance summaries, these are qualitative studies intended to provide guidance on the overall direction of development. The goals of these studies are to discover users' needs and existing behavior patterns, and to test the appropriateness of different approaches to support them.

Since sensemaking systems (or exploratory search systems, or exploratory data analysis systems) are a relatively new field of research, summative evaluations are rare (Qu and Furnas 2008). The most closely related type of summative evaluation is to study performance on a somewhat open-ended task (such as writing an essay on a topic) on a *search* or *browsing* interface. Evaluations tend to take the form of laboratory studies in which participants are asked to complete a pre-defined task within a given amount of time, usually a few hours. There are guidelines on how to design tasks that will elicit elicit exploratory search behavior in these settings (Wildemuth and Freund 2012; Kules and Capra 2009). At the end, participants produce a summary or some other set of results, which are evaluated by human judges. The self-reported levels of satisfaction of participants are also used as a measure.

Formative evaluations are more common in the field at this stage (Lam *et al.* 2012; Qu and Furnas 2008). Lam *et al.* (2012) conducted a comprehensive literature review and found three main ways of conducting formative evaluations. These are, first, long-term case studies conducted with actual practitioners in their own environments, second, interviews with practitioners to gain knowledge about workflows and current practices, and, third, controlled experiments to refine specific system components.

2.5 Evaluating WordSeer

We used all of these formative evaluation methods to guide the development of our tool, WordSeer. Our goal was to reach a point where the tool was successful in a real-world setting. In other words, an active scholar in the humanities or social sciences should be able to use it (after instruction) to investigate a research question of current interest to them, and be able to discover otherwise-inaccessible knowledge of value to them in their work.

We conducted three rounds of case studies, one after each iteration of design. We achieved our goal after the third iteration. We followed the Multi-dimensional In-depth Long-term Case-study (MILC) evaluation method proposed by Shneiderman and Plaisant (2006), which is now widely adopted as a way of conducting formative case studies with exploratory data analysis tools (Lam *et al.* 2012). Feedback from each round of case studies was used to develop new features and to refine existing functionality.

The first case study was a collaboration with Professor Bryan Wagner of UC Berkeley's English department. As a result, we incorporated our first exploratory features, and added ways to isolate sets of items for analysis. The second round of case studies occurred as part of a semester-long undergraduate class. WordSeer 2 was used along with four comparative text analysis tools produced by other researchers in a class to introduce students to literary analysis of Shakespeare. Groups of students used the tools to analyze Shakespeare's *Hamlet* and reported on their experiences weekly on the class blog. These case studies are described in more detail in Chapter 3.3.

After the class, we realized we could no longer operate based on the design recommendations of Pirolli and Card's sensemaking model, which was based on intelligence analysts. We needed more detailed, specific knowledge of the text analysis process in the humanities and social sciences. We therefore conducted interviews with twelve English and History PhD students at UC Berkeley to gather this information. Our findings are described in Chapter 4, and the resulting design principles in Chapter 5.

After gathering this information, we redesigned WordSeer for the third time. The resulting system, WordSeer 3, is described in Chapter 5.The following three case studies all acheived the goal of allowing a scholar to discover useful, and otherwise inaccessible knowledge on a research question of existing interest. These case studies are described in Chapter 6.

At this point, we had established the usefulness of showing grammatical relationships between words, and discovered a need for finding sentences based on a set of examples. To refine these interface components, we conducted controlled experiments. We conducted a 400-person Amazon Mechanical Turk study on how to display syntactic relationships between words in an easily recognizable way. We also conducted a controlled experiment to measure whether relevance feedback was more effective than ordinary keyword search at collecting sentences matching a given literary theme based on examples. These experiments are described in Chapter 7.

Chapter 3

Formative Case Studies

3.1 Case Study 1: Stereotypes in North American Slave Narratives

3.1.1 Motivation

The first case study was an exploratory collaboration with Professor Bryan Wagner of UC Berkeley's English department. Our focus was specifically on comparing texts, getting a sense of stylistic and thematic similarities, and tracing patterns of language use. These tasks were not widely supported by any current software, but if humanities researchers wanted to use digitized text collections on a larger scale, they would need to do exactly such things.

We decided to analyze a collection of interest to Professor Wagner, a selection of North American pre-civil-war slave narratives digitized by the University of North Carolina (Andrews 2004). These are a collection of around 300 works written by fugitive slaves in the decades before the Civil War with the support of abolitionist sponsors. Scholars agree about the slave narrative's most basic conventions (Andrews 1988) but it is likely that these narratives, with their extreme repetitiveness, have other regularities that have yet to be detected. We aimed to uncover these patterns with computer-assisted techniques.

We investigated a specific type of regularity: Olney (1984)'s so-called *master plan* for slave narratives. In his 1984 work on the subject, James Olney set out a number of narrative stereotypes that, he asserted, were "so early and firmly established that one can imagine a sort of master outline drawn from the great narratives and guiding the lesser ones". His master plan began as follows:

1. a first sentence beginning "I was born...", then specifying a place, but not a date of birth;

2. a sketchy account of parentage often involving a white father;

3. a description of a cruel master, mistress, or overseer, details of first observed whipping and numerous subsequent whippings, with women very frequently the victims;

4. an account of one extraordinarily strong, hardworking slave – often “pure African” – who, because there is no reason for it, refuses to be whipped;
5. record of the barriers raised against slave literacy and the overwhelming difficulties encountered in learning to read and write;

And continues on to:

> 10. description of successful attempt(s) to escape, lying by during the day, travelling by night guided by the North Star, reception in a free state by Quakers who offer a lavish breakfast and much genial thee/thou conversation;

And so on, in this specific manner. Our question was simple, how true is this assertion? Are these narrative conventions really present in all the narratives? Are there narratives that do not match these patterns? Are there some conventions more “conventional” than others?

We chose this area of exploration because it satisfied two criteria: first, it was interesting to the literary scholar we were collaborating with, and not just a task created to demonstrate the computational powers of a tool. Second, it would be a novel problem in the field of text analysis systems, and not just a simple application of previous tools.

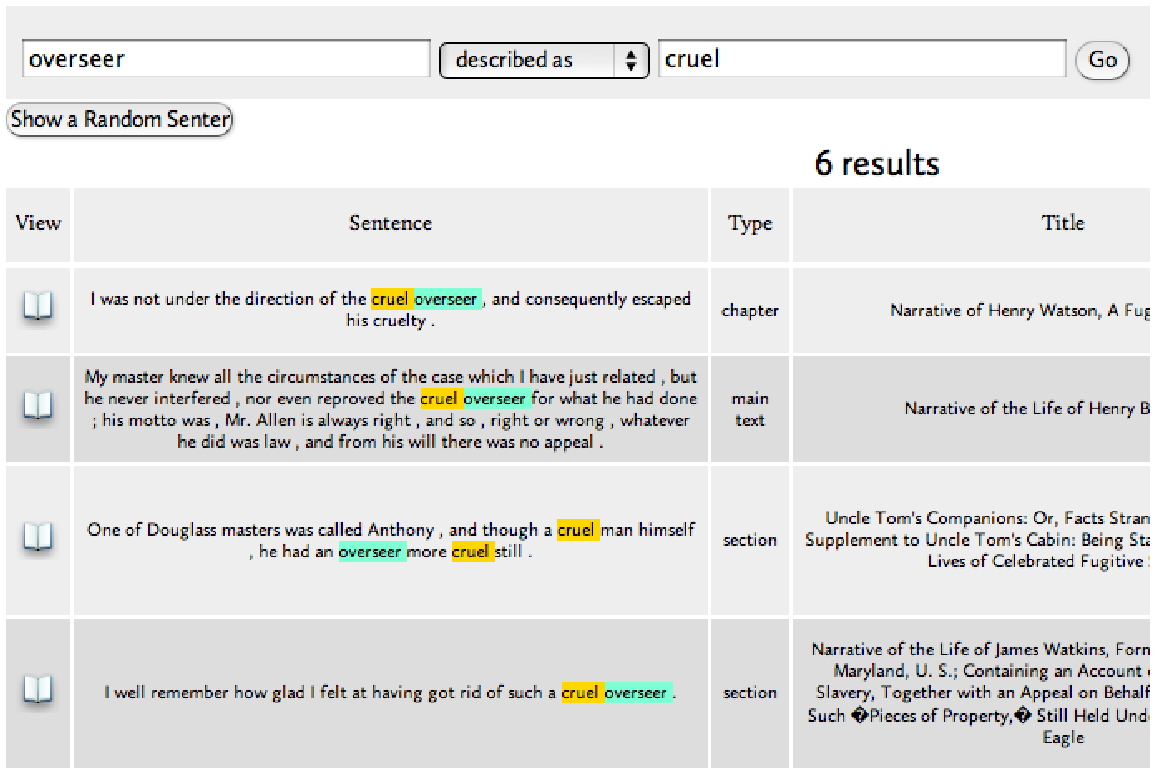
We proceeded with the following two-phase plan of action: Phase 1, find examples of the stereotype. We would develop a way to gather examples of stereotypes, and attempt to translate these instances of stereotypes into some representation in terms of words and relationships between them. In Phase 2, we would develop a computational tool for examining the prevalence of any particular sterotype, and for comparing different sterotypes. We would use this tool to examine the patterns of occurrence of the various computationally-represented stereotypes, and draw conclusions about Olney’s master plan.

We called the system we built WordSeer version 1.

3.1.2 Phase 1: Grammatical Search

We were in need of a good way to search the narrative collection for stereotypes. Keyword search is an approximate way of searching. This is because our true information need cannot always be expressed with a fixed set of words or phrases. When searching for descriptions of cruel overseers, the words “cruel”, “harsh”, and “overseer” come to mind. However, simply typing the words “cruel harsh overseer” does not capture our true information need, which is more precisely stated as: text in which an overseer *is described as* cruel or harsh.

To address this problem, we implemented grammatical search. Existing natural language processing technology has made it possible to automatically extract grammatical relationships between words (Jurafsky and Martin 2009). For example, in the sentence “The good God has given every man intellect”, it is possible, using freely-available tools, to automatically extract that the adjective “good” is being applied to the word “God”, and that “God” is the agent of the verb “give”. The process of extracting this information from text is called *dependency parsing*, and the relationships between words are called *dependencies.*



6 results

View	Sentence	Type	Title
📖	I was not under the direction of the cruel overseer , and consequently escaped his cruelty .	chapter	Narrative of Henry Watson, A Fug
📖	My master knew all the circumstances of the case which I have just related , but he never interfered , nor even reproved the cruel overseer for what he had done ; his motto was , Mr. Allen is always right , and so , right or wrong , whatever he did was law , and from his will there was no appeal .	main text	Narrative of the Life of Henry B
📖	One of Douglass masters was called Anthony , and though a cruel man himself , he had an overseer more cruel still .	section	Uncle Tom's Companions: Or, Facts Stran Supplement to Uncle Tom's Cabin: Being Sta Lives of Celebrated Fugitive
📖	I well remember how glad I felt at having got rid of such a cruel overseer .	section	Narrative of the Life of James Watkins, Forn Maryland, U. S.; Containing an Account Slavery, Together with an Appeal on Behalf Such �Pieces of Property,� Still Held Und Eagle

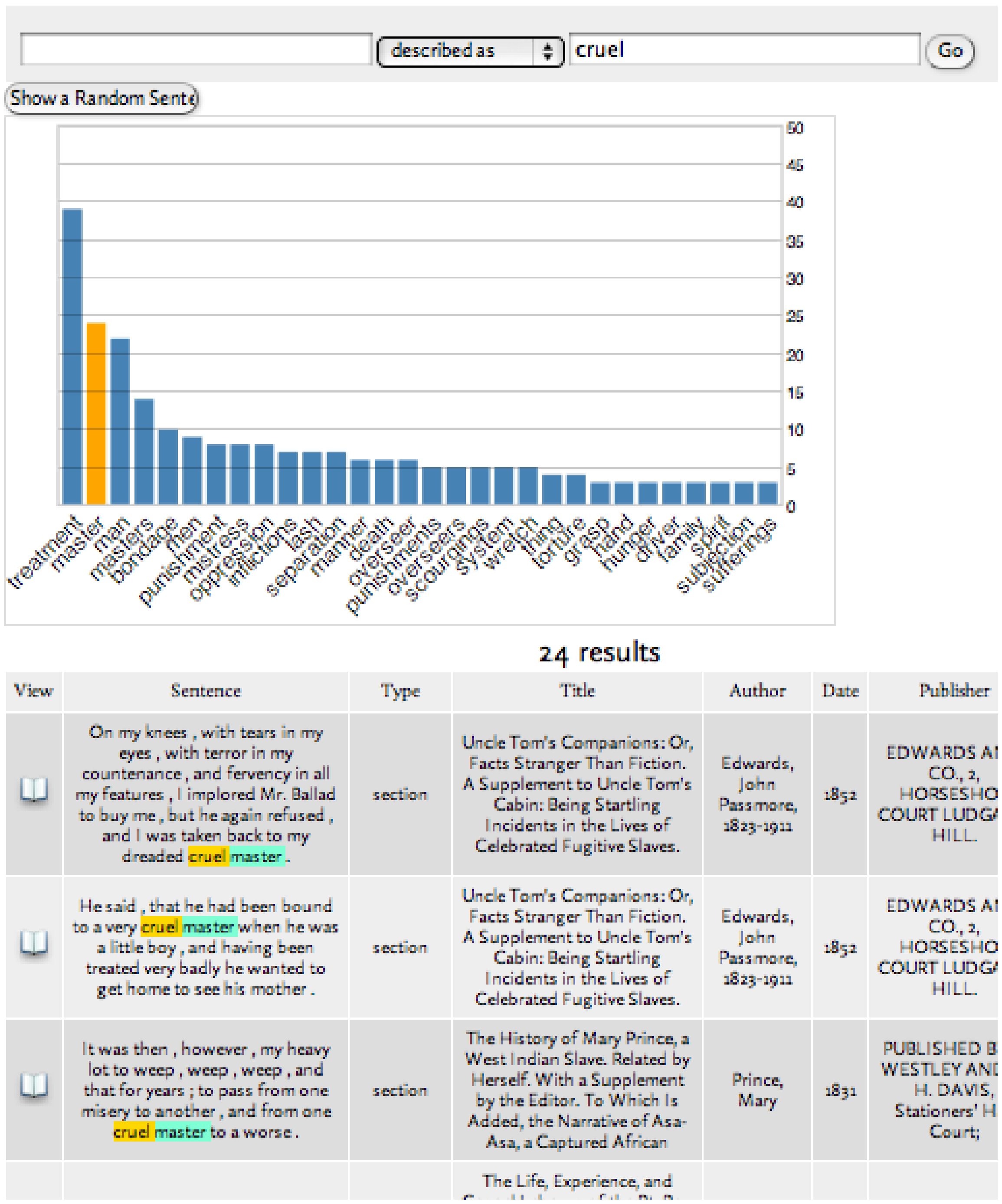
Figure 3.1: Grammatical search for the cruel treatment stereotype.

We created a search engine based on this technology. We performed dependency parsing on the entire slave narratives collection and created a way for a scholar to specify the grammatical relationships between words. Grammatical search allows users to be precise about the relationships between query words.For example, in Figure 3.1, instead of just typing in 'overseer cruel' to retrieve sentences in which 'overseer' is described as 'cruel', the scholar directly specifies that the amod (adjective modifier) relationship should exist between 'overseer' and 'cruel'. Figure 3.1 shows the results. Even though there are only six, each result precisely represents an instance of an overseer being described as cruel.

Grammatical search can also be used for discovery. Leaving one side of the search box blank returns all the words that match the query in the form of a bar graph. For example, as shown in Figure 3.2 , we can search for all the "sources of cruelty" by leaving the first box blank and issuing the query "___ (described as) cruel". This returns all the words that are modified by the adjective "cruel", arranged in order from most to least frequent.

The top 20 words that most frequently described as 'cruel' are displayed in an interactive graph. In the slave narratives, there are close to 40 instances of "cruel treatment", followed by "cruel master", "cruel man", and "cruel masters". Clicking on a word in the graph filters the result set to match only that word. In the figure, the results have been filtered on the word 'master' – zooming in on the 24 instances of "cruel master".

This type of analysis would be extremely time-consuming in other tools developed for



24 results

View	Sentence	Type	Title	Author	Date	Publisher
📖	On my knees , with tears in my eyes , with terror in my countenance , and fervency in all my features , I implored Mr. Ballad to buy me , but he again refused , and I was taken back to my dreaded cruel master .	section	Uncle Tom's Companions: Or, Facts Stranger Than Fiction. A Supplement to Uncle Tom's Cabin: Being Startling Incidents in the Lives of Celebrated Fugitive Slaves.	Edwards, John Passmore, 1823-1911	1852	EDWARDS AI CO., 2, HORSESHO COURT LUDGA HILL.
📖	He said , that he had been bound to a very cruel master when he was a little boy , and having been treated very badly he wanted to get home to see his mother .	section	Uncle Tom's Companions: Or, Facts Stranger Than Fiction. A Supplement to Uncle Tom's Cabin: Being Startling Incidents in the Lives of Celebrated Fugitive Slaves.	Edwards, John Passmore, 1823-1911	1852	EDWARDS AI CO., 2, HORSESHO COURT LUDGA HILL.
📖	It was then , however , my heavy lot to weep , weep , weep , and that for years ; to pass from one misery to another , and from one cruel master to a worse .	section	The History of Mary Prince, a West Indian Slave. Related by Herself. With a Supplement by the Editor. To Which Is Added, the Narrative of Asa-Asa, a Captured African	Prince, Mary	1831	PUBLISHED B WESTLEY ANI H. DAVIS, Stationers' H Court;
			The Life, Experience, and			

Figure 3.2: The most frequent 20 words described as cruel, filtered (by clicking) on the word "master". The visualization creates an explorable picture of the sources of cruelty in a slave's life as reflected by the narratives in the collection.

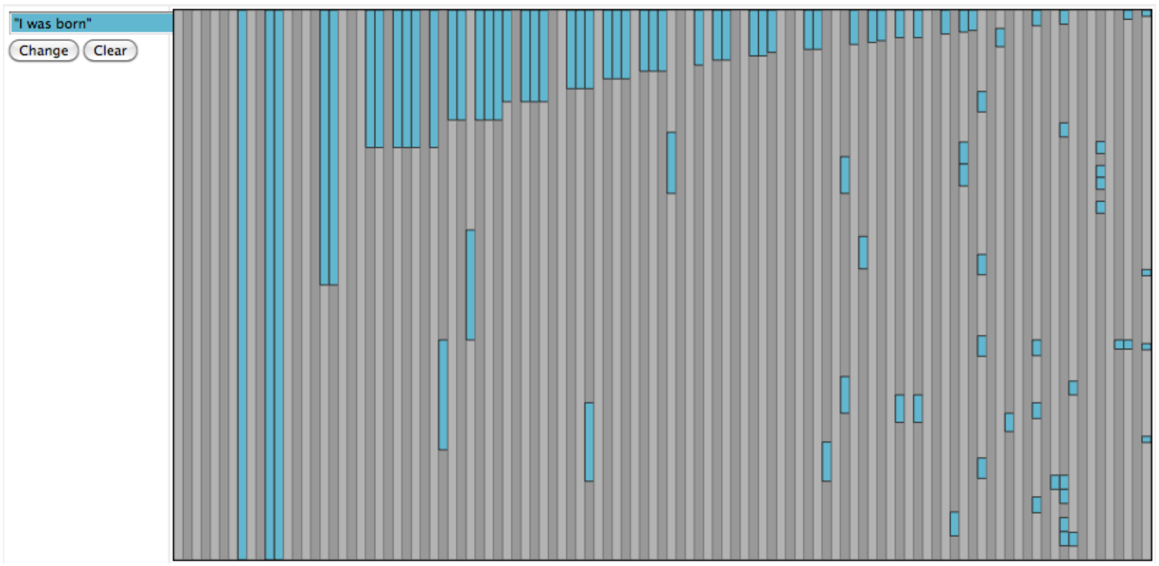


Figure 3.3: The distribution of the stereotype "I was born".

humanities scholars like MONK Clement *et al.* (2009) and Voyant Rockwell *et al.* (2010). For example, in either of these systems, searching the slave narratives for sources of cruelty in a slave's life would entail keyword search with terms like "cruel", and guesses like "overseer", "punishment", and "master". These systems return an unstructured list of results that is fairly uninformative. They provide no automated way to discover the vocabulary of the collection – what are all the things described as cruel? – and do not offer overviews of the collection-wide prevalence and predominant usage contexts of a set of terms.

3.1.3 Newspaper-column Visualization

The ability to investigate the prevalence of a stereotype was central to the second phase of our analysis. To this end, we developed a visualization of the entire collection (Figure 3.3) using the *newspaper column* visualization (Eick *et al.* 1992; Fekete and Dufournaud 2000). Each vertical column is a narrative. The narratives are segmented into vertical blocks corresponding to 30 sentences each. A block is highlighted if the term occurs anywhere within those sentences.

Essentially, this amounts to arranging the narratives side by side, and highlighting occurrences of the query in a given color. Such visualizations are popular in text analysis interfaces when allowing a user to visualize the distribution of a search query.

For example, Figure 3.3, shows the distribution of the exact phrase "I was born". It is immediately apparent that this phrase occurs mostly towards the beginnings of narratives. Compared with the stereotype about north-star-guided escapes (Figure 3.4), it also seems much more prevalent.

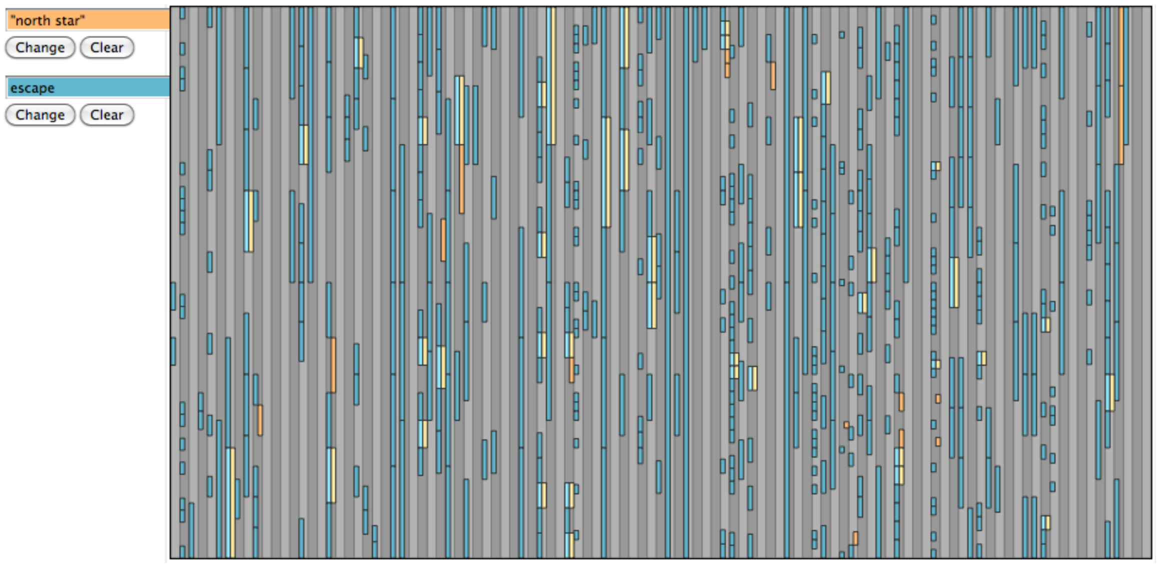


Figure 3.4: Stereotypes compared. The visualization shows the occurrences of the "I was born" stereotype (blue) and the "North Star" guided escape stereotype (yellow). The first is relatively more prevalent, and the two do not occur in the same places.

3.1.4 Prevalent Stereotypes

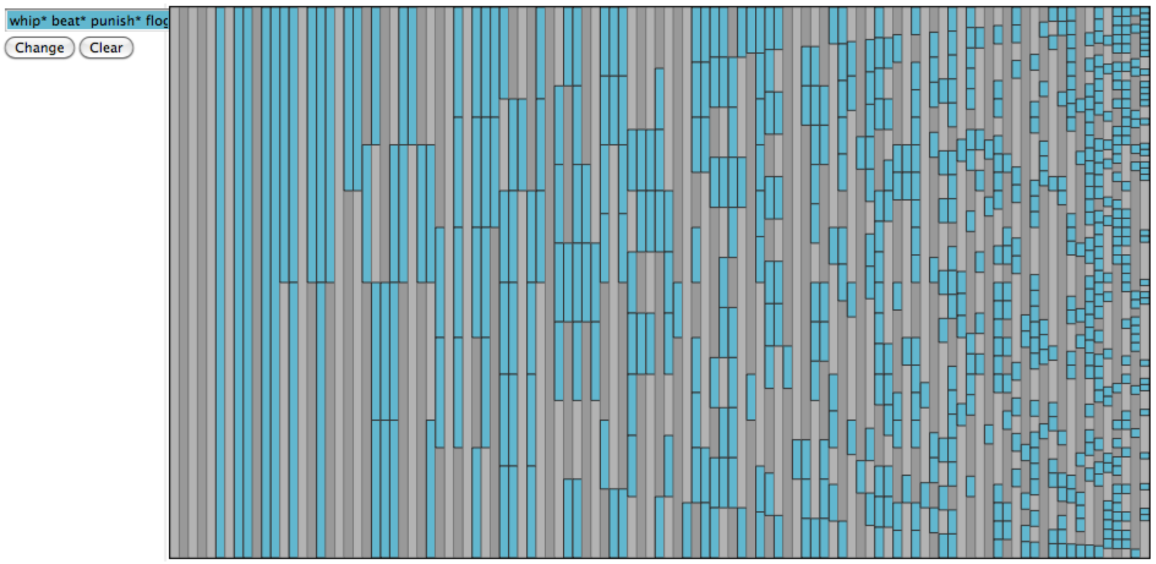
Professor Wagner was able determine that there were indeed certain events that occurred so frequently in the collection as to be rightly called stereotypical. The first of these was the "cruel treatment" stereotype. Of the many listed by Olney, it was also the easiest to search for. As shown in Figure 3.5a, almost all the narratives had multiple occurrences of the simple keywords: "punish", "beat", or "whip". Such events were not restricted to the few narratives Olney mentioned by name. Grammatical search (Figure 3.5b) reveals the other side of the stereotype. The numbers on the graph confirm the high prevalence revealed by the newspaper-column visualization, but the adjectives accompanying these actions paint a more complete picture: one of severe, cruel, and inhuman treatment.

Other similar stereotypes were separation from parents (Figure 3.6), escape (Figure 3.7), and the "I was born" stereotype (Figure 3.3).

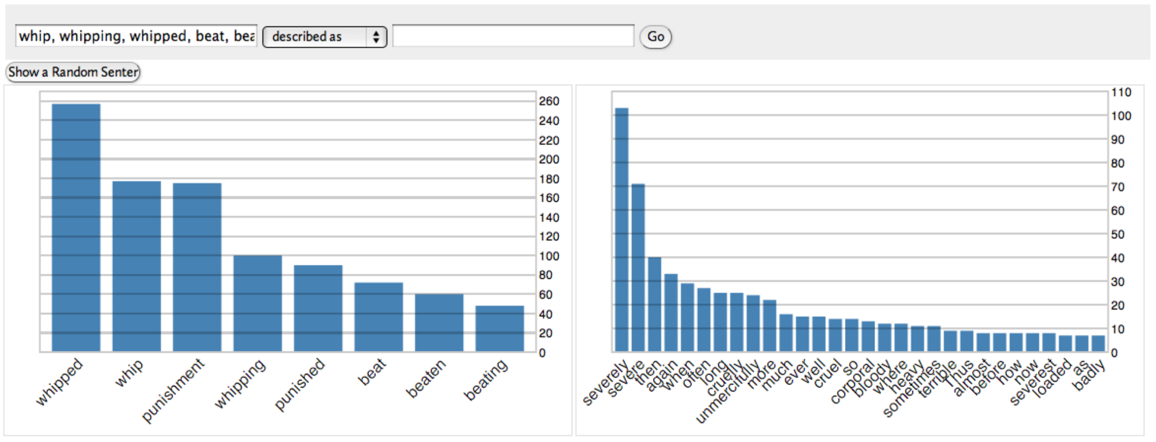
3.1.5 Less Prevalent Stereotypes

He was also able to identify at least two sterotypes that did not appear to be as prevalent in the collection as implied by Olney's language: those of escapes guided by the north star, and of being received by Quakers.

As already shown in comparison with "I was born", mentions of the north star (Figure 3.4) do occur in some narratives. Hovering over the highlighted blocks shows the matching text, with a link to view the sentence in the original context. Upon reading these mentions and their contexts, we found that almost all occurrences of the "north star" are indeed related to



(a) Widespread usage of words related to punishment.



(b) Widespread description of punishments in cruel terms.

Figure 3.5: Evidence for the cruel treatment stereotype

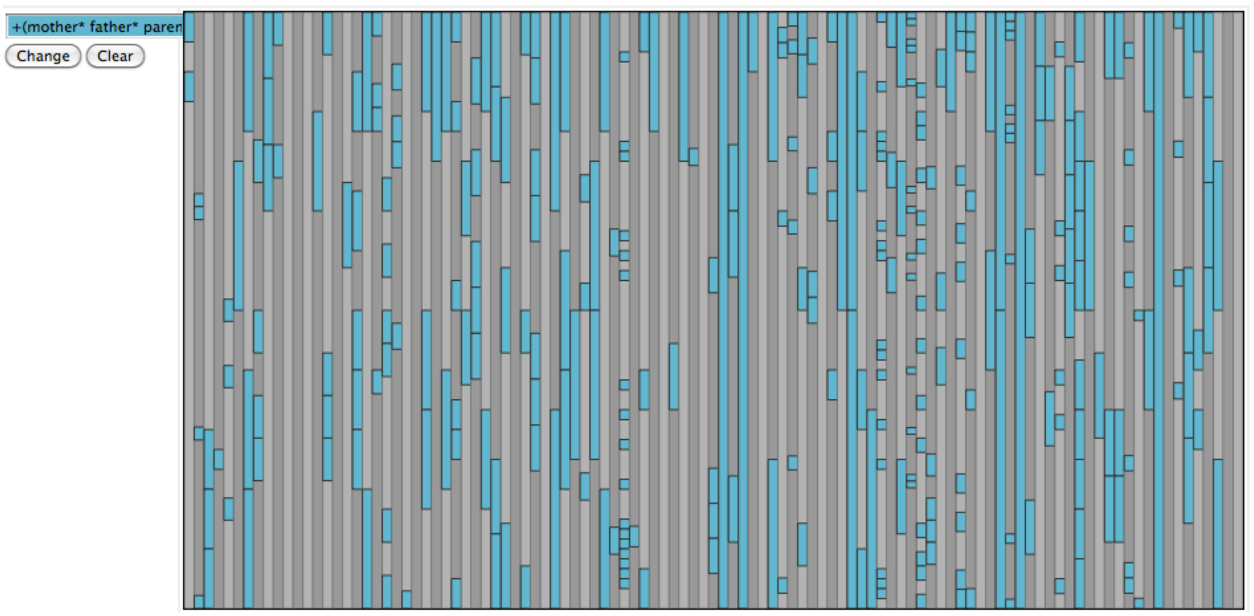


Figure 3.6: Separation from parents.

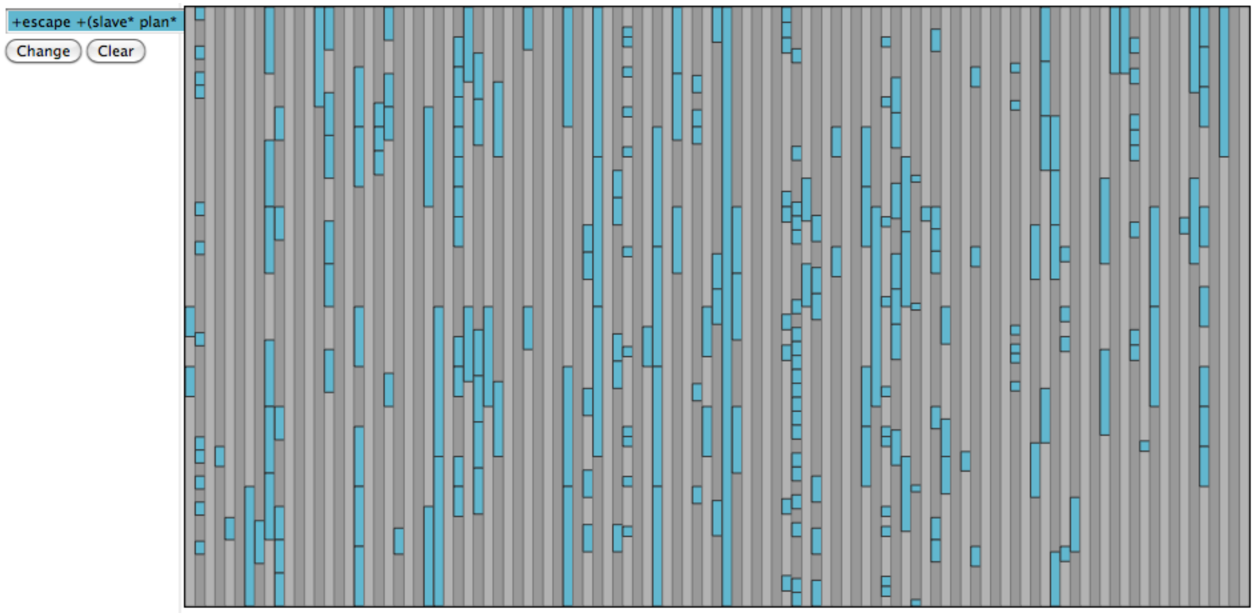
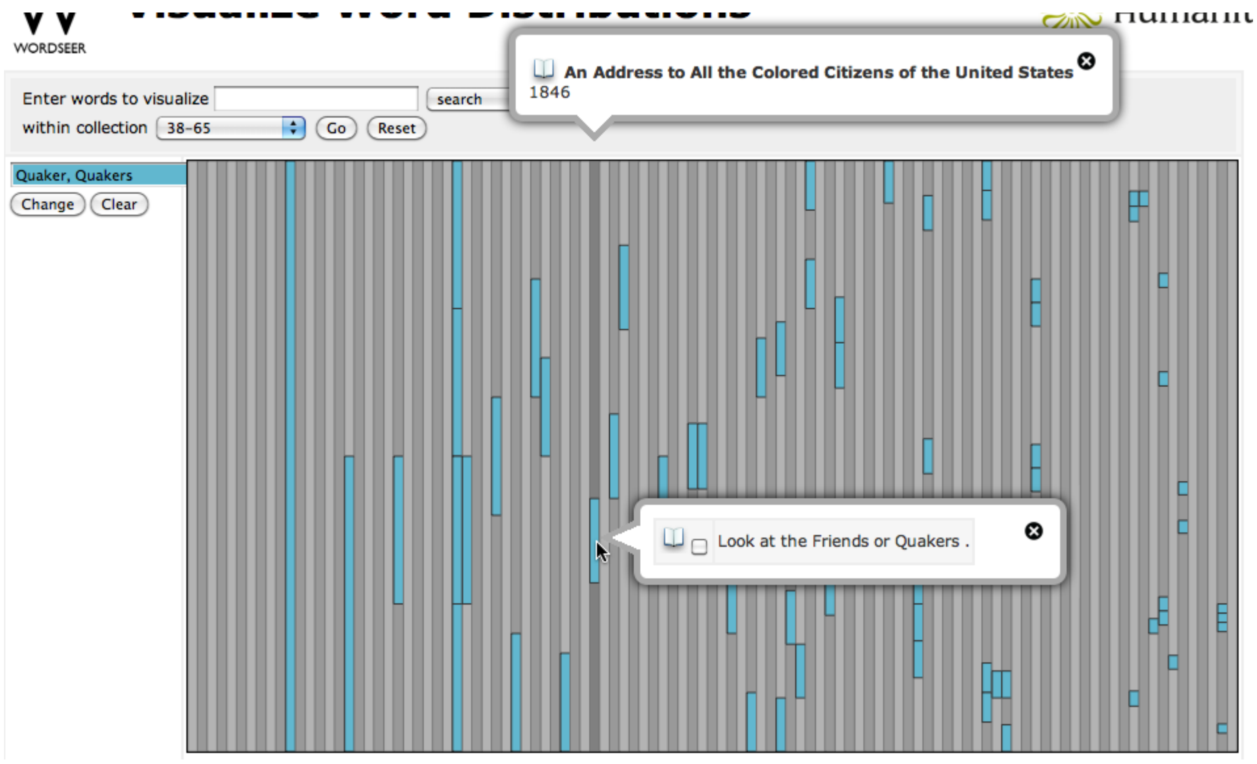


Figure 3.7: Escape from slavery.



An Address to All the Colored Citizens of the United States
1846

Look at the Friends or Quakers .

Figure 3.8: Quakers: a less prevalent theme.

escapes. The remaining referred to a periodical called "North Star". The accounts of escape, however, are far from being representative of the collection as a whole. Instead they are clustered around a few narratives that mention the north star multiple times. In particular, the title "Uncle Tom's Companions: Or, Facts Stranger Than Fiction. A Supplement to Uncle Tom's Cabin: Being Startling Incidents in the Lives of Celebrated Fugitive Slaves" contains 20 of the 86 references to the north star in the collection. A much more conservative "stereotype" around the north star seems to be indicated – whenever the north star is mentioned, it is always in relation to escape, but the converse is not true. In these narratives, escape is not necessarily guided by the north star.

Similarly, Quakers are mentioned in some, but not all narratives (Figure 3.8). When mentioned, they are always examples of kindness, and sympathy towards the abolitionist cause. Nevertheless, this convention is far less prevalent than the cruelty, escape, or separation stereotypes. Like the north star, a more conservative re-statment seems more appropriate. It is stereotypical to portray Quakers as sympathetic to slaves' escapes, but not all escapes involve reception by Quakers.

3.2 Problems and Redesign

[1]

Though somewhat successful, the initial analysis of the slave narratives revealed the true difficulty of the problem of supporting large-scale literature study. While visualization and text mining are essential, what Professor Wagner really wanted was a tool that helped him explore documents in a collection, compare different collections of text, take notes, and organize his thoughts as his hypothesis evolved.

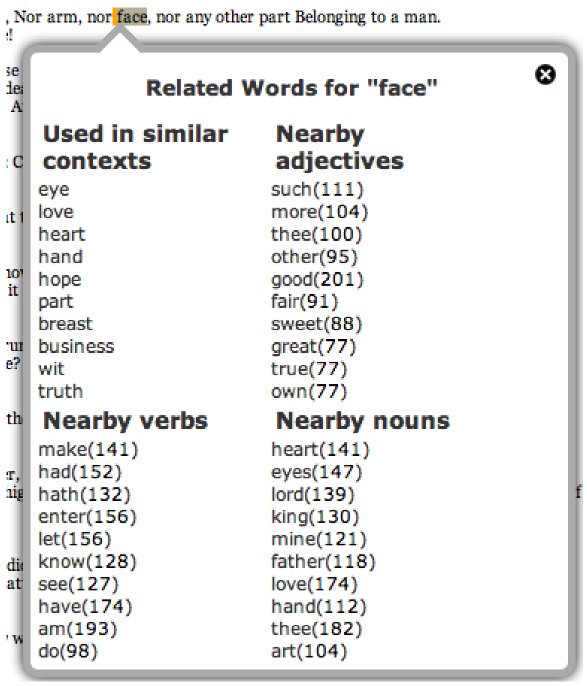
Our experiences with the slave narratives made it clear that, even from a computational standpoint, literature study was not a cut-and-dry hypothesis testing activity. Instead, it is a form of exploratory analysis: a cycle of reading, interpretation, exploration and understanding. The next version of WordSeer, version 2, attempted to support some of these activities.

3.2.1 The Vocabulary Problem

Even though grammatical search expressed certain concepts very precisely, others were more difficult to describe. These included the "white father" concept, the "barriers against slave literacy" concept, and the "description of amounts and kinds of food and clothing" stereotype. We identified two sources of this difficulty: the vocabulary problem, and the " characterization" problem.

The vocabulary problem (Furnas *et al.* 1987) is a well-known problem in search interfaces. It refers to the great variety of words with which different people can express the same concept. In our case, it manifested itself as the problem of not knowing which words were used to describe a particular stereotype in the narratives.

[1]Portions of this section were previously published in Muralidharan and Hearst (2013).



, Nor arm, nor face, nor any other part Belonging to a man.

Related Words for "face"

Used in similar contexts	Nearby adjectives
eye	such(111)
love	more(104)
heart	thee(100)
hand	other(95)
hope	good(201)
part	fair(91)
breast	sweet(88)
business	great(77)
wit	true(77)
truth	own(77)

Nearby verbs	Nearby nouns
make(141)	heart(141)
had(152)	eyes(147)
hath(132)	lord(139)
enter(156)	king(130)
let(156)	mine(121)
know(128)	father(118)
see(127)	love(174)
have(174)	hand(112)
am(193)	thee(182)
do(98)	art(104)

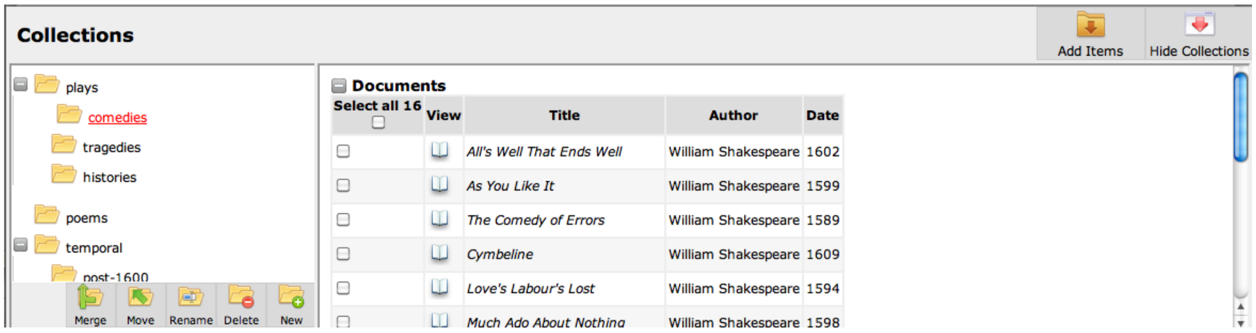
Figure 3.9: Related words for `face`

Professor Wagner would think of a few words and synonyms, but would not be able to capture the stereotype because of the multitude of words with which it was actually expressed in the text. Grammatical search helped by revealing other words that entered into the same grammatical relationships, but often, he would not be able to include these words in a search because they would also capture a lot of other events in the text that were not related of the stereotype.

To address this problem we created a related words feature. Clicking on any word while reading now brought up a small window showing related words. These words are either commonly used in similar contexts (Lin 1997), or commonly used within a 10-sentence window of each other. Figure 3.9 shows the Related Words feature, with the words related to 'face' in the plays of Shakespeare. The popup shows that other body parts are frequently mentioned, along with `love`, `fair`, and `sweet`.

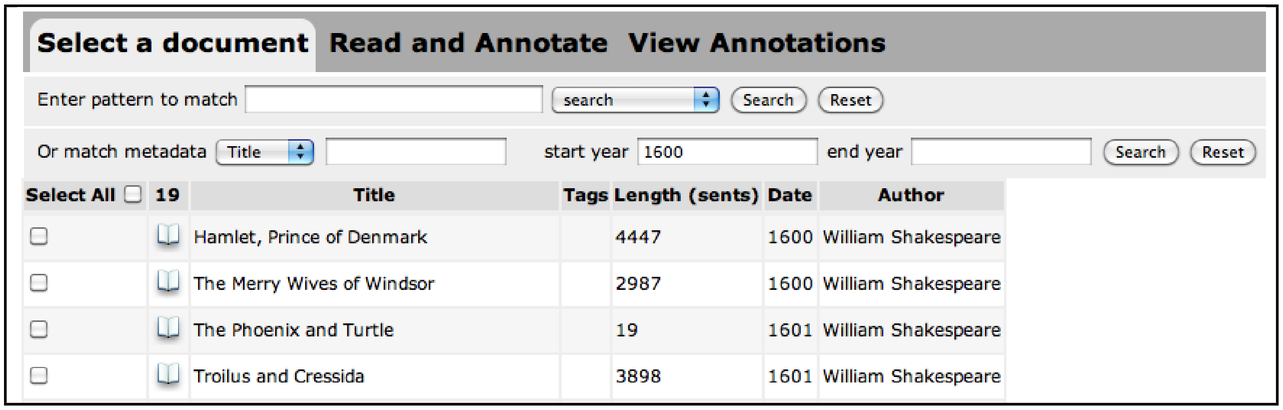
The other problem was that some types of events were hard to describe, even in terms of grammatical searches. For example, one of the stereotypical conventions was a description of a white father. However, this was usually conveyed over multiple lines of text, and interspersed with other events from the narrator's childhood. It was rarely the case that the adjective "white" was directly used to describe "father", but it was often understood that the father was white because, for example, he was also the slave owner.

We investigated a possible solution to this problem: using relevance feedback. This is a technology that uses relevance judgements made by users on their search results to improve the results of the next round. Chapter 7.1 describes this investigation.



Select all 16 ☐	View	Title	Author	Date
☐	📖	*All's Well That Ends Well*	William Shakespeare	1602
☐	📖	*As You Like It*	William Shakespeare	1599
☐	📖	*The Comedy of Errors*	William Shakespeare	1589
☐	📖	*Cymbeline*	William Shakespeare	1609
☐	📖	*Love's Labour's Lost*	William Shakespeare	1594
☐	📖	*Much Ado About Nothing*	William Shakespeare	1598

Figure 3.10: The collections bay showing a number of collections containing documents. Collections can also contain words, sentences, and "snippets" of text.



Select All ☐	19	Title	Tags	Length (sents)	Date	Author
☐	📖	Hamlet, Prince of Denmark		4447	1600	William Shakespeare
☐	📖	The Merry Wives of Windsor		2987	1600	William Shakespeare
☐	📖	The Phoenix and Turtle		19	1601	William Shakespeare
☐	📖	Troilus and Cressida		3898	1601	William Shakespeare

Figure 3.11: The document listing, used to select a document to read, or to add to a collection. Documents can be filtered using a grammatical or keyword search query on the text, by search on title or author, or by date. For easy scanning, the list of results is sortable by title, date, document length, and author.

3.2.2 Collection and Comparison

The second problem with our analysis was that the collection of slave narratives was not homogenous. Some narratives had been written with more white intervention than others, some were poetry, letters, or newspaper articles and some were biographies, and not autobiographies. Each of these categories was interesting by itself and Professor Wagner felt that a more meaningful analysis could be performed if there was a way to separate and compare them.

We therefore implemented a Collections feature. This allowed comparative analysis (Figure 3.10) by allowing users to collect and organize documents into hierarchical sub-collections. Collections could be moved, renamed, merged, and modified. The document listing (Figure 3.11) was used to add the appropriate plays, which were sortable and filterable by date, title, full-text search, grammatical search, and length. Collections were created using the "collections" bay, a collapsible window at the bottom of the screen.

For example, WordSeer 2's collections feature could be used to divide Shakespeare's plays into *comedies*, *tragedies*, and *histories* – the three most commonly-accepted categorizations of Shakespeare's plays. Then the results of grammatical searches or visualizations of word prevalence could be compared across these categories. Temporal differences can be investigated by creating the *pre-1600*, and *post-1600* categories.

3.3 Case Study 2: Hamlet in the Humanities Lab

After the redesign following Case Study 1, we were able to use WordSeer 2 to replicate part of Froehlich (2012)'s masters thesis on gendered language in Shakespeare (Muralidharan and Hearst 2013). Still, we were unsure of its ability to support this type of analysis in an authentic setting. To evaluate the new design's performance on basic analysis in the humanities, we collaborated with Professor Michael Ullyot at the University of Calgary, who used it as part of his undergraduate Shakespeare class.

In Spring 2012, 24 undergraduate students in English 203, 'Hamlet in the Humanities Lab' (Ullyot 2012) spent three weeks becoming familiar with WordSeer along with four other computational text analysis tools. Then, during the rest of the semester, they used the tools to analyze a topic of their choice within an act of Hamlet. This was a remote case study – we had no contact or prior communication with the students. They learned how to use WordSeer based on instructional videos posted online, and the other tools based on manuals. The students recorded their experiences through weekly posts on the class blog (`http://engl203.ucalgaryblogs.ca/`).

The four other tools were:

1. Voyant Tools `voyant-tools.org` (Rockwell *et al.* 2010). A 'web-based reading and analysis environment' developed at McMaster university. Voyant allows users to input a selection of text and view multiple different visualizations of the content of the text.

2. Tapor Tools (now subsumed by Voyant Tools) (Rockwell 2003b). A similar web-based text analysis tool kit, an earlier version of Voyant.

3. WordHoard `wordhoard.northwestern.edu` (Mueller 2008). An application for analyzing the Internet Shakespeare Editions' XML files by calculating word counts and lists of most frequent words.

4. MONK `monkproject.org` (Llora *et al.* 2008) A text classification toolkit. The web interface allows users to upload texts, which they can divide into test and training sets, and run various classification algorithms.

During the course, students hypothesized answers to quantifiable questions – How do different characters express doubt? How much of the action relies on Horatio's presence? – before testing their hypotheses with visualizations and quantitative evidence. The students investigated questions in groups.

3.3.1 Problems

Despite our earlier efforts to support sensemaking, the students' blog posts revealed several analysis activities that were not supported well by WordSeer. These were:

1. Narrowing down the text by metadata, such as (in Shakespeare) a particular speaker, act, or scene.

2. Getting a sense of a collection's contents based on lists of frequent words.

3. Collecting and investigating a group of words or phrases together.

4. Comparing two or more visualizations side-by-side, or referring to multiple tools simultaneously

5. Getting ideas for a new search or analysis based on the results of a previous one, and following up on those ideas.

Some of these were completely absent: there was no way to get an overview of the contents of a text based on the most frequent words; and collecting, grouping, and comparing passages of text based on metadata properties was also impossible.

The other problems were cumbersome user interface flows for important or frequently-performed activities. Searching or visualizing a group of words required long search queries. Students often wanted to perform side by side comparisons, but had to open multiple browser windows and re-navigate to the website in order to do this. They frequently wanted to follow up on new trains of thought, but this meant interrupting what they were doing, and losing state because new searches replaced the contents of the current page.

The most crippling flaw was the inability to isolate specific passages of interest based on metadata. The entire first half of the class was based on analyzing a particular act and scene within Hamlet.

> "The main issue that our group has noted is we cannot seem to isolate a single scene within Hamlet, and therefore have had problems when comparing 3.4 to the rest of the play." – madelynbrakke 3/7/12

As for individual characters, all 24 students made at least two posts referencing an analysis of particular individual's speeches (123 of the 180 posts) and 18 students made repeated references to manually isolating portions of the text relating to character developent and the speeches of individual characters (33 posts):

> "To properly distinguish between Hamlet when he was speaking and Hamlet as referenced by another character, I gave him the code name Hmlt when he appeared as a speaker and repeated the process for Gertrude (Grte), Polonius (Plns) and the Ghost (Ghst)... Using this system, not only was I able to separate speakers

> from speech, I was also able to track the frequency of each speaker though out the scene using the code names." – kathrynmcintosh 3/2/12
>
> "We decided that using each of our individual tools, we would look at character development in Hamlet, and how the characters seem to change from the first act to the last." – amybroddy 3/16/12
>
> "I start by looking at the speaking frequency of each character, to then go and list out their common words." – kira 3/26/12

After isolating a portion of text, students would often use Voyeur's word clouds or WordHoard's lists of frequent words to get a sense of the contents, making interesting discoveries along the way. 20 of the 24 students made references to using lists of words in this way, across 86 posts:

> "I looked up Act 2 with separation to 2.1 and 2.2 and looked up common words that connect to themes that are seen throughout Hamlet 2.1 and 2.2." – hayleydunmire 3/16/12
>
> "For example, after discovering that 'alas' was a commonly used word amongst Gertrude's vocabulary, I searched it through Monk, and was able to get a better grasp on the particular situations in which she used this specific word." – hannahlacoursiere 4/25/12
>
> "They are limited, yes, but the word lists provided allow me to focus on specific words and results I will look at again in another close reading of the scene, allowing my mind to be enlightened a little more than it was before." – kira 3/7/12
>
> "Laertes' use of words is similar to that of Claudius; his speech is very action related, mainly in regards to revenge: Laertes' words show that he is spurred to take vengeance, and his references to and 'sister' highlights the reason for this action" – kira 4/24/12

WordSeer, however, had no content-based overviews of any kind.

Another poorly-supported activity was investigating a group of words together. After reading, or looking at lists of frequent words, students would become interested in some of them, start to notice a theme, or want to investigate certain words further. They would then search for, visualize, or otherwise compare sentences containing these groups of words. 19 students made references to this kind of activity, across 80 posts:

> "This is my second step into the process, looking up 'soul', 'heaven' and 'devil' or any words of the like." – teresastapor 3/27/12
>
> " I took the list of most frequent words and searched on the most active words. I chose to look up words such as: death, revenge, I'll, come, stand, gone, shall, away, etc.. I omitted words like: lord, father, blood, sister, daughter, saint, king, etc.. By searching mostly verbs or words associated with actions and leaving out

> relationship describing words, I not only narrowed down my search, but also was able to get a better idea of relationships through context." – allison 3/26/12

In all of the tools, this required manually typing and re-typing in long search queries. This was frustrating, especially when the words were related:

> "I scan the text and look for words that seem to pop out or seem to be an underlying trend and then search those but the fact that I can't use related words almost defeats the purpose of that." – daynaaasen7 3/21/12

> "By slowly typing in each word in search bar that was a part of the body, my Phase Two group was able to make the theme of our presentation based on senses (eyes, hearing, and speaking/ tongue)." – CarlyFerguson 4/24/12

Like other studies of sensemaking behavior, we found that comparisons were very common. 21 of the 24 students made references to comparative analyses, across 82 posts. One fruitful type of comparison was direct, side-by-side visual comparison:

> "I was surprised particularly by the comparative frequencies of the word 'father'. In Act 1 it is mentioned 9 times by Hamlet, but in Act 5 in is only mentioned by him once. I thought this result was interesting because Hamlet's main motive throughout this act is to avenge his father, but he hardly mentions him in the moments leading up to, and immediately following Claudius' death." – stephanievandewark 3/16/12

> "I copy and pasted honesty and fair side by side to compare. It was not until I looked further into Hamlet's word choices, that I realized how often Hamlet used honest and fair." – Carly Ferguson 3/21/12

> "The first thing I noticed immediately after separating their lines, is that it looked like Laertes was saying a lot more words and had longer lines than Ophelia did." – amybroddy 3/21/12

> "Using Voyeur's Bubble Line tool I compared the words 'know', 'known', and 'knowledge'; in doing this, I found out that the word know appears a lot more often than the other two do, it also appears in conjunction with itself in two points and in conjunction with knowledge in one point, whereas it is never in conjunction with known. This leads me to believe that what is already known is not of the same importance as the desire to know things in Act II of Hamlet." – kassidybridgeman 3/21/12

However, WordSeer (and the other tools) only had limited, roundabout support for this. It was technically possible to compare two visualizations – but only by opening a separate browser window, navigating to the WordSeer Shakespeare website, and re-typing the search parameters for the second visualization. Such transitions were frustrating. Speaking of a similar situation in MONK, one student said:

> "Classification only allows you to use one workset at a time which I find very frustrating because I have to write down my findings then switch over to the other workset, do the same thing then manually compare the two together." – hayleydunmire 3/2/12

Another common occurrence was noticing something while working, and wanting to follow up on that train of thought. 15 of the 24 students made references to this type of activity across 32 posts:

> "Especially with the use of "or," contrasting words like "heaven" and "hell" are set against each other and provide scrumptious brain-food for thought.Âă In my case, I was spurred on by this specific search and borrowed many of the opposing words I found and came up with some of my own, to discover what other secrets lie within the play." – nicolericher 3/2/12
>
> "Something interesting I noticed on this map is the difference between the usage of the words "Kill" and "death" – with the first column representing "death" and the final column representing "kill". Why is there such a difference in the amount of times each word is used? Does it mean anything?" –richelleatkin 3/5/12
>
> "In the opening stages of the scene he is forceful and confidently lecturing his mother. By the scenes end he seems disoriented, and intensely trying to defend his sanity to his mother and get out as soon as possible ... This stream of thought would not have played out had it not been for my intrigue upon first seeing the distribution of the word "good" in the List Words tool as pictured below" – mattgigg 3/7/12
>
> "A frequent word which points me to a possible theme is the use of the word good... these discoveries bring up new questions for me about the scene: why is the adjective used frequently near the end of the scene? Why is Hamlet the only character to use it? What is the purpose of this repetition of 'good night." – kira 3/7/12

None of the tools explicitly supported this activity. There was no way, for example, to click on the word 'good', and get an overview of its behavior, without interrupting the current activity. In WordSeer and all the other tools, one would have to manually type in a new search, which would then replace the contents of the current page.

3.4 Implication: More Observation Needed

Although we had tried to add sensemaking functions to WordSeer 2 after Case Study 1, Case Study 2 revealed other activities that we had not even realized were important. What else were we missing? And how did the newly-revealed list of activities fit together? What were common sequences of activity that we should support in frictionless ways?

The sensemaking models we had been working with so far (Chapter 2) could not answer these questions. They were either too general, or not applicable to our domain. For example, even the most detailed model, Pirolli and Card's step-by-step of intelligence analysts' workflows (Pirolli and Card 2005), was underspecified. It spoke of 'collecting items of interest' and 'narrowing down'. However, the items of interest to intelligence analysts are documents, but in the humanities, they are anything from passages of text, to sentence fragments, to groups of words and phrases. WordSeer only allowed separating and collecting documents, but not slicing and collecting smaller pieces based on the metadata properties of individual passages, or the results of searches.

Another example was investigating a group of words or phrases together. Students in the class reported doing it frequently. While reading, they would collect words and phrases into groups representing themes, and then want to search for them together. They would try to collect the sentences matching these 'themes', and see how they intersected with metadata like Act, or Speaker.

More than anything else, Case Study 2 showed the need to understand the text analysis processes of our target audience in more detail. We needed to understand the different activities involved, how they fit together, and what other activities there might be. In order to do so, we conducted a series of interviews with humanities and social sciences scholars. These are described in the next chapter.

Chapter 4

Characterizing the Sensemaking Cycle for Humanities Scholars

4.1 Background

The aim of this chapter is to describe how humanities and social sciences scholars work with primary sources, specifically those that are text-based and consumed primarily by reading. These are usually cultural and historical documents like letters, newspapers, books, and magazines. Scholars look through these items for evidence that supports their characterizations, descriptions, or arguments about historical events, people, culture, or society (Palmer and Cragin 2008; Palmer *et al.* 2009; Blanke and Hedges 2013).

We address a gap in current descriptions of these scholars' workflows. In the past, most analyses have focused on how these scholars use reference material. Particularly, within the discipline of library of information science, humanities scholars' needs and approaches to referencing, assimilating and citing library materials have been investigated since the 1980s (Stone 1982). By contrast, there is very little work on the ways in which these scholars use primary source documents.

In the sensemaking literature, there have been studies of other types of people engaged information-intensive tasks. The general situation of making sense of large numbers of documents was studied by Cowley *et al.* (2005) and Russell *et al.* (2006). In a more domain-specific way, Pirolli and Card (2005) studied intelligence analysts, and more recently, Hong (2013) studied online shoppers, and Peterson and Bryant (2012) studied programmers who reverse-engineered code from assembly language.

However, in Case Study 2 (Chapter 3.3), we found that there were so many domain specific differences between fields that we could not use the design recommendations given by other models. In order to gather information on text-related workflows the humanities and social sciences we interviewed twelve PhD students in the literature and history departments at UC Berkeley. Their descriptions of their own work practices around primary source texts revealed a large number of activities that make up the text analysis process. The activities fell into steps with a common purpose, and scholars seemed to cycle between these steps depending on how far along they were in their research.

In this chapter, we describe the overall cycle, and the specific activities undertaken at each step along with examples, inefficiencies and complaints that scholars described. The process we found does have overall similarities to another detailed sensemaking model previously described by Pirolli and Card (2005) for intelligence analysts. However, there are important differences: these scholars' arguments are more qualitative, leading to a close, careful reading and annotation behaviors; and collected evience is almost always a snippet of text rather than a whole document, leading to different ways of managing, retrieving, and organizing it.

From this model and the inefficiencies and complaints that scholars described, we extract design principles for computational text analysis systems, which we describe in the next chapter.

4.2 Interviews

The interview goals were the following:

1. To elicit descriptions of the different activities comprising a research project dealing with primary sources, and how they related to each other.
2. To understand where humanities scholars perceived difficulties or inefficiencies with their methods.

4.2.1 Method

The interviews were around 40 minutes long and took the form of a semi-structured, open-ended conversation about recent research activities. We followed the ACTA interview method (applied cognitive task analysis) (Militello and Hutton 1998) which is an interview procedure that results in semi-structured discussion and enumeration of important or frequent activities. The questions are designed to elicit a complete description of the steps involved in a mentally demanding activity from interviews with expert practitioners. We chose this method because it was previously used to discover the sensemaking process of intelligence analysts (Pirolli and Card 2005). That model was detailed enough to provide concrete tool-design suggestions, which was our goal as well.

We engaged interviewees in conversation by starting with questions such as, "Tell me about your research" and, "What have you been working on this month?" The interviewees were chosen to have active, well-developed research agendas, so they explained what books they had been reading, the questions they had been thinking about, and what their current investigations were.

From there, following the ACTA method, the researcher asked about the following:

- Activities an observer watching them work might see, and descriptions of recent incidents of those activities.

- Roles of those activities, and how they connected with other activities as well as the overall 'big picture'

- Details of different activities they described, their reasons for doing things a specific way.

- Relationships between the activities they described to sensemaking abstractions like having a new idea, verifying a hypothesis, collecting evidence, or making an argument.

With the interviewee's permission, the session was audio-taped and (later) anonymized and transcribed.

In our interviews, the ACTA method ensured that the discussion was always about an activity that participants had named, described, and linked to other activities. As a result, analysis of the interviews was straightforward. We simply summarized the different activities they listed, and organized them into a process diagram based on the sequences and procedures that had unanimous support.

Participants

We solicited participants through mailing lists, flyers, and word of mouth. They were offered a $20 Amazon.com gift card for their participation. The interviewees were twelve PhD students from UC Berkeley, seven in the English department, and five in the History department, whose research dealt mostly with text-based primary sources .

4.3 The Humanities Text Analysis Process

4.3.1 Overview

The interviews revealed that the goal of a research project in the humanities and social sciences is usually to produce a book-length monograph or dissertation (although in some cases, chapters double as journal articles). Scholars described how they assembled their works chapter by chapter, with each chapter starting as a loose collection of vaguely-related snippets of evidence accompanied by notes, and evolving over time into a curated presentation of evidence supporting a clear line of reasoning.

The scholars collected these snippets from the documents they read and copied them out manually into a word processor. They usually roughly organized these snippets into ad-hoc groups based on source, and added bibliographic notes. Then, they created the structure of their chapter by re-organizing their collected notes and freewriting (a form of stream-of-consciousness writing popularized by Elbow (1983), widely adopted as a way to generate ideas and improve the quality of writing (Belanoff *et al.* 1991)).

Not everything they read got copied over into their notes. Annotation was an intermediate stage – an indexing of items of interest. As scholars read, they underlined, highlighted or otherwise marked passages of interest, which they later revisited and sometimes copied into their notes. The first time they read, they tried to get a general sense of style and content, and were relatively indiscriminate in the passages they annotated. At this stage, new ideas were occurring to them, and they hypothesized new ways of categorizing or characterizing what they read. They often returned to what they had already collected, and reorganized it, or added more notes. Later on, while assembling evidence for a specific argument, their reading became more focused: more like a search for text that matched some criteria, rather than an exploration. Their previous annotations now acted like an index for noteworthy passages and helped them triangulate to relevant places in the text.

To summarize, the activities that all scholars mentioned, roughly in order of occurrence are:

1. Reading
   (a) Annotating or marking the text
   (b) Ideation, making assessments, or hypothesizing new characterizations or categorizations, reorganizing notes if necessary
   (c) Searching and re-reading previously-read text
2. Writing
   (a) Copying out salient snippets from the text into notes
   (b) Making sense of collected snippets by grouping, categorizing, and freewriting, returning to the text to search for more when necessary
   (c) Crafting a written draft from previous notes and freewriting

New discoveries or new information needs trigger transitions between activities. Figure 4.1 shows the entire sequence along with the loops. It summarizes the activities and transitions that all scholars reported. We describe each of the above activites in more detail in the following sections.

4.3.2 Reading

> "I'm trying to develop a sense of what Cambodian life was like in the 1960's, so I went to bookstores, and got all these novels, and then I read them, and I also went to the archives - the Paris national archives - and got all these documents and read them." – P12

All scholars reported that they had two different modes of reading: exploratory initial reading, and focused re-reading. In the exploratory mode, they read to become familar with the text. This occurred earlier on in the research and was accompanied by annotation (underlining, marking or margin notes) on the document, but not copying-out passages or longer note taking. Three of them mentioned serendipitous discovery as a side benefit of reading in this way.

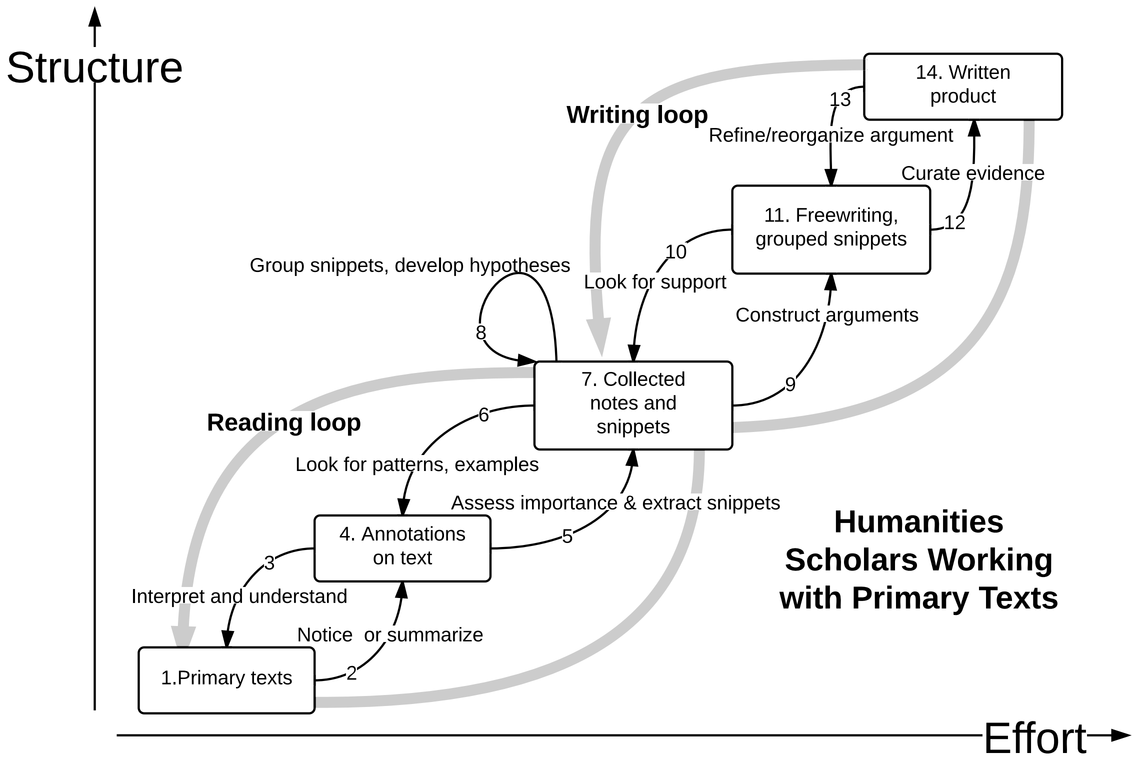

Structure
Effort
Writing loop
Reading loop
14. Written product
11. Freewriting, grouped snippets
7. Collected notes and snippets
4. Annotations on text
1.Primary texts
2 Notice or summarize
3 Interpret and understand
5 Assess importance & extract snippets
6 Look for patterns, examples
8 Group snippets, develop hypotheses
9 Construct arguments
10 Look for support
12 Curate evidence
13 Refine/reorganize argument
Humanities Scholars Working with Primary Texts

Figure 4.1: The humanities research process for primary texts.

When reading for the first time, all scholars described making generalizations, detecting and matching patterns, making comparisons, and recognizing and evaluating evidence. Additionally, eight of the twelve said that they held a number of different interests in mind while reading, and three described making connections.

The scholars we interviewed described a great variety of items of interest, which fell broadly into two categories: content and style. In other words, they paid attention to concepts mentioned in the text, and also the language and style used to represent them.

Specifically, scholars mentioned tracking individual words or phrases, as well as tracking mentions of a concept or relationship between concepts. By 'tracking', they meant everything from underlining or otherwise marking every instance they found, using a computational tool to search the document for occurrences, and in one case, making a manual tally of every time a particular word occurred.

> "I've been trying to find out if anyone has ever been positive towards the 'new woman', being approving of her. I would love to see how often that phrase shows up during the 1960s" – P12

Most scholars elaborated on this with more details. Nine mentioned searching for literary forms such as symbols, stereotype, allusion, and imagery (see Abrams and Harpham (2011) for a list). Eight mentioned paying specific attention to how the relationship between concepts was changing or evolving through the text:

> "I'm trying to understand what the trajectory was for ways of thinking about what art can do, what literature can do, and what eating can do, from a theological perspective." – P2

Seven mentioned paying attention to the different ways in which a concept was discussed.

There were other related items that were mentioned as being of interest:

1. Examples of a concept being discussed in a particular way (4/12)
2. How a concept is perceived (5/12)
3. How common a concept is (3/12)
4. A distinctive style of writing (2/12)
5. Syntactic or grammatical patterns and regularities (rhymes, lists, particular sentence constructions, etc.) (2/12)
6. The presence of a cultural influence (2/12)

The second mode that all scholars described was focused re-reading. This always occured later in the scholarly process, after the scholar had come up with an argument, and was looking for specific pieces of evidence. We describe this behavior later on in this chapter, after the steps that lead to it.

4.3.3 Annotating

As an accompaniment to their mental activities during exploratory reading, all scholars annotated portions of text that interested them, even if they could not pin down their immediate significance. All twelve described annotation as a central activity and made repeated references to it. Annotations represented the first layer of structured knowledge abstracted from raw text, without which further analysis could proceed. They reported that these kinds of annotations served as an index or a set of sign-posts when they later revisited the text in a more focused way

> "What I'm doing when I create the annotations is creating a record for myself, an index of the pdf." – P11, (P10 used almost the same words)
>
> "I'll just flip through to see where I've underlined." – P9

The most common types of annotations were margin notes (done by all scholars), and underlining (done by eight of the twelve). The remaining types of annotations were:

1. Flags (or other markers stuck onto pages, or placed between them) (4/12)
2. Arrows between text, or between annotations and text (2/12)
3. Written symbols (1/12)
4. Lines alongside paragraphs (2/12)
5. Circles around text (1/12)

Five of the twelve reported adding observations about the text in the margins, or making short summaries as margin notes. These findings are consistent with previous studies of annotation while 'work'-reading (Adler *et al.* 1998).

Interestingly, five of the scholars described an additional mode of annotation that served a more immediate and introspective purpose. This was to help themselves understand the text better in the moment, or focus their attention on the text while reading.

4.3.4 Collecting Excerpts

All scholars reported copying out quotes or excerpts of material into their note-taking tools. Like their reading habits, their evidence-collection activities seemed to evolve from exploratory to focused. In the exploratory phase, they all described their notes as a more detailed layer of annotation – the notes were organized according to their position within source documents, and always accompanied by bibliographic information or citation information.

> "I try to have the entire MLA citation, so it helps me when I'm writing." – P5

They said they copied out quotes because they gave them an idea, seemed important in some way, or seemed interesting. Four described doing so if they simply thought they might need the quotes later, and four said it helped them understand and remember the material. Nine of the twelve scholars reported adding notes about ideas they got from the quote, and six reported that they added notes about why they found the quote relevant.

> "I will usually make a note for every page of my PDF documents so that I have a record of what is in this file. That's a note for me for later if I would ever need to, and I try to use keywords that would make sense to me to search for later. If I'm really interested in a document and I think it's important, like the language is really important... then I will take extensive notes, sometimes outlining the document and sometimes outright typing up the language that's based on that." – P11

All the scholars used Microsoft Word documents to collect excerpts. Other tools mentioned were Scrivener (an authoring tool with bibliography-management assistance), or note-taking or bibliographic tools like Evernote, EndNote or Microsoft OneNote. One scholar described a system she had devised that used blogging software. She would input individual quotes as posts, and organize them using multiple tags for the various topics and categories that applied to the quotes.

Quotes varied in length, from shorter than a sentence, to multiple paragraphs. Of these, the rarest were multi-paragraph snippets, which the foreign-language scholars had copied out in order to translate. The most common were two to four sentences in length, which all the scholars had collected.

4.3.5 Freewriting, Evidence Gathering and Re-Reading

All 12 participants reported that they used writing as a tool to help organize their thoughts. They would begin by writing an outline, or drafting an argument. Free-form writing was an important way to generate ideas. All participants described how chapters gradually took shape from short, informal fragments of text such as outlines, summaries, or bullet points.

> "The actual thinking happens when I'm in my document, typing. It might be just typing out bullet-point-type things to get the ideas out, or sometimes I'll write myself an email. That is less risky to me than actually putting it in my Word document. I'm just writing out some ideas, and then it is somehow a lot easier, because it's for yourself..." –P4

> " I did all that research and collected all of these Word documents that I have, that are saved by file number or whatever. And then I stepped away from that and I began outlining a chapter. So I try to get a little away from the details of the research to say, 'What does this chapter need to do? What will the sections be?' And I start free writing." – P11

As part of this process, they would review and organize the excerpts they had collected into an argument. They would group and re-group the items they had collected in various ways according to conceptual similarity, or by the way language was used.

> " I often will organize [quotes] around a heading: a theme, or a subsection of a chapter. The quotes relating to that will be under that heading." - P4
>
> "So if I'm writing about waste, then I go to the blog where I've tagged every instance that I've found... Then I can go through and refresh my memory about what I found and figure out what makes sense. Do I want to make it historical? Do I want to start at the beginning and then move down?" – P2

Focused evidence-gathering tended to occur at this point. The sources had already been read (or skimmed) once, interesting excerpts had been collected, and an argument had been drafted. Scholars would then realize they needed to return to the text for more material to support the argument that was emerging.

> "In writing about [the passages I'd collected], I would find where the gaps of my understanding are, which would require me to then go through the other parts of the text that I've previously not paid attention to and fill it in." – P9

The scholars then collected quotes that provided evidence for specific parts of their argument. Four described collecting a quote if it summarized the author's point or argument well, and two collected quotes that they wanted to translate into a different language. They located these passages based on their own memory and their previous annotations, or if they happened to have kept electronic notes, by keyword searching.

> "So that's some of the searching I do... I know what I'm looking for; I just need where this guy lived." – P8
>
> "For me, I want to say that, 'People were intrigued by the quadroons'. I have a million primary sources and I'm taking little bits of each of them." – P6

However, if their notes did not contain the piece of evidence they were looking for, they would have to return to the text and re-read it in a more focused way. They would search over the text (either manually or with keyword search) to find all occurrences of an idea, or locate a specific passage. When searching in this way, all scholars reported having their notes open and copying passages over into them, as well as writing notes about the found passages.

> "Of course, sometimes things just don't make it in the Word file, and then I'm frantically reading and writing at the same time." – P3

4.3.6 Writing

As their writing on an idea became more and more refined, all scholars mentioned moving it over to a 'chapter'. This was a separate document (or separate area of the document), containing the 'clean' dissertation draft so far. Scholars would continue to refine this draft, but at this stage, they would seek feedback from others. Occasionally, they would need to go back and add evidence, but these were relatively small pieces.

4.4 Correspondence With Other Sensemaking Models

We now compare the humanities sensemaking process described in this chapter (Figure 4.1) to Pirolli and Card's model, and outline the differences.

Overall, the reading process (Step 1 in Figure 4.1) corresponds to Step 1 in Pirolli and Card's sensemaking process (Figure 2.4). One main difference is that intelligence analysts draw upon many different formats of data from an unknown and variable-size collection. By contrast, the complete set of source documents that a humanities scholar uses while working with primary texts is well-determined in advance. Because of their reliance of manual annotation, the scholars usually know roughly what kinds of documents are in their collection, even if they have not read every single one.

In comparison to Pirolli and Card's model, annotations correspond with the "Shoebox". Humanities scholars appear not to have a separate holding area for the contents of their annotations – they prefer to make them directly on their texts.

The humanities sensemaking process (Figure 4.1) diverges slightly from Pirolli and Card's description (Figure 2.4) at this step. Instead of "7. Collected notes and snippets", their model has two holding areas, the "7. Evidence file" for collected items, and the "10. Schema" for organizing collected items into an argument. In our investigations, we found no organized split. There was one place – the notes – in which evidence was collected. Reorganization and regrouping did occur (arrow 8 in our process), but it happened directly in these notes and not between two different holding areas (arrows 8 and 10 in their process). Further, scholars reported using this single place – their notes – in the same way as Pirolli and Card's "Schema" – as a reference while formulating ideas into an argument.

Re-reading corresponds to arrows 6 and 3 in both the humanities and the intelligence-analysis sensemaking cycles. Those steps represent the cyclical nature of the sensemaking process. As understanding evolves, new information needs emerge, which require going back to source documents and previously collected information.

Finally, the two steps in the writing loop correspond with the last two steps (13 and 16) in Pirolli and Card's sensemaking process, except that the final product is a longer written argument, and not a presentation or report.

We now characterize the difficulties scholars experienced while analyzing text, and extract design implications for text analysis interfaces.

4.5 Pain Points

4.5.1 Reading

The scholars we interviewed reported three specific frustrations with reading, all related to the problem that they had simply no other way to engage with the contents of a text collection than to read it. This is mainly because text is more difficult to analyze, summarize, and condense that numerical or categorical data. It has both linear and hierarchical structure, its meaning is ambiguous given its representation, it has tens of thousands or hundreds of thousands of features, and frequencies of words are usually distributed via a power law. Even a small fragment of text does not stand alone, but is densely interconnected with other text through linguistic phenomena. To illustrate, consider this 13-word fragment of text from Shakespeare's "Romeo and Juliet", in which Romeo reacts to finding out that Juliet is a Capulet:

```
ROMEO:
    Is she a Capulet?
    O dear account! My life is my foe's debt.
```

Here 'dear' is used the sense of expensive – Romeo is lamenting that his love for Juliet comes at a heavy price. His life is in the hands of his enemy. There are many different associations that a reader might make. For instance:

- Each of the words in the excerpt above could be a jumping-off point to looking for other contexts in which they occur, as well as to other words that mean the same thing, or that tend to be used near them.
- Similarly, each two-, three-, or n-word phrase could be linked with other sentences in which it occurs.
- The grammatical relationships between words (such as "dear" being an adjective modifier of "account") might trigger associations to other words that enter that relation (other adjectives describing "account", or other things that are "dear").
- Literary devices, such as the exclamation "dear account", or the imagery of debt, could be connected with other occurrences of those phrases, or the different phrasings with which those concept surface in the play.

The more structure there is to text, the more kinds of associations are possible. Shakespeare's plays have metadata, such as speaker, act, and scene, so associations based on these dimensions also exist:

- A scholar might want to look at all other speeches by Romeo
- Act 1, Scene 4 could be compared with other scenes in that act. All of the speakers in that scene could be isolated and compared.

To a scholar trying to make sense of an idea, some associations may be deeply meaningful. Transitions and associations are therefore central to the text analysis process, but without computational analysis, the only way to follow them is by reading. It is therefore important for text analysis systems to support not just algorithmic and visual analysis, but the transitions: slicing, filtering and exploration that lead from analysis to analysis, visualization to visualization, and finally to insight.

The scholars we interviewed reported three specific frustrations with reading: information overload, subjectivity of perception, and the difficulty of locating occurrences of complex linguistic phenomena.

1. Information Overload

The first was information overload. Four of the twelve scholars explicitly mentioned having too much material to read, but having no other way to engage with the text. Five scholars mentioned using archives of newspaper material, books, and pamphlets that were too large for a single person to read in entirety. This meant that scholars had to skim over material, and were not able to read all of it in detail.

Design Implication. Tools should give scholars 'a sense' of the way a word or phrase is used, without their having to read the text and manually identify instances of the word. Characterizing the contexts of words should be done with visualizations such as word trees (Wattenberg and Viegas 2008) or dominant-phrase clusters (Käki and Aula 2005).

2. Subjectivity of Perception

A related difficulty with reading was that scholars' perceptions were subject to state of mind. When scholars had too much material to read in detail, and therefore relied on things 'popping out' at them, their state of mind influenced what they noticed, and therefore influenced their overall perception of the text. Three scholars mentioned returning to a portion of text and noticing something important they did not notice the first time.

Design Implication. Overviews of the salient aspects of a collection should always be visible. Terms that characterize the contents of a collection (such as frequent words, themes, or topics) should be automatically identified and displayed in order of frequency. To aid in navigation, these overviews should double as filters, which allow zooming into documents and sentences that contain that type of content. Such browsing aids would make the collection easier to navigate when the scholar was searching for evidence, because items of interest would not have to be individually located by skimming. By being comprehensive and objective, they might also alleviate scholars' dependence on their own attention spans to identify and mark interesting items.

3. Complex Literary Phenomena

It is currently very difficult for humanities scholars to find instances of linguistic phenomena that are more complex than words. In order to do so, they have to resort to reading:

> "It's so easy to search for words, but it's not so easy to search for things that are only one step up. Even something as simple as related words with different endings... I usually just have to re-read the whole book in order to find the examples that support it." – P3

> "I also looked at different literary techniques that are used. Shifts between past and present is what I was most attuned to." – P4

Design Implication. Text analysis tools should incorporate techniques that allow scholars to express more complex queries. One approach is relevance feedback, which would allow scholars to express complex interests by example, without having to specify search queries. In Chapter 7, we describe our investigation into this question. Another approach is grammatical or syntactic search, which would allow scholars to specify certain types of sentence structures. For example, one scholar in the final case studies (Chapter 6) mentioned being interested in comparative sentences, and another was interested in conccessive structures (like 'although' or 'even though' clauses).

Design Implication. During search, related ideas should be made visible by showing or auto-suggesting co-occurring terms, usage contexts, and synonyms.

Additionally, machine learning and advanced natural language processing could be applied to the problems of automatically detecting instances of certain literary forms of interest, such as tense and direct speech.

4.5.2 Annotation

The main frustration that scholars expressed with annotations were that they were ephemeral and easily lost or forgotten. Three scholars mentioned this.

> "I underline a lot, but I'm wary of the way underlining can be lost. Because I underline a book and then close it and then put it on the shelf and never see it again." - P3

Also, two scholars mentioned that because they were made directly on the pages or within individual PDF files, they were difficult to skim over and retrieve.

The biggest objection, however, was to the necessity of making annotations in the first place. Seven of the twelve scholars mentioned that unless they had already annotated a text by marking items of interest, they could not proceed with analysis. And when a new item of interest came up, they would have to go through the text and annotate all the places in which the new interest came up.

Design Implication. Text analysis interfaces should support making annotations while reading. To make them less ephemeral, these electronic annotations should be indexed by all available contextual cues: time, document, location within the document, the contents of the annotation, and the annotated text. This will make them easier to remember and retrieve later (Teevan *et al.* 2004). Second, their content can implicitly support later analysis. For example, Golovchinsky *et al.* (1999) exploited the fact that annotations implicitly indicate relevant passages to significantly improve search result quality over explicit relevance feedback.

4.5.3 Note Taking

Copying out and working with snippets are central activities. All scholars reported using Microsoft Word documents to collect their notes, but eight complained that collecting many items in one file became too large and unwieldy over time. Four pointed out that if they tried to separate their collected notes into multiple files, their collections began to feel too fragmented. Finally, three scholars mentioned having difficulty searching over their own notes because they did not remember the terms they had used.

Design Implication. Text analysis tools must support collecting of variable-length snippets, that are linked to their citation information. Tools should support adding notes to these collections, and reorganizing and grouping them in arbitrary ways.

Approaches from personal information management can be applied to the problem of making large numbers of collected snippets and notes easier to manage. Studies in other domains have shown that automatic categorization by contextual cues and salient phrases makes items in personal history easier to manage (Ringel-Morris *et al.* 2003; Teevan *et al.* 2004). Systems should also aim to keep the overhead for note creation low because note-taking is such a frequent activity (Van Kleek *et al.* 2009) .

4.5.4 Searching for Evidence

After they have formulated an argument, scholars frequently look for specific evidence to support it. When they find it, they will want to copy it out into their notes. All scholars mentioned saving search results and organizing saved results into groups as part of their work flow.

> “So I do a search for vomit, and I’m just cataloging and clipping every primary source I can find that has vomit. And I start classifying how vomit is being used.” – P2

Design Implication Text analysis interfaces should support collecting and categorizing information *simultaneously* with reading, searching, and making annotations, without interrupting it. Hearst and Degler (2013) suggest interface ideas for saving search results that could be adapted for this purpose.

Seeing Search Results in Context

While searching, four of the twelve scholars mentioned that seeing the search results in the context of the original source document would be more useful in judging relevance.

> “I could see an argument being made about people in a certain kind of relationship wanting to use lines to talk about relationships. And depending on context, what if maybe he was thinking about his lover?” – P1

Design Implication Text analysis interfaces should support viewing sentences in their original context whenever sentences appear alone in search results.

4.5.5 Searching and Revisiting Previously-Read Texts

Humanities scholars and social scientists have very specific information needs when searching over previously-read material. Therefore, it is important to support issuing precise queries over the kinds of information listed in section 4.3.2.

Design Implication Eight of the twelve scholars specifically mentioned that they would like to be able to express more precise queries in the following ways:

1. Boolean search (7/12)
2. Grammatical options: part of speech, case, tense (2/12)
3. Stemming (2/12)
4. Suggested search terms: synonyms (3/12)
5. Word sense disambiguation of search terms (1/12)

4.6 Summary

Based on observations from interviews, we have described a sensemaking cycle for humanities scholars consisting of two phases: reading and writing. These phases contain sub-steps for examining source documents, extracting various kinds of information, organizing it into an argument, and composing a finished presentation of the knowledge.

We also contributed a list of difficulties encountered each step, and identified the activities and workflows that would benefit from computational assistance. In the next chapter, we translate this knowledge into a user interface design.

Chapter 5

WordSeer

In the previous two chapters, we examined the humanities and social science text analysis process through case studies and interviews. These allowed us to create a detailed, descriptive model of the sequence of activities that occurred while these scholars were analyzing text. This chapter translates that knowledge into a real-world system.

We begin by listing the technological and user interface requirements of the system. Each requirement attempts to solve a difficulty identified in Chapter 4.5. We then describe a set of design guidelines called 'text sliding', which satisfy these requirements, and finally WordSeer, Version 3, which has been redesigned according to these principles.

The system was tested with three more case studies, described in the following chapter. This final redesign of WordSeer, version 3, has achieved the goal of supporting real-world analyses that yield meaningful results that were not easily discoverable by other means. We describe the processing pipeline, back end, and user interface, and show how the design principles were applied in each case. After these case studies, the visual design of the interface was improved by Raymon Yee, a Master's student at UC Berkeley's School of Information. This chapter shows the new design, but the case studies in the next chapter show the original visual design. The features of both are nearly identical.

To illustrate the user interface, we show how WordSeer 3 was used to analyze the 180 blog posts that the students in Case Study 2 posted (Chapter 3.3). 24 students posted on the blog weekly over the entire 12-week semester and discussed their experiences using the different computational text analysis tools to study Shakespeare's *Hamlet* as well as any interesting results they were able to find.

5.1 Requirements for Text Analysis Systems

The pain points we described in Chapter 4.5 led to a list of user interface design suggestions that might address these difficulties. We condensed these into a concrete set of user interface and interaction requirements:

1. **Getting a Sense** of a collection was difficult through reading alone.

- Requirement: Show overviews of content such as lists of the most most frequent words and phrases, overviews of how many sentences match different metadata properties

2. **Visualizing** the text in different ways, and comparing and transitioning between different views was not well supported.

    - Requirement: Incorporate different text visualizations that can all operate on an underlying subset of data, and allow users to switch between different visualizations without re-typing the query. The content and metadata overviews should also filter these visualizations.

3. **Navigating** the collection based on metadata or content was difficult and required manually finding and isolating the matching portions of text.

    - Requirement: Allow overviews of metadata and lists of frequent words and to act as filters, enabling navigation to specific types of content.

4. **Exploring** the collection from a new starting point was difficult.

    - Requirement: For any given word or category, make it easy to see related terms, synonyms, contexts, and co-occurring terms and to start new threads of inquiry based on these associations with low overhead, and without losing current state.

5. **Collecting** items of interest is time consuming and interrupts flow.

    - Requirement: Make it easy to collect documents, sentences, and text fragments into groups for analysis, without interrupting the current activity.

5.2 Design Principles For Text Analysis Systems

We introduce 'text sliding', a set of user interface design guidelines intended to meet the above design goals and make the densely interconnected contents of text easy to navigate. These are guidelines for important interactions that the user interface should support, and different kinds of information it should display. The concepts of *slices* and *views* are central to text sliding. A slice is a set of sentences, and a view is a visual representation of the data in a slice. A view consists of two components: an overview of the contents of the slice, and some kind of visual representation of the data. This can range from a simple vertical list of the sentences in the slice, to more complex linguistic processing combined with visual analytics.

Slices are associated with other slices that contain the same words, phrases, or ideas. If there is metadata accompanying the text, such as date or topic tag, there are *metadata associations* as well. *Sliding* is a way to transition between related views with one or two clicks, and without interrupting the current task. A related view is either a different view of the same slice, or a view of an associated slice.

At present, slices are defined in terms of sets of sentences. Even slices based on user-created word or phrase sets, or on collections of documents are implemented in terms of the sentences that match those words or documents. However, in principle, this could be any size unit, or several different sizes of units, such as words, phrases, and paragraphs.

The principles of text sliding are the following:

- A slice is an arbitrary set of sentences
- Slices can be assembled in the following ways:
  - Sentences that match a word or phrase
  - Sentences to which a particular metadata property applies.
  - Ad-hoc, user-assembled collections of sentences
  - Intersections and unions of all the above
- A view is a visualization of the data in a slice.
- Every view should give an overview of the text contents and metadata properties of the sentences in the slice. Contents can be overviewed by, for example, listing the most frequent or distinctive phrases, and words of different parts of speech.
- Overviews in a view should double as filters that allow the user to zoom in to the sub-slice that matches the filter.
- From any given view, the following should be viewable with no more than two clicks:
  - A different view of the same slice, or
  - A view of the following associated slices:
    - A more specific slice (created by adding a search or filter), or
    - A less specific slice (created by removing a search or filter), or
    - A new slice based on a word or phrase in the current slice (moving laterally) or
    - A new slice based on words or sentences related to those in the slice (also moving laterally).
    - The sentence in its original context within a document (in the case of search results)

The remainder of the chapter describes WordSeer 3, which was redesigned according to these principles.

5.3 System Description

5.3.1 Input

The tool is run on a collection of text documents. The input is a set of XML or HTML files in a directory, each representing a document in the collection, and the output is a web application with search, visualization, and annotation capabilities. We chose HTML because many news-related data sources in the social sciences are collected by scraping web pages, and XML because TEI (`http://tei-c.org`), an XML specification for encoding documents, is a widely-adopted digitization standard in the humanities and social sciences. Many documents of interest to literature scholars are encoded as TEI-XML files.

5.3.2 Software Architecture

The tool is implemented as a web application. On the client side, the user interface is developed in JavaScript, HTML5 and CSS3. The data is stored in a collection of MySQL tables. When the user makes a request for a particular visualization of a slice, a number of PHP scripts read data from these tables, perform computations, and pass the data on to the client application in the form of JSON responses. Appendix A has a more detailed description of the architecture.

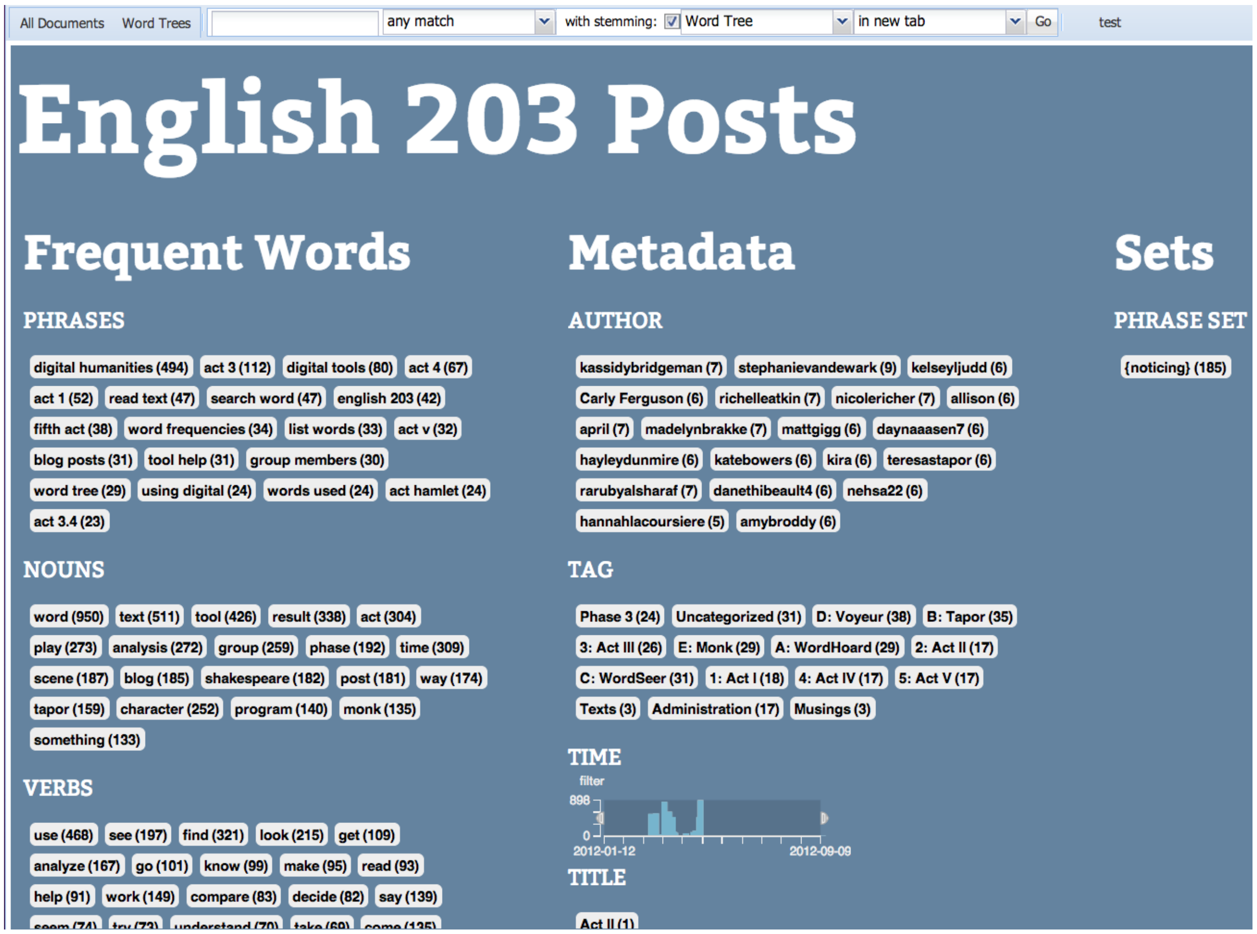
Our software architecture is guided by the information visualization software design patterns identified by Heer and Agrawala (2006) In particular, a central principle of text sliding: enabling multiple different views of the same data, was made easy to implement by following the principle of separating the software objects that represent the data being visualized from the objects that represent the components of the visualization. We also use their 'table' metaphor to represent data sets on the client side, where each data item is represented as a row in a table with pre-defined columns.

5.3.3 Initial Overview

After users log into WordSeer, the first page they see shows summary statistics of the collection (Figure 5.1). The collection being overviewed in the figure is the set of 180 blog posts from students in *Hamlet in the Humanities Lab* in Case Study 2 (Chapter 3.3).

There are three columns in the overview. The first lists the most frequent phrases, nouns, verbs, and adjectives in the collection (with counts of how often they occur). The second column shows the metadata. For text-formatted metadata, these appear as a list of bubbles, each with the value (for example, the Tag '`Phase 3`') and the number of times it occurs in the collection (24 posts). These bubbles can also act as filters. The user can click on any one of them to open a view showing a list of the sentences matching that word, phrase, or property value.

For numerical metadata, such as the date of the post, the overview is a histogram of how many documents match each value. The final column, Sets, starts out empty. When the user



English 203 Posts

Frequent Words

PHRASES

digital humanities (494) act 3 (112) digital tools (80) act 4 (67) act 1 (52) read text (47) search word (47) english 203 (42) fifth act (38) word frequencies (34) list words (33) act v (32) blog posts (31) tool help (31) group members (30) word tree (29) using digital (24) words used (24) act hamlet (24) act 3.4 (23)

NOUNS

word (950) text (511) tool (426) result (338) act (304) play (273) analysis (272) group (259) phase (192) time (309) scene (187) blog (185) shakespeare (182) post (181) way (174) tapor (159) character (252) program (140) monk (135) something (133)

VERBS

use (468) see (197) find (321) look (215) get (109) analyze (167) go (101) know (99) make (95) read (93) help (91) work (149) compare (83) decide (82) say (139) seem (74) try (73) understand (70) take (69) come (135)

Metadata

AUTHOR

kassidybridgeman (7) stephanievandewark (9) kelseyljudd (6) Carly Ferguson (6) richelleatkin (7) nicolericher (7) allison (6) april (7) madelynbrakke (7) mattgigg (6) daynaaasen7 (6) hayleydunmire (6) katebowers (6) kira (6) teresastapor (6) rarubyalsharaf (7) danethibeault4 (6) nehsa22 (6) hannahlacoursiere (5) amybroddy (6)

TAG

Phase 3 (24) Uncategorized (31) D: Voyeur (38) B: Tapor (35) 3: Act III (26) E: Monk (29) A: WordHoard (29) 2: Act II (17) C: WordSeer (31) 1: Act I (18) 4: Act IV (17) 5: Act V (17) Texts (3) Administration (17) Musings (3)

TIME

TITLE

Act II (1)

Sets

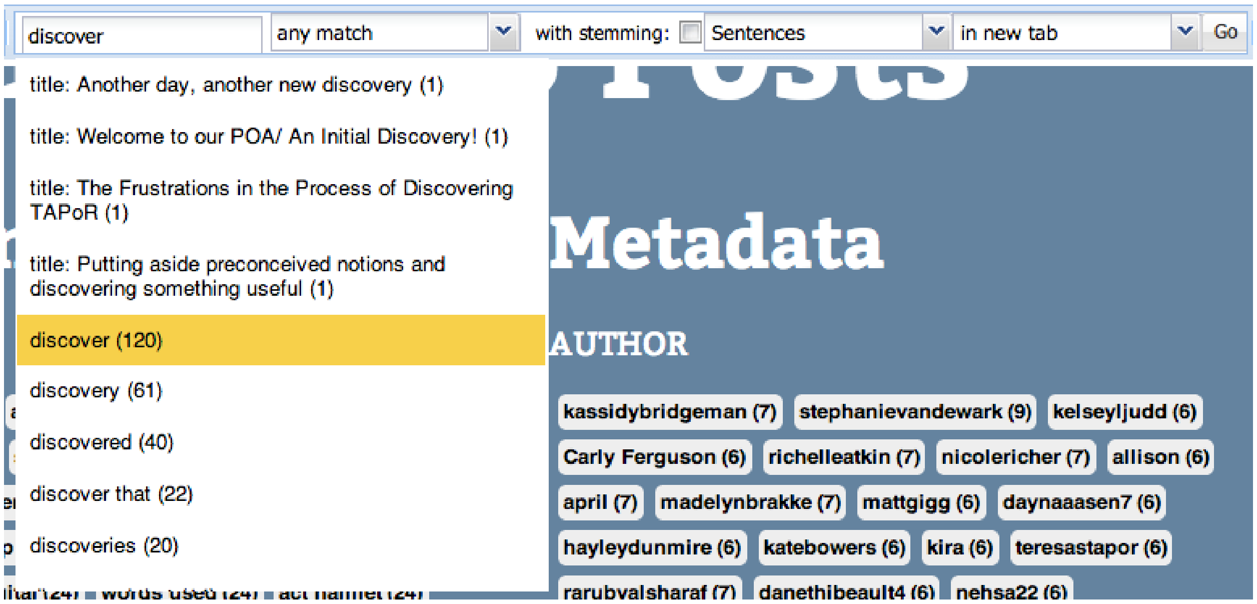
PHRASE SET

{noticing} (185)

Figure 5.1: The initial view of the collection of blog posts in Case Study 2 (Chapter 3.3)

creates custom collections of words, sentences, or documents, these appear, along with how many items are in the set.

5.3.4 Slices


discover

any match

with stemming: ☐

Sentences

in new tab

Figure 5.2: Searching for a word auto-suggests possible completions and shows how frequent they are in the collection. It also brings up metadata values that match. In this case, typing in 'discover' matches the title of four of the posts. It also matches the words `discover` (in 120 sentences), `discovery` (in 61 sentences), `discovered` (40 sentences), the phrase `discover that` (22 sentences) and `discoveries` (20 sentences). If the user had selected the "with stemming" check box, the words `discover`, `discovered`, `discoveries` and `discovery` would have been combined and their total counts would have been 241.

Users create slices by intersecting *searches* and *filters*. A search restricts the collection to just the sentences matching the query, and a filter restricts it to just sentences matching a particular metadata value.

The search bar at the top of the page (Figure 5.2) is one way to create slices. The user can perform grammatical search (as in Chapter 3.1), a keyword search, or a metadata search. Words, phrases and metadata categories are auto-suggested as the user types. As a query preview (Donn *et al.* 1996), the number of matches for each suggestion is shown.

5.3.5 Views

Views are window-like panels containing visualizations and overviews of the data in a slice (Figure 5.3). They contain the following components:

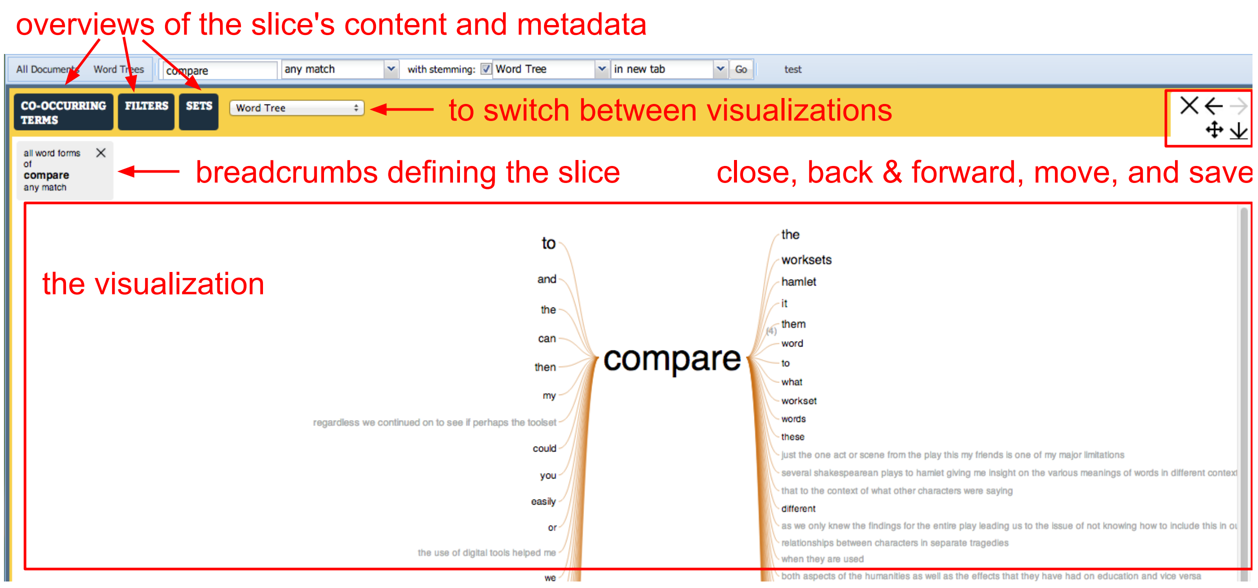

overviews of the slice's content and metadata
All Documents
Word Trees
compare
any match
with stemming: Word Tree
in new tab
Go
test
CO-OCCURRING TERMS
FILTERS
SETS
Word Tree
to switch between visualizations
all word forms of compare any match
breadcrumbs defining the slice
close, back & forward, move, and save
the visualization
to
and
the
can
then
my
regardless we continued on to see if perhaps the toolset
could
you
easily
or
the use of digital tools helped me
we
compare
the
worksets
hamlet
it
them
word
to
what
workset
words
these
just the one act or scene from the play this my friends is one of my major limitations
several shakespearean plays to hamlet giving me insight on the various meanings of words in different context
that to the context of what other characters were saying
different
as we only knew the findings for the entire play leading us to the issue of not knowing how to include this in ou
relationships between characters in separate tragedies
when they are used
both aspects of the humanities as well as the effects that they have had on education and vice versa

Figure 5.3: The components of a view. In addition to the main visualization, there are breadcrumbs that tell the user the slice currently being seen, three top menu buttons each for a different kind of overview of the slice, a drop-down menu to switch the visualization and navigation buttons to close the panel or go back and forwards in the query history.

1) A drop-down menu for switching to a different view of the same slice,
2) Breadcrumbs describing the searches and filters that define the current slice,
3) A visualization of the data in the slice.
4) A button to show the Filters overview, which gives summary statistics of how many sentences within the slice match different metadata categories,
5) A button to show the Co-occurring Terms overview, which lists frequent nouns, verbs, adjectives and multi-word phrases in the slice.
6) A button to show the Sets overview, which lists the different groups of phrases, sentences, and documents the user has created, and shows how many of those items are in the current slice.

Navigation

The user can open multiple panels in the interface to facilitate side by side comparison. Figure 5.4 shows two side-by-side views. On the left, there is a Word Tree (Wattenberg and Viegas 2008) of the word `compare`, and on the right, a Word Tree of the word `search`. The views do not have to be of the same visualization or the same slice, they are not linked. On larger screens, there can be a greater number of panels, but a laptop screen gets crowded after two are three are open (Figure 5.5 shows three panels). The tool allows the panels to be collapsed and back and forward buttons allow the user to revisit earlier views. The user's

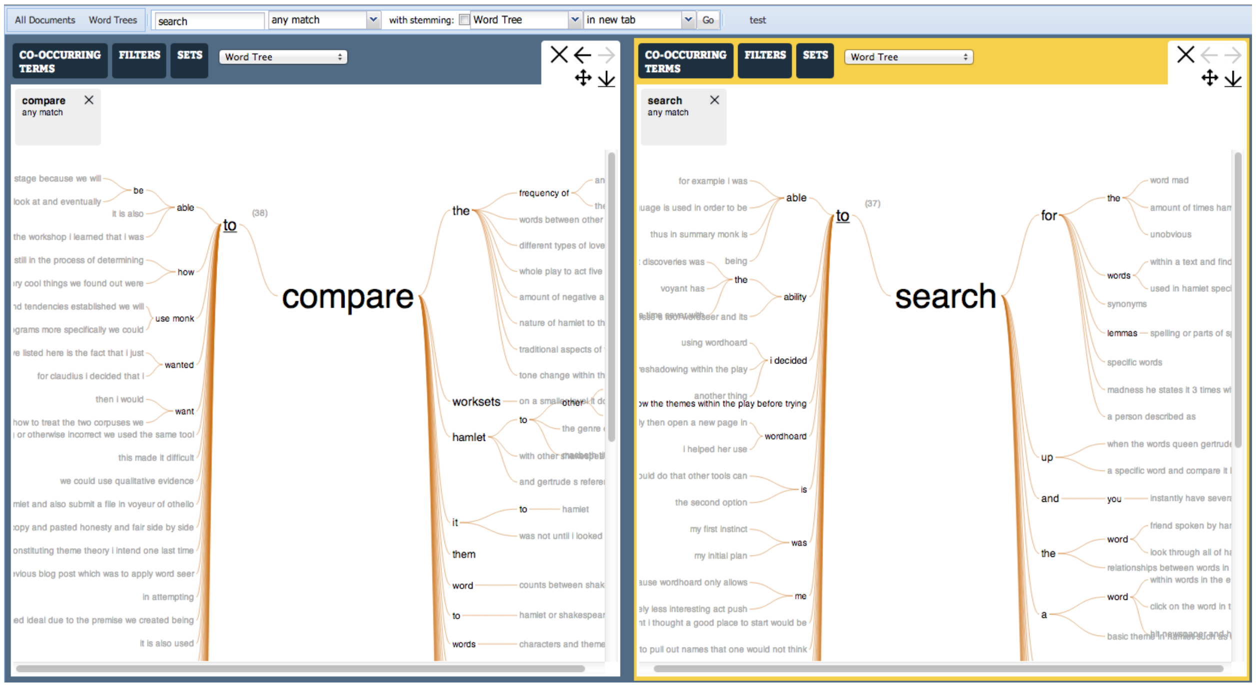
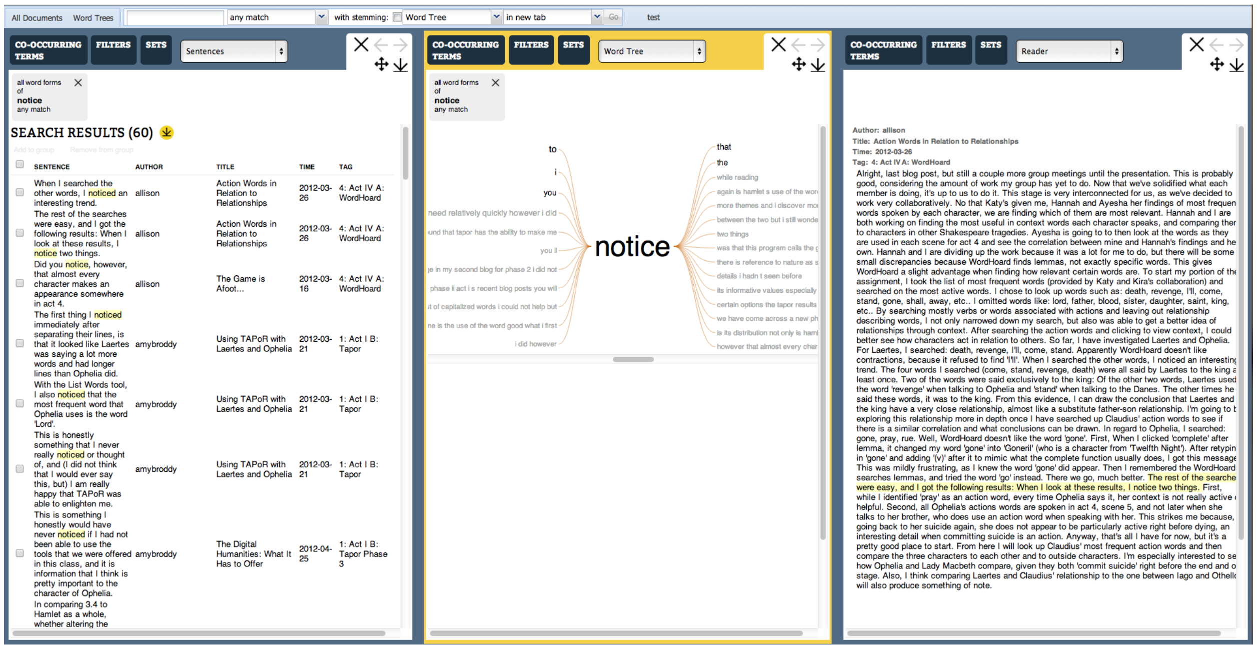


Figure 5.4: An example of two side-by-side views, each with a Word Tree of a different word. The currently active panel is indicated by a yellow background. In the figure above, the right-hand-side panel is the currently-active panel.


All Documents
Word Trees
any match
with stemming:
Word Tree
in new tab
Go
test
CO-OCCURRING TERMS
FILTERS
SETS
Sentences
all word forms of notice any match
Add to group
Remove from group
Word Tree
to
i
you
need relatively quickly however i did
und that tapor has the ability to make me
you ll
je in my second blog for phase 2 i did not
phase ii act i s recent blog posts you will
t of capitalized words i could not help but
ne is the use of the word good what i first
i did however
notice
that
the
while reading
again is hamlet s use of the wor
more themes and i discover mo
between the two but i still wonde
two things
was that this program calls the g
there is reference to nature as s
details i hadn t seen before
its informative values especially
certain options the tapor results
we have come across a new ph
is its distribution not only is haml
however that almost every char
Reader

SEARCH RESULTS (60)

	SENTENCE	AUTHOR	TITLE	TIME	TAG
☐	When I searched the other words, I noticed an interesting trend.	allison	Action Words in Relation to Relationships	2012-03-26	4: Act IV A: WordHoard
☐	The rest of the searches were easy, and I got the following results: When I look at these results, I notice two things.	allison	Action Words in Relation to Relationships	2012-03-26	4: Act IV A: WordHoard
☐	Did you notice, however, that almost every character makes an appearance somewhere in act 4.	allison	The Game is Afoot...	2012-03-16	4: Act IV A: WordHoard
☐	The first thing I noticed immediately after separating their lines, is that it looked like Laertes was saying a lot more words and had longer lines than Ophelia did.	amybroddy	Using TAPoR with Laertes and Ophelia	2012-03-21	1: Act I B: Tapor
☐	With the List Words tool, I also noticed that the most frequent word that Ophelia uses is the word 'Lord'.	amybroddy	Using TAPoR with Laertes and Ophelia	2012-03-21	1: Act I B: Tapor
☐	This is honestly something that I never really noticed or thought of, and (I did not think that I would ever say this, but) I am really happy that TAPoR was able to enlighten me.	amybroddy	Using TAPoR with Laertes and Ophelia	2012-03-21	1: Act I B: Tapor
☐	This is something I honestly would have never noticed if I had not been able to use the tools that we were offered in this class, and it is information that I think is pretty important to the character of Ophelia.	amybroddy	The Digital Humanities: What It Has to Offer	2012-04-25	1: Act I B: Tapor Phase 3
	In comparing 3.4 to Hamlet as a whole, whether altering the				

Author: allison
Title: Action Words in Relation to Relationships
Time: 2012-03-26
Tag: 4: Act IV A: WordHoard

Alright, last blog post, but still a couple more group meetings until the presentation. This is probably good, considering the amount of work my group has yet to do. Now that we've solidified what each member is doing, it's up to us to do it. This stage is very interconnected for us, as we've decided to work very collaboratively. No that Katy's given me, Hannah and Ayesha her findings of most frequen words spoken by each character, we are finding which of them are most relevant. Hannah and I are both working on finding the most useful in context words each character speaks, and comparing ther to characters in other Shakespeare tragedies. Ayesha is going to to then look at the words as they are used in each scene for act 4 and see the correlation between mine and Hannah's findings and he own. Hannah and I are dividing up the work because it was a lot for me to do, but there will be some small discrepancies because WordHoard finds lemmas, not exactly specific words. This gives WordHoard a slight advantage when finding how relevant certain words are. To start my portion of the assignment, I took the list of most frequent words (provided by Katy and Kira's collaboration) and searched on the most active words. I chose to look up words such as: death, revenge, I'll, come, stand, gone, shall, away, etc.. I omitted words like: lord, father, blood, sister, daughter, saint, king, etc.. By searching mostly verbs or words associated with actions and leaving out relationship describing words, I not only narrowed down my search, but also was able to get a better idea of relationships through context. After searching the action words and clicking to view context, I could better see how characters act in relation to others. So far, I have investigated Laertes and Ophelia. For Laertes, I searched: death, revenge, I'll, come, stand. Apparently WordHoard doesn't like contractions, because it refused to find 'I'll'. When I searched the other words, I noticed an interesting trend. The four words I searched (come, stand, revenge, death) were all said by Laertes to the king a least once. Two of the words were said exclusively to the king: Of the other two words, Laertes used the word 'revenge' when talking to Ophelia and 'stand' when talking to the Danes. The other times he said these words, it was to the king. From this evidence, I can draw the conclusion that Laertes and the king have a very close relationship, almost like a substitute father-son relationship. I'm going to b exploring this relationship more in depth once I have searched up Claudius' action words to see if there is a similar correlation and what conclusions can be drawn. In regard to Ophelia, I searched: gone, pray, rue. Well, WordHoard doesn't like the word 'gone'. First, When I clicked 'complete' after lemma, it changed my word 'gone' into 'Goneril' (who is a character from 'Twelfth Night'). After retypin in 'gone' and adding '(v)' after it to mimic what the complete function usually does, I got this message This was mildly frustrating, as I knew the word 'gone' did appear. Then I remembered the WordHoard searches lemmas, and tried the word 'go' instead. There we go, much better. The rest of the searche were easy, and I got the following results: When I look at these results, I notice two things. First, while I identified 'pray' as an action word, every time Ophelia says it, her context is not really active helpful. Second, all Ophelia's actions words are spoken in act 4, scene 5, and not later when she talks to her brother, who does use an action word when speaking with her. This strikes me because, going back to her suicide again, she does not appear to be particularly active right before dying, an interesting detail when committing suicide is an action. Anyway, that's all I have for now, but it's a pretty good place to start. From here I will look up Claudius' most frequent action words and then compare the three characters to each other and to outside characters. I'm especially interested to se how Ophelia and Lady Macbeth compare, given they both 'commit suicide' right before the end and o stage. Also, I think comparing Laertes and Claudius' relationship to the one between Iago and Othell will also produce something of note.

Figure 5.5: An example of three side-by-side views. The left panel has a search for all word forms of the word `notice`, the midde panel (the active panel with the yellow background) has a word tree for the same search, and the right-most panel has a blog post showing one of the sentences in context (highlighted in yellow)

currently-active panel is indicated with a yellow background.

To open a view in a new panel, the user can select the 'in new tab' option when issuing a query through the search bar (Figure 5.6). If the user selects the other option, 'In current tab', the contents of the currently-active panel are replaced.

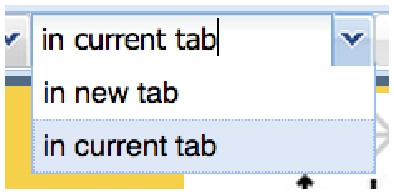


Figure 5.6: The 'In new tab' option in the search bar opens a new panel for the query. The 'In current tab' option replaces the content displayed by the currently-active panel.

Navigation buttons on the top right corner of every panel (Figure 5.7) allow the user to close the panel, to go back to the previous contents, to go forward to the next contents (if the back button had already been clicked), to re-order the panel in the view, and to save its contents to a tab-separated values (TSV) file (if the contents were displayed in a table) or as a .png image if the view was a visualization.

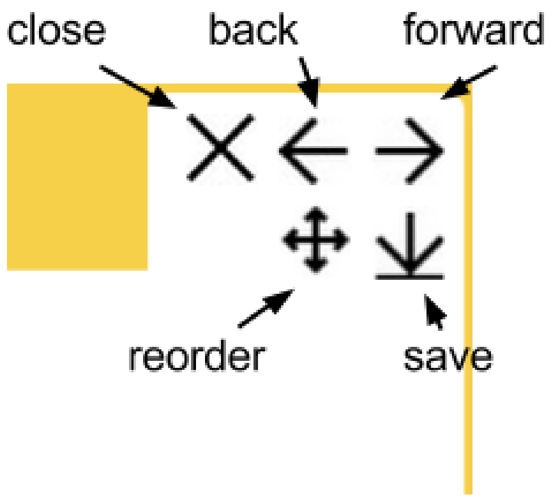


Figure 5.7: The navigation buttons at the top right of every panel allow closing, navigating back and forward in the user's search history in that panel, changing the left-to-right order of the panel, and saving the contents as either a tab-separated values (TSV) file (if the contents are shown as a table) or as a .png image if it is a visualization.

5.3.6 Overviews of a Slice

Each view shows three types of overviews: the most frequently co-occurring terms, the most frequent metadata property values, and overlaps between the slice and user-created sets. These can be viewed using the three buttons at the top of the view, and appear as overlays.

Computing Overviews

In order for WordSeer to be able to calculate the most frequent content and metadata for arbitrary sentences, each sentence is indexed by the following information:

- Each word in the sentence, and its part of speech (noun, verb, adjective, etc.),
- Each consecutive two-, three-, and four-word sequence in the sentence,
- Each metadata value belonging to the document and sub-structures to which the sentence belongs
- Each user-created category (Phrase Set, Sentence Set, or Document Set) to which the sentence belongs

At retrieval time, these associations are traversed and the most frequent terms and property values are computed. There are six content tables and four cross-reference tables that store the necessary associations. The six content tables are `UNIT`, `SENTENCE`, `WORD`, `SEQUENCE`, `METADATA` and `SET`. Each of these are responsible for assigning unique ID's to the document sub-structures, sentences, words, n-word sequences, distinct property values, and user-created groups that appear in the text. These tables are linked to each other by four cross-reference tables, which are `SENTENCE_IN_UNIT`, `WORD_IN_SENTENCE`, `SEQUENCE_IN_SENTENCE` and `SET_CONTENTS`.

The `UNIT` table is the main book-keeper for the structures within the text collection. During database construction, XML input documents are processed one by one, and the `UNIT` table assigns a unique `unit_id` to each document and to each sub-unit within the document. Sub-units are extracted from XML tags that occur within the main document's tag. The table also records each unit's `type` ('document', 'sentence', or a custom value such as 'Act' or 'Scene' for Shakespeare's plays) and `number` (the order of occurrence).

In WordSeer, all documents must contain sentences (which are extracted from plain text inside XML tags), but documents can optionally contain sub-units which can hierarchically contain other sub-units and then sentences. This hierarchy is how WordSeer represents the different acts within a play, the different scenes within an act, and the different speeches within a scene in a single Shakespeare play. Hierarchy is stored using a `parent_id` column. For every unit, the parent units that encompass it can be found by recursively traversing the `parent_id` column. Similarly, the sub-units of a particular unit can be found by recursively querying for units that have that `unit_id` as their `parent_id`. The `parent_id`s of documents are null, because they are top-level units.

The SENTENCE table stores the raw text of each sentence under the unit_id that was assigned by the UNIT table. The text is used whenever a sentence needs to be displayed in the user interface, such as in search results.

The SENTENCE_IN_UNIT table links the UNIT and SENTENCE tables. This link is not strictly necessary, because the SENTENCE table already uses the unit_ids from the UNIT table, and as previously mentioned, it is possible to compute all the units that contain a given sentence by making recursive calls to the database. However, such calls are time consuming and introduce an inconvenient overhead at query time. The SENTENCE_IN_UNIT table therefore records all the larger unit_ids that contain every sentence_id during pre-processing.

The METADATA table stores the metadata property values associated with the different units. It has the fields unit_id, property_name and property_value. For example, if a particular document with unit_id = 3 has the title 'Hamlet: Prince of Denmark', then the METADATA table row representing the information would be:

unit_id	property_name	property_value
3	'title'	'Hamlet: Prince of Denmark'

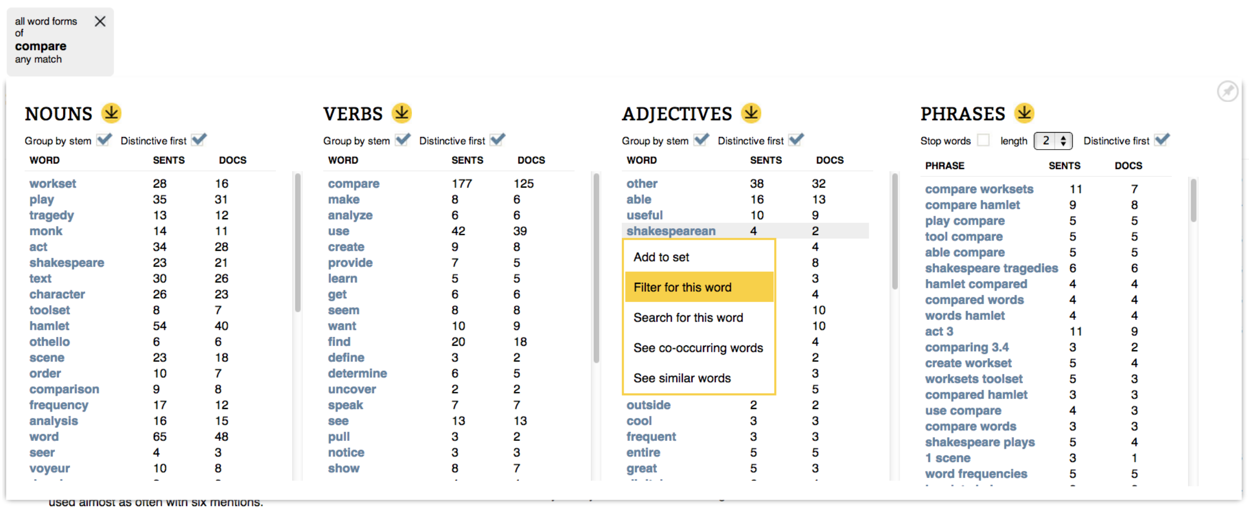
Table 5.1: An example of a table row in the METADATA table storing the title of a play.

The WORD table assigns each word a distinct word_id. It distinguishes between different parts of speech. For example, the verb sense of cost ('It cost me a dollar.') and the noun sense of cost ('What is the cost?') would have different word_ids. The WORD_IN_SENTENCE table links each word_id with all the sentence_ids in which it appears, and records the surface capitalization, spaces following and preceding, and position within the sentence.

Similarly, the SEQUENCE table assigns each 2-, 3-, and 4-word sequence a distinct sequence_id. For each phrase, variants with and without stop words, and with and without lemmatization are also computed and assigned different sequence_ids. The SEQUENCE_IN_SENTENCE table links each sequence_id with all the sentence_ids in which it appears, and records the starting position of the sequence within the sentence.

Finally, the SET table gives a name, type ('phrase set', 'sentence set' or 'document set') and unique set_id to each user-created set. Sets are hierarchical, so there is a parent_set_id column. The contents of the set are recorded in the SET_CONTENTS table, which has the columns set_id and content_id. In the case of sentence and document sets, the content_ids are unit_ids. In the case of phrase sets, they are phrase_ids.

We compute the associations for a slice, which is simply a set of sentence_ids, by traversing links between the above tables. We query for all the word_ids, sequence_ids, property_names and property_values, and set_ids that are linked to sentences in that slice. Frome those lists, we are able to create links that the user can follow to associated slices.



NOUNS

Group by stem ✓ Distinctive first ✓

WORD	SENTS	DOCS
workset	28	16
play	35	31
tragedy	13	12
monk	14	11
act	34	28
shakespeare	23	21
text	30	26
character	26	23
toolset	8	7
hamlet	54	40
othello	6	6
scene	23	18
order	10	7
comparison	9	8
frequency	17	12
analysis	16	15
word	65	48
seer	4	3
voyeur	10	8

VERBS

Group by stem ✓ Distinctive first ✓

WORD	SENTS	DOCS
compare	177	125
make	8	6
analyze	6	6
use	42	39
create	9	8
provide	7	5
learn	5	5
get	6	6
seem	8	8
want	10	9
find	20	18
define	3	2
determine	6	5
uncover	2	2
speak	7	7
see	13	13
pull	3	2
notice	3	3
show	8	7

ADJECTIVES

Group by stem ✓ Distinctive first ✓

WORD	SENTS	DOCS
other	38	32
able	16	13
useful	10	9
shakespearean	4	2
		4
		8
		3
		4
		10
		10
		4
		2
		3
		5
outside	2	2
cool	3	3
frequent	3	3
entire	5	5
great	5	3

PHRASES

Stop words ☐ length 2 Distinctive first ✓

PHRASE	SENTS	DOCS
compare worksets	11	7
compare hamlet	9	8
play compare	5	5
tool compare	5	5
able compare	5	5
shakespeare tragedies	6	6
hamlet compared	4	4
compared words	4	4
words hamlet	4	4
act 3	11	9
comparing 3.4	3	2
create workset	5	4
worksets toolset	5	3
compared hamlet	3	3
use compare	4	3
compare words	3	3
shakespeare plays	5	4
1 scene	3	1
word frequencies	5	5

used almost as often with six mentions.

Figure 5.8: The Co-Occurring Terms overview shows the terms that most frequently co-occur with a query word (In this case `compare`, as shown by the breadcrumb). Clicking on a term brings up a contextual menu. Users can add the term to a set, filter the slice by that term, or view the slice for just that term.

Content Overviews

The Co-Occurring Terms overview shows an overlay with counts of the most frequent nouns, verbs, adjectives, and phrases in the slice (Figure 5.8). Clicking on any term in these lists brings up a contextual menu. Users can add the term to a set, filter the slice by that term, or view the slice for just that term. If the user chooses to filter the slice, the overviews, breadcrumbs, and visualization all change to reflect the filtered slice.

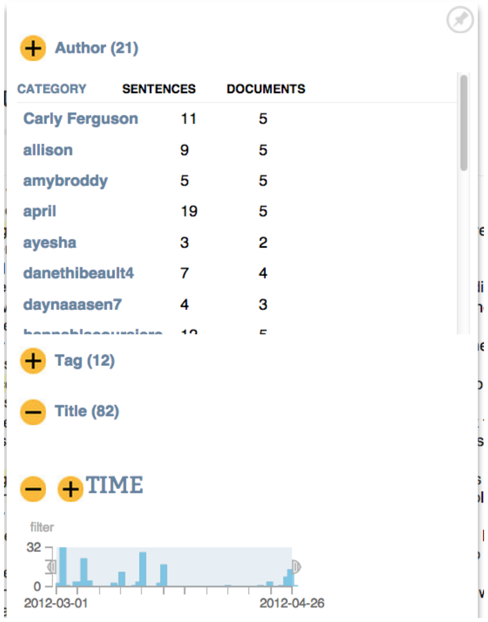
The user can sort terms either by raw frequency or 'distinctiveness', which is computed using tf-idf (Jones 1972; Salton and Buckley 1988). The words with the highest tf-idf scores are those that appear with the proportionally highest frequencies in the slice, relative to their overall frequency in the collection. For a slice S, we compute a the tf-idf for each word (or phrase) w in the slice as:

$$\text{tf-idf}(w, S) = |s \in S : w \in s| \times \log\left(\frac{|D|}{|d \in D : w \in d|}\right), \tag{5.1}$$

where D is the set of all documents in the collection, s is a sentence in the slice S, and d is a document.

Metadata Overviews

The metadata properties of the slice are summarized in the Filters pane (Figure 5.9). This overview shows the different property values present in the slice, along with the number of sentences and documents that match each value. Clicking on any of the values filters the



CATEGORY	SENTENCES	DOCUMENTS
Carly Ferguson	11	5
allison	9	5
amybroddy	5	5
april	19	5
ayesha	3	2
danethibeault4	7	4
daynaaasen7	4	3

Figure 5.9: The Filters Overview summarizes the property values matching a slice.

slice to only sentences that match that value. To reflect the new slice created by the filter, and to allow the filter's removal, a new breadcrumb showing the property value is added to the filtered view.

5.3.7 Visualizations

WordSeer includes a variety of visualizations to allow users to analyze the data in a slice in different ways. In previous versions of WordSeer, users had to reissue their search query

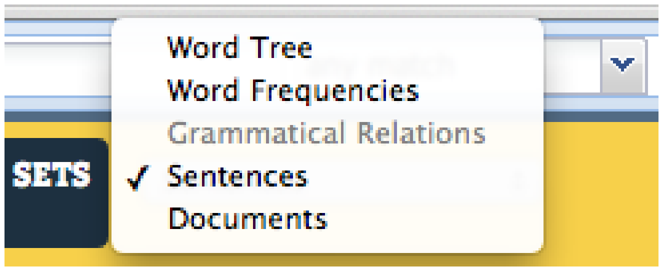


Figure 5.10: The drop-down menu for switching between visualizations of a slice.

every time they wanted to see a different visualization, even if it was of the same data. In this version, there is a drop-down menu at the top of each view which provides this function (Figure 5.10). Selecting a different view from the menu switches the visualization in the panel. Currently, the choices are:

– A list of sentences,
– A list of documents that match the sentences in the slice,
– An interactive Word Tree of the most common word in the slice, or the search term, if specified,
– Word Frequencies: charts showing distributions of the slice's sentence counts across various metadata categories,
– A document reader,
– Bar charts showing how often different words in the slice appear in grammatical relations.

WordSeer Versions 1 and 2 (Chapter 3) included an additional visualization: the newspaper-strip. These have been replaced by the Word Frequencies visualization, which is capable of conveying the same (and more) information, but is easier to interpret visually. The newspaper strip visualization had several problems. It was intended to visualize both the prevalence and position of occurrence of a search term in the documents of a collection. It used visual density to used to represent frequency. A more frequent term occurred in more places within the document, and therefore had more marks in the column representing the document. However, lengths are easier for humans to assess and compare than visual densities, which means that the information was more effectively conveyed by a bar chart. Nevertheless, the advantage of the density representation in the newspaper-strip visualization was that it also showed how the term was distributed within the document. But after WordSeer became able to support within-document metadata, such as line number, this information could also be conveyed more precisely with a chart. For example, in Shakespeare, the Word Frequencies chart for a search term shows its occurrence at different line numbers using a line graph.

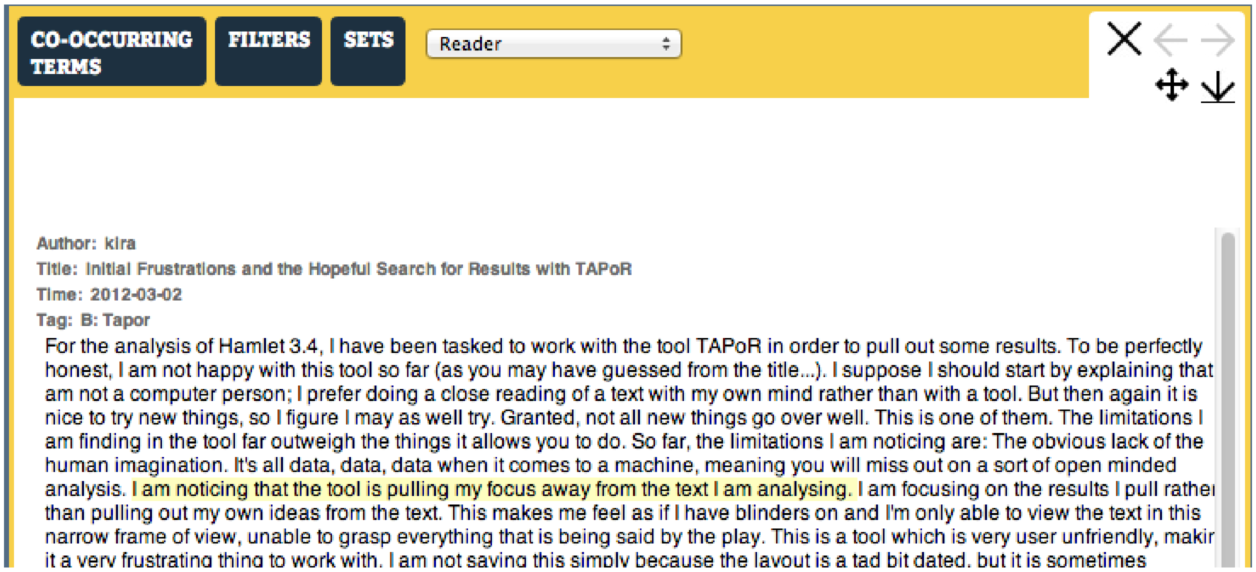
The Reader

This view (Figure 5.11) shows the contents of a document. There are two ways to open it: by clicking on a sentence elsewhere in the system and selecting 'See in Context' from the contextual menu, or opening a document from the Documents view. If the Reader was opened in order to see a sentence in context, the sentence is highlighted in yellow and the view is scrolled to that sentence.

Sentences

The Sentences view (Figure 5.12) is simply a list of sentences that matches the results of a slice. To create this view, the user can either search for a term using the search box and select the 'Sentences' option, or click on a tag in the landing page overview.

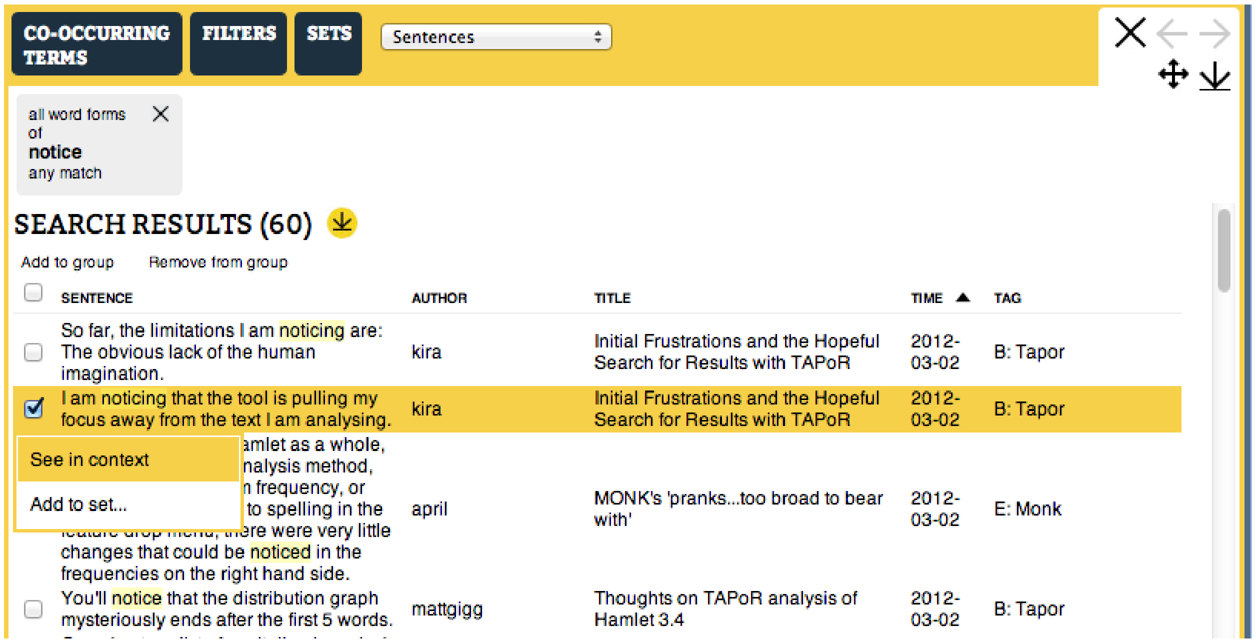
The user can sort the list by metadata values, and filter the list by any of the words, phrases, sets or metadata values in the overviews. Clicking on any of the sentences brings up a menu with options to add it to a set, or to view it in the Reader.



Author: kira
Title: Initial Frustrations and the Hopeful Search for Results with TAPoR
Time: 2012-03-02
Tag: B: Tapor

For the analysis of Hamlet 3.4, I have been tasked to work with the tool TAPoR in order to pull out some results. To be perfectly honest, I am not happy with this tool so far (as you may have guessed from the title...). I suppose I should start by explaining that am not a computer person; I prefer doing a close reading of a text with my own mind rather than with a tool. But then again it is nice to try new things, so I figure I may as well try. Granted, not all new things go over well. This is one of them. The limitations I am finding in the tool far outweigh the things it allows you to do. So far, the limitations I am noticing are: The obvious lack of the human imagination. It's all data, data, data when it comes to a machine, meaning you will miss out on a sort of open minded analysis. I am noticing that the tool is pulling my focus away from the text I am analysing. I am focusing on the results I pull rathei than pulling out my own ideas from the text. This makes me feel as if I have blinders on and I'm only able to view the text in this narrow frame of view, unable to grasp everything that is being said by the play. This is a tool which is very user unfriendly, makir it a very frustrating thing to work with. I am not saying this simply because the layout is a tad bit dated, but it is sometimes

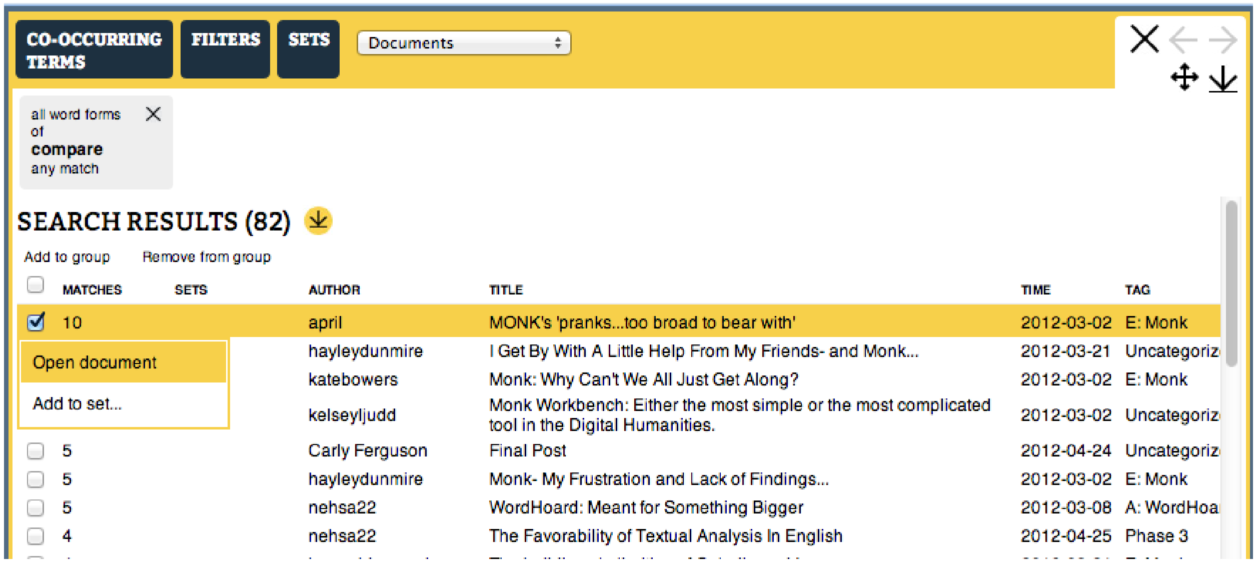
Figure 5.11: The Reader shows an entire document. If it was opened to show a sentence in context, the sentence is highlighted and the view is scrolled to that point.



SEARCH RESULTS (60)

SENTENCE	AUTHOR	TITLE	TIME	TAG
So far, the limitations I am noticing are: The obvious lack of the human imagination.	kira	Initial Frustrations and the Hopeful Search for Results with TAPoR	2012-03-02	B: Tapor
✓ I am noticing that the tool is pulling my focus away from the text I am analysing.	kira	Initial Frustrations and the Hopeful Search for Results with TAPoR	2012-03-02	B: Tapor
amlet as a whole, nalysis method, n frequency, or to spelling in the ere were very little changes that could be noticed in the frequencies on the right hand side.	april	MONK's 'pranks...too broad to bear with'	2012-03-02	E: Monk
You'll notice that the distribution graph mysteriously ends after the first 5 words.	mattgigg	Thoughts on TAPoR analysis of Hamlet 3.4	2012-03-02	B: Tapor

Figure 5.12: The Sentences view lists the sentences in the slice. The columns can be sorted, and clicking on a sentence produces a contextual menu with options to see the sentence in context or add it to a set. Selecting 'See in context' opens the Reader view and shows the document containing that sentence scrolled to that point. Selecting 'Add to Set' opens a sub-menu with options for each existing Sentence Set and a text field into which the user can input a name for a new set.



SEARCH RESULTS (82)

Add to group   Remove from group

	MATCHES	SETS	AUTHOR	TITLE	TIME	TAG
✓	10		april	MONK's 'pranks...too broad to bear with'	2012-03-02	E: Monk
			hayleydunmire	I Get By With A Little Help From My Friends- and Monk...	2012-03-21	Uncategoriz
			katebowers	Monk: Why Can't We All Just Get Along?	2012-03-02	E: Monk
			kelseyljudd	Monk Workbench: Either the most simple or the most complicated tool in the Digital Humanities.	2012-03-02	Uncategoriz
	5		Carly Ferguson	Final Post	2012-04-24	Uncategoriz
	5		hayleydunmire	Monk- My Frustration and Lack of Findings...	2012-03-02	E: Monk
	5		nehsa22	WordHoard: Meant for Something Bigger	2012-03-08	A: WordHoa
	4		nehsa22	The Favorability of Textual Analysis In English	2012-04-25	Phase 3

Figure 5.13: The Documents view lists the documents in the slice. The columns can be sorted, and clicking on a document produces a contextual menu.

Documents

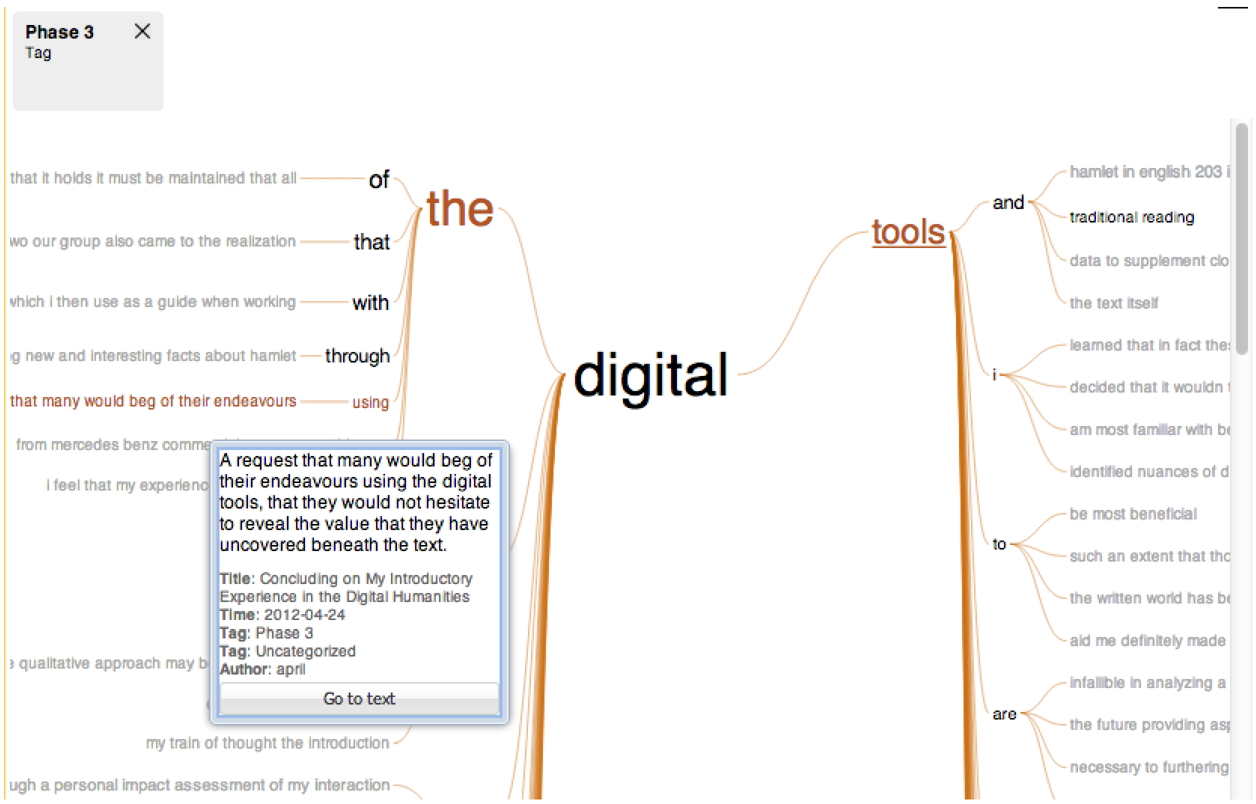
The Documents View (Figure 5.13) is another simple view that lists the documents that correspond to a slice. The user can open it up from the top bar by clicking “All Documents", or search directly using the search box. Clicking on a document in this view produces a contextual menu with options to add it to a set or open it in the Reader.

Word Trees

The Word Tree visualization (Figure 5.14) was developed by Wattenberg and Viegas (2008). It allows for quick overviews and exploration of the contexts in which a word occurs. Like a concordance, it has a central word, but it condenses the display by collapsing sentences that share a common prefix, into a tree-like structure, branching away from the central word.

We modified the original Word Tree design in two ways. First, the original showed either the left context or the right, but we show both. Second, the original Word Tree showed all the branches of the tree at once, but we found that for frequent words, the branches of the most common prefixes and suffixes filled the screen, making it necessary to scroll to get a sense of the overall usage. In our modified tree, we show a summarized version if there are more than 50 sentences. Only the first level of branches (for the words that immediately precede and follow the center word) are shown. Clicking on a branch then shows the full tree for that prefix or suffix.

Users can make a new word tree panel by choosing the ‘Word Tree’ option when issuing a query in the search bar. Clicking on a top-level branch in the word tree expands it to just that context. Then, clicking on branches further down filters the tree to just sentences with



A request that many would beg of their endeavours using the digital tools, that they would not hesitate to reveal the value that they have uncovered beneath the text.

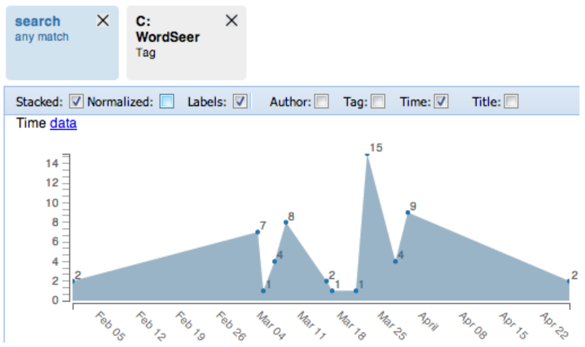
Title: Concluding on My Introductory Experience in the Digital Humanities
Time: 2012-04-24
Tag: Phase 3
Tag: Uncategorized
Author: april

Figure 5.14: A word tree for the slice produced by filtering on the tag `Phase 3`. This produces a tree of the most frequent word in that slice, which is the word `digital`. Clicking on a top-level branch shows the tree for just that prefix or suffix. Here, we have clicked on the branch for `‘digital tools’`, which shows the tree for only sentences that contain that phrase. Clicking on branches further down or on the other side filters the visualization to just sentences that contain the entire prefix or suffix, starting at that branch. For example, in the figure above, clicking on `with` on the left side would show the tree for only sentences that contained `with the digital tools`. Hovering over an individual sentence (grey text) highlights the ‘path’ through the tree taken by the sentence in red, and shows a popup with the full sentence and any associated metadata. The popup’s ‘Go to text’ button opens the Reader showing the sentence in the context of the entire document. Clicking on words in the sentence brings up Word Menu (Section 5.3.9) which allows for further exploration or starting a new search.

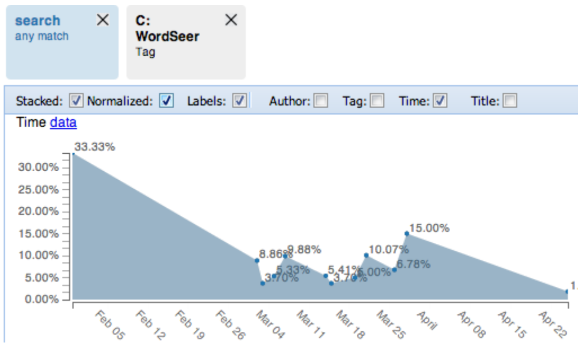
that prefix or suffix. The contexts are arranged in order from most to least frequent. The branches in grey are individual sentences. Hovering over a sentence branch shows a popup containing more information about the sentence, and the option to open up the document viewer for the play at that sentence.

If a slice is a combination of filters, and does not have a search term specified, WordSeer will use the most frequent word in the slice. Adding additional filters will change the visualization to reflect the most frequent word in the filtered set. In the blog posts, Figure 5.14 shows the result of a Word Tree with just a filter for the tag `Phase 3`. If a search term is specified, the Word Tree's center word is fixed to that term.

Word Frequencies


search any match
C: WordSeer Tag
Stacked: ✓ Normalized: ☐ Labels: ✓ Author: ☐ Tag: ☐ Time: ✓ Title: ☐

(a) The frequency graph for the number of sentences matching the term `search` in posts tagged `C: WordSeer` over time.


search any match
C: WordSeer Tag
Stacked: ✓ Normalized: ✓ Labels: ✓ Author: ☐ Tag: ☐ Time: ✓ Title: ☐

(b) The same graph with the normalization option checked. This converts the raw counts into the percentages that match the slice of the total number of sentences with that value.

Figure 5.15: The Word Frequencies view shows relationships between metadata properties.

Word Frequency charts are designed for comparing the frequencies of words across different categories. The user can make a new word frequency graph panel by selecting the 'Word Frequencies' option while issuing a query from the top bar.

Figure 5.15 illustrates the utility of this visualization with an example question, 'How are posts that mention the term `search` under the tag `C: WordSeer` spread out over time?' To answer, a user can make a new Word Frequencies graph for the term `search` and filter it to just `tag: C WordSeer`. The time graph then reveals the distribution of the sentences containing the term `search` under this tag over time.

The "Normalize" option (Figure 5.15b) converts between showing the raw number and the percentage of sentences from each property value that match the slice. For example, if there are N sentences that matched a particular value – say 'posted on `Mar 11`' – and n of them match the term `search`, the un-normalized Time graph would show the raw frequency n for that day, and the normalized version would show n/N (as a percentage). Similarly, for

Tag – the normalized graph would show the percentage of sentences that matched the term `search`, out of the total number of sentences in all posts with that Tag.

The same chart without normalization looks very different, because there are different numbers of blog posts every day, which means that the the relative frequency of the term is not proportional to its raw frequency.

Word Frequency graphs are interactive and can be filtered in order to show relationships between metadata properties (Figure 5.16). Clicking on a bar, or selecting a range (in the case of continuous values) filters the other graphs to show only matches in that range. For example, if a user wanted to perform the analysis of the usage of the term `search` by every author, they could take advantage of this filtering function. As Figure 5.16 shows, the user could create a word frequency plot for the term `search`. Then, by clicking the authors one by one, the user could see the time trend of the term for each author reflected in the 'Time' graph.

Word Frequency graphs can show multiple words. The user can either stack the charts or group them. The frequency or percentage of each term is displayed using a different colored bar. For example, Figure 5.17 shows the distribution of three words, `search`, `compare` and `analyze` across the different tags.

Grammatical Relations

The Grammatical Relations visualization (Figure 5.18) was developed specifically for grammatical search queries. It is like the Sentences view, but is augmented with bar charts of how many words match the grammatical relationship.

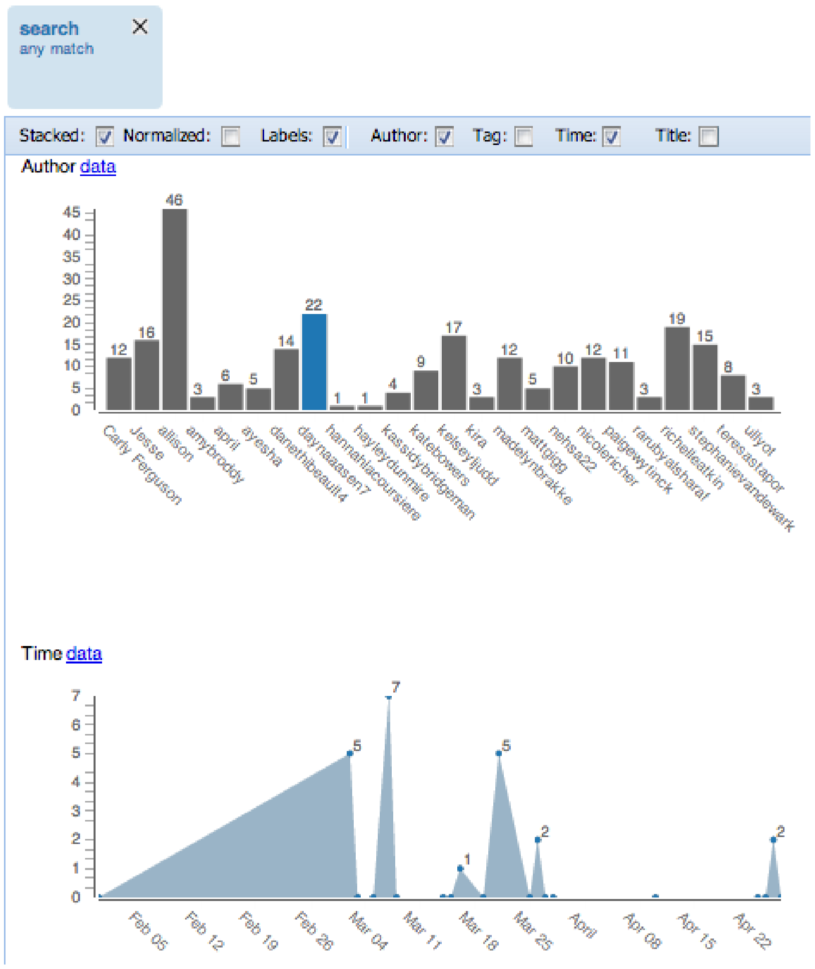
Figure 5.18 shows how this visualization can be used to investigate the act of comparison. The query is '`compare (all word forms) [done to] ______`'. The bar charts above show how often the different word forms of `compare` appears in a `done to` (verb-object) relationship, and the different words with which they appear. The right-hand-side chart shows that `words` is the most commonly compared item, at 9 times.

The list of matching sentences is shown below the chart, with the matching words on either side of the grammatical relationship highlighted in different colors. The charts are interactive, and can be used to filter each other as well as the sentences below. Clicking on a word filters the list of sentences to match that word, as shown in. The sentences can be filtered to particular words by clicking on the bars for those words. Multiple words can be used.

For example, clicking on `hamlet`, and `characters` (dark blue) filters the list of sentences in Figure 5.19 to only those in which some form `compare` appear with those two words.

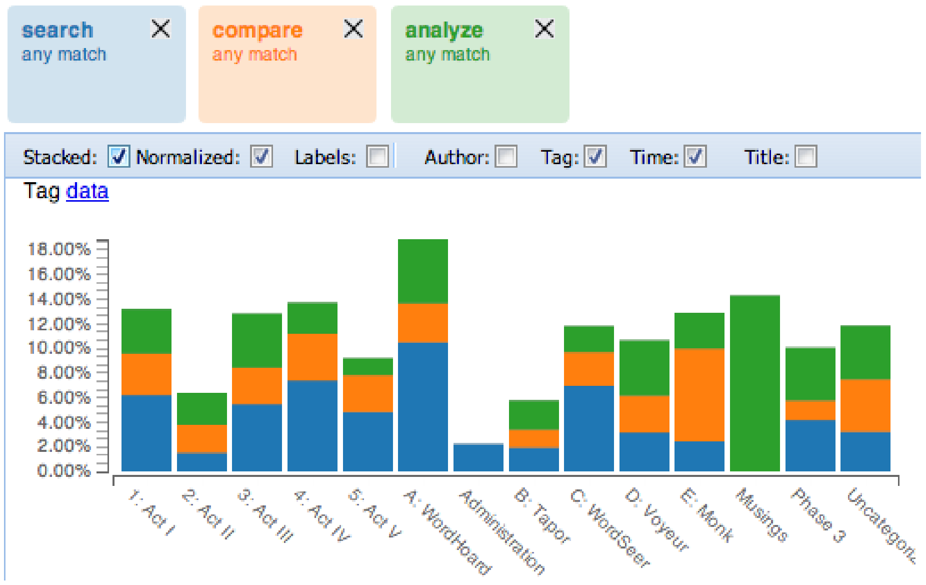
5.3.8 Sets

WordSeer allows users to construct custom slices through Phrase-, Sentence-, and Document Sets. These custom slices behave like any other slices, which means that they can be summarized in views, analyzed, filtered and searched.

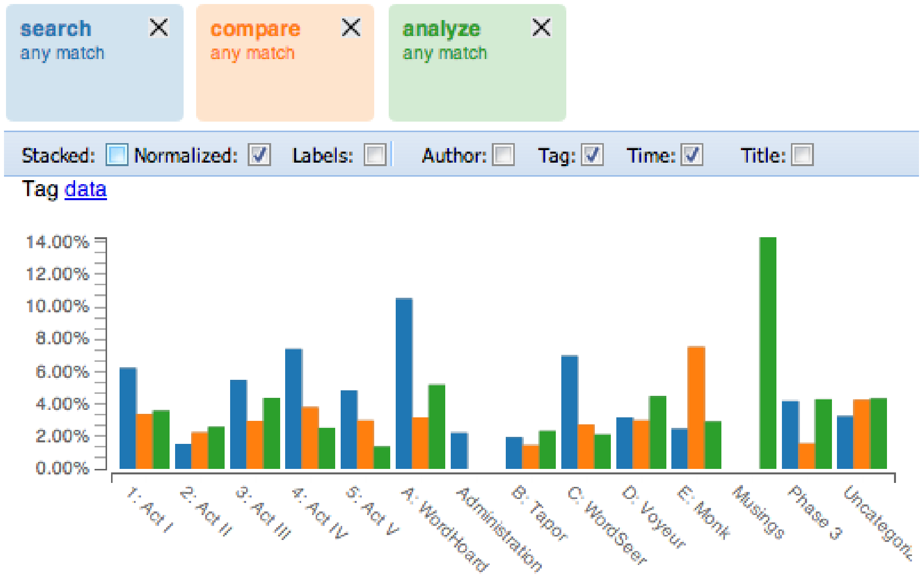

search
any match

Stacked: [✓] Normalized: [ ] Labels: [✓] Author: [✓] Tag: [ ] Time: [✓] Title: [ ]

Figure 5.16: The Word Frequencies views are interactive. Clicking on a bar in the top graph filters the bottom graph to just sentences that match that value. The bottom graph now shows the time trend for the use of the word `search` by the author `daynaasen7`.

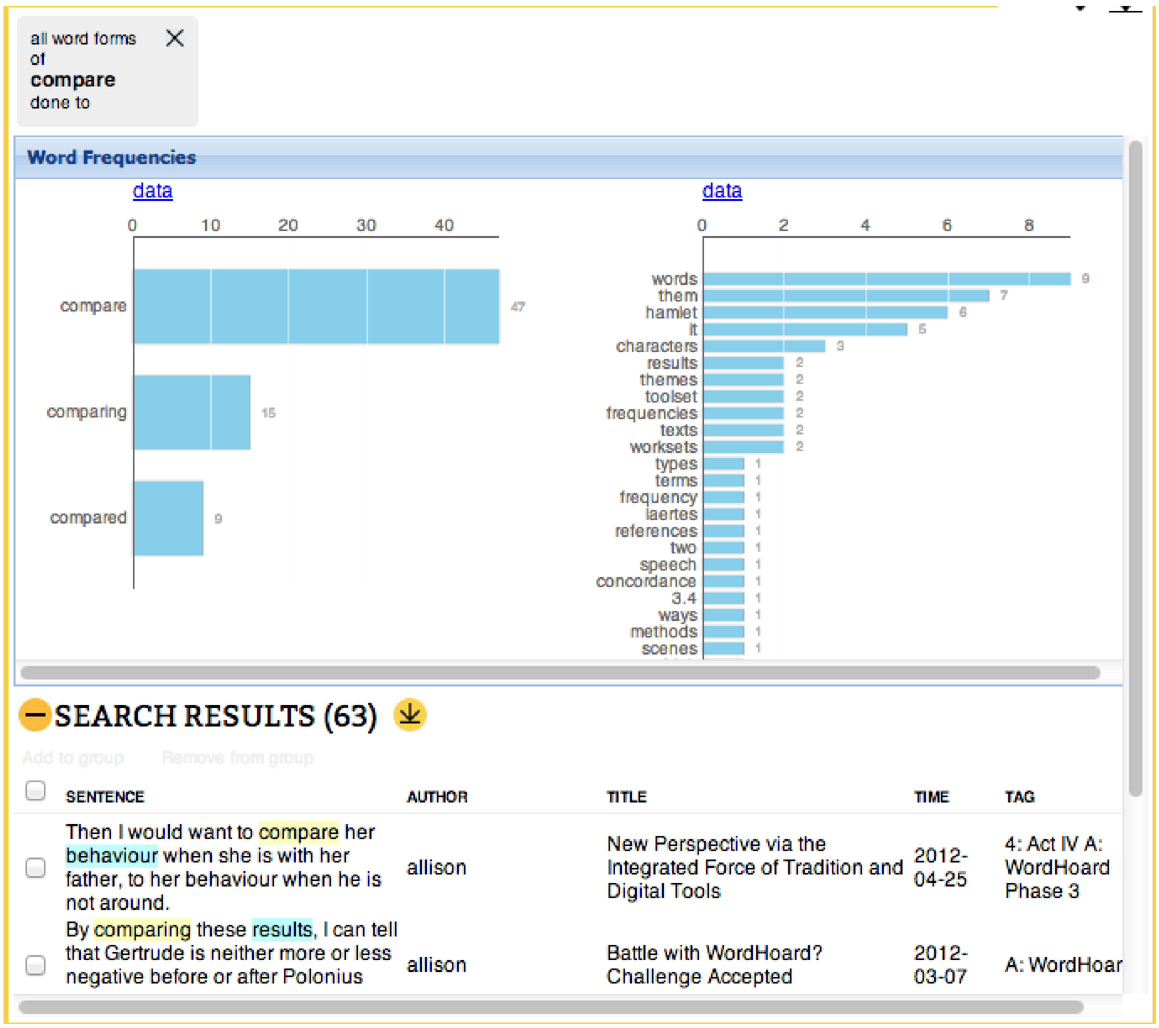


(a) The Word Frequencies graphs can visualize multiple words. Stacking the graphs, as above, shows the total percentage (or frequency in the case of non-normalized graphs). The graph above shows the distribution of the terms `search`, `compare`, and `analyze` across the different tags.



(b) The same graph with the terms un-stacked. This shows their frequencies in different-colored bars side by side.

Figure 5.17: The Word Frequencies views can visualize multiple words.

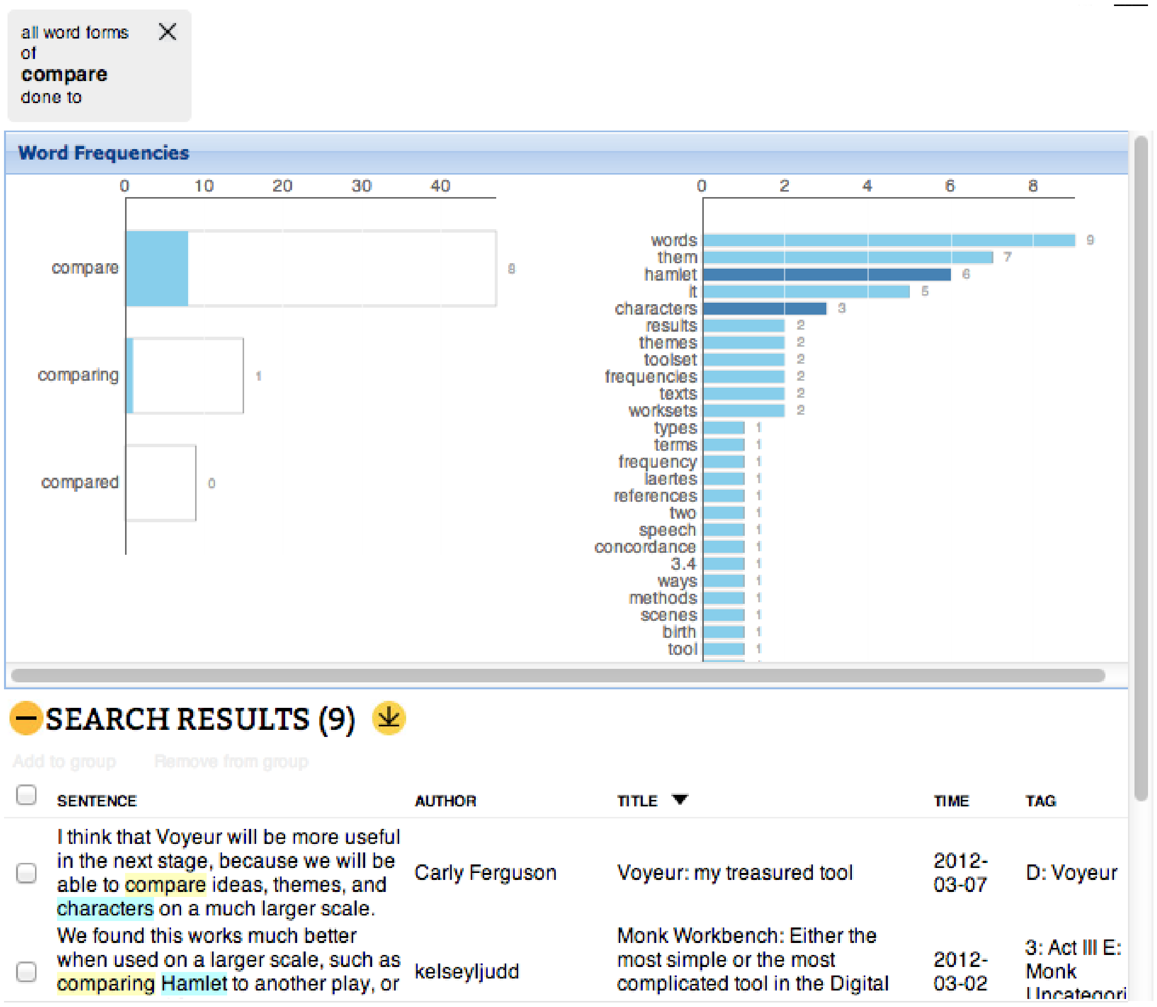


all word forms of **compare** done to

SEARCH RESULTS (63)

	SENTENCE	AUTHOR	TITLE	TIME	TAG
☐	Then I would want to compare her behaviour when she is with her father, to her behaviour when he is not around.	allison	New Perspective via the Integrated Force of Tradition and Digital Tools	2012-04-25	4: Act IV A: WordHoard Phase 3
☐	By comparing these results, I can tell that Gertrude is neither more or less negative before or after Polonius	allison	Battle with WordHoard? Challenge Accepted	2012-03-07	A: WordHoar

Figure 5.18: Searching for `compare (all word forms) [done to] ___` with the Grammatical Relations visualization.

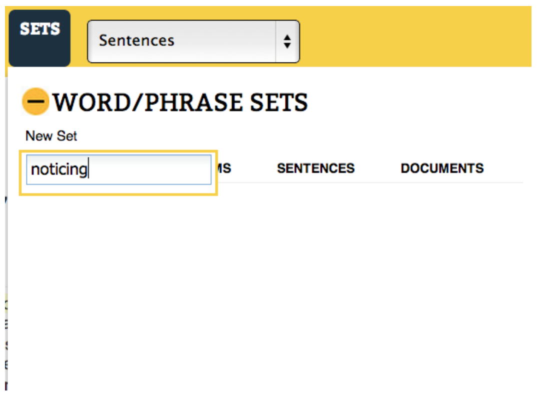


	SENTENCE	AUTHOR	TITLE ▼	TIME	TAG
☐	I think that Voyeur will be more useful in the next stage, because we will be able to compare ideas, themes, and characters on a much larger scale.	Carly Ferguson	Voyeur: my treasured tool	2012-03-07	D: Voyeur
☐	We found this works much better when used on a larger scale, such as comparing Hamlet to another play, or	kelseyljudd	Monk Workbench: Either the most simple or the most complicated tool in the Digital	2012-03-02	3: Act III E: Monk Uncategori

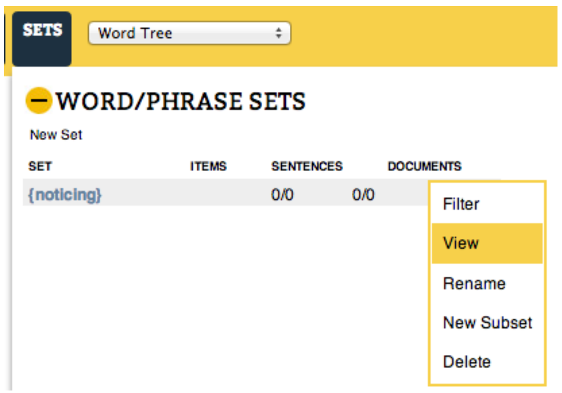
Figure 5.19: Using the bar charts to filter the list of sentences matching `compare (all word forms) [done to] ___` to just the words `hamlet` and `characters`.

Sets Overview

The Sets Overview above every View (Figure 5.20) shows the list of sets that the user has created. It allows users to create a new, empty set using the 'New Set' button. A menu appears containing a text field into which the user can enter the name of the new set (Figure 5.20a).



(a) Creating a new set. The name of the set can be typed into the text entry field.



(b) Clicking a set brings up a menu with options to use it as a filter, view its contents, rename it, create a new subset, or delete it.

Figure 5.20: The Sets Overview allow users to view and edit sets.

A newly-created set will appear as an entry in the Sets Overview under the appropriate category. Clicking on a set in this overview brings up options to delete the set, create a new subset, rename the set, view the set, or filter the current slice by the set (Figure 5.20b). A new set will have no items in it, and therefore no matching sentences or documents. As items are added, the sentence and document counts update. They show how many sentences or documents in the current slice match the set, out of the total number in that set. If the set is a Document Set, then the sentence count is the total number of sentences in the documents. If it is a Sentence Set, then the document count is the number of distinct documents to which the sentences belong. If it is a Phrase Set, the document and sentence counts are the number of documents and sentences that contain words or phrases in the set.

Once a Set is created, the search box shows a drop-down option for the set (Figure 5.21), which opens a slice containing the sentences that match that set.

Phrase Sets

Phrase Sets are collections of words or multi-word phrases. In addition to creation through the Sets Overview, they can be created on the go by clicking on a word or phrase and selecting the 'Add to Set' menu option in the Word Menu (described in Section 5.3.9) (Figure 5.22).

The contents of Phrase Sets can be viewed and edited by selecting the 'View' option in the Sets Overview menu (Figure 5.20b). This brings up a small window showing the words

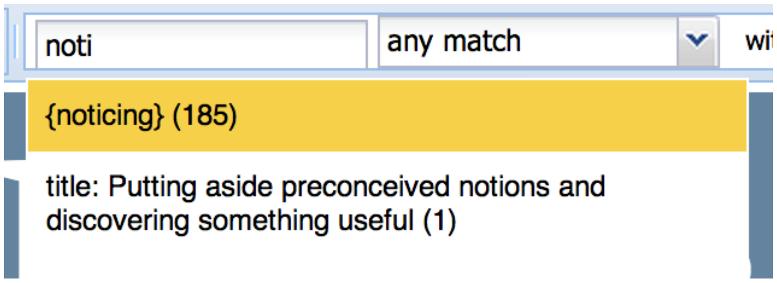

noti
any match
wi
{noticing} (185)
title: Putting aside preconceived notions and discovering something useful (1)

Figure 5.21: Sets appear in the auto-suggest options as the user types. If selected, they open the matching slice of sentences.

.

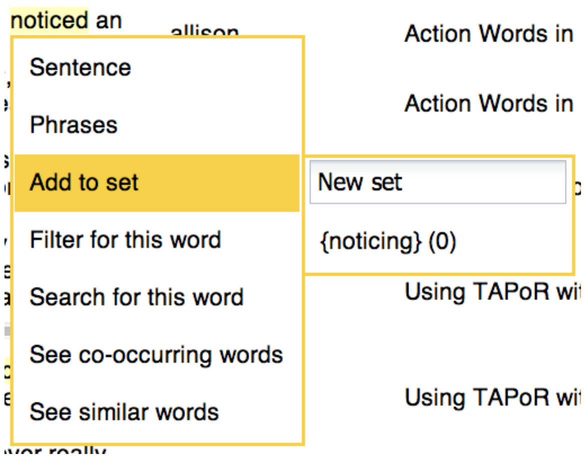
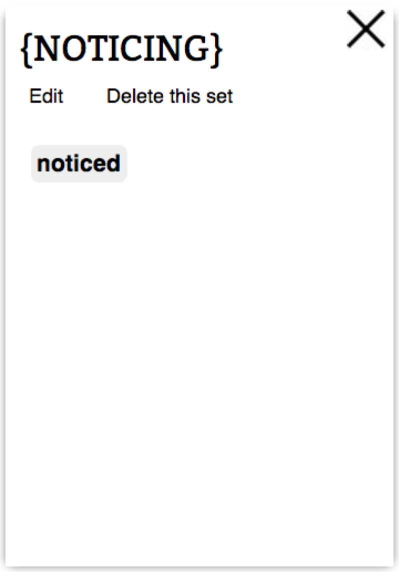
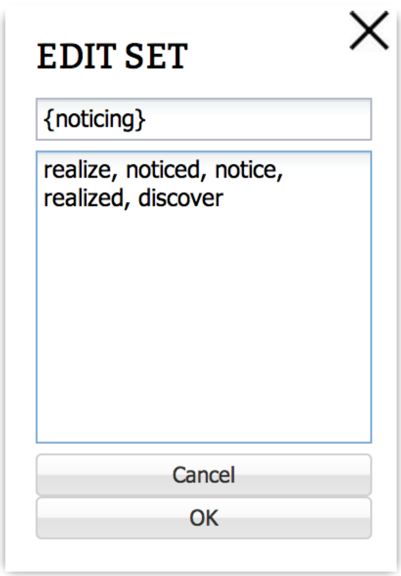


Figure 5.22: The Word Menu has options to add a word or phrase to existing sets, or to create a new set to which to add the term.



(a) Clicking on the 'View' menu option for a Phrase Set opens the word set in a small window.



(b) Clicking on the 'Edit Set' menu option in the window allows the user to type words and phrases directly into the set

Figure 5.23: Viewing a Phrase Set and editing its contents by typing.

in the set (Figure 5.23a). The window has buttons to edit and delete the set. If the 'Edit Set' button is clicked, the main window body is replaced by an editable text field. The user can change or remove the existing words and phrases in the set, or add new ones (Figure 5.23b).

Sentence Sets

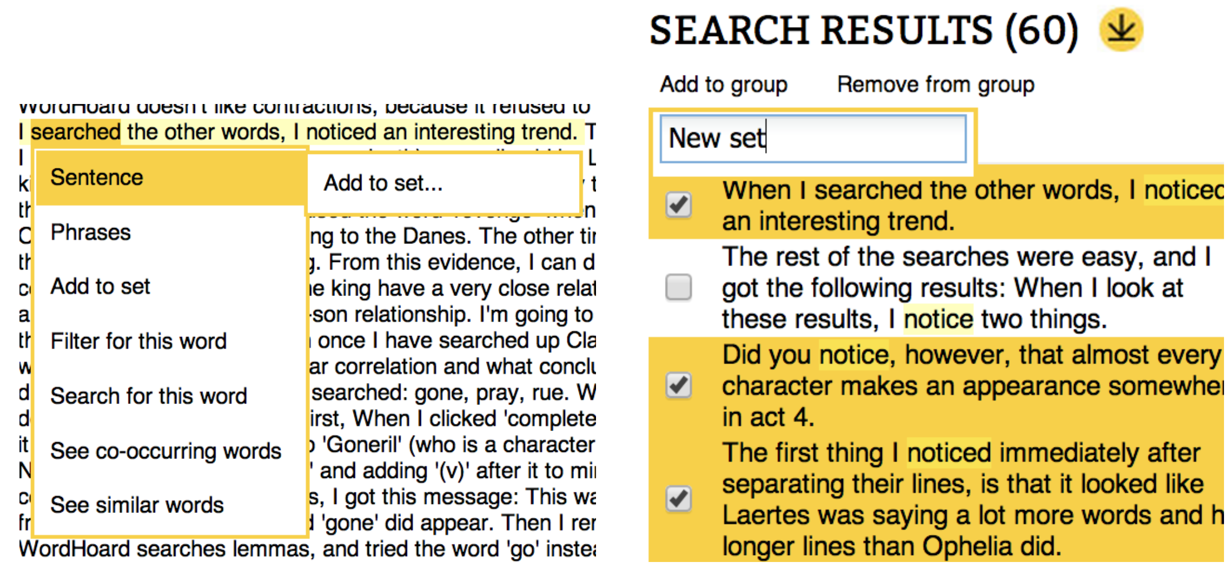
Sentence Sets are collections of sentences. Users can either hand-pick them while reading (Figure 5.24a) or select them from search results (Figure 5.24b).

Document Sets

Documents Sets are collections of documents. Users can select them from the Documents view (Figure 5.25), which lists the documents that match a slice.

5.3.9 The Word Menu

The Word Menu (Figure 5.26) is designed to allow the user to explore word usage without interrupting the current task. It appears when the user clicks on a word in search results, while reading, in frequent word overviews, or in Phrase Set listings. If the word appears as part of a sentence, then the menu also contains options for acting on that sentence and the sequences starting at that word. It has options for examining the grammatical relations in



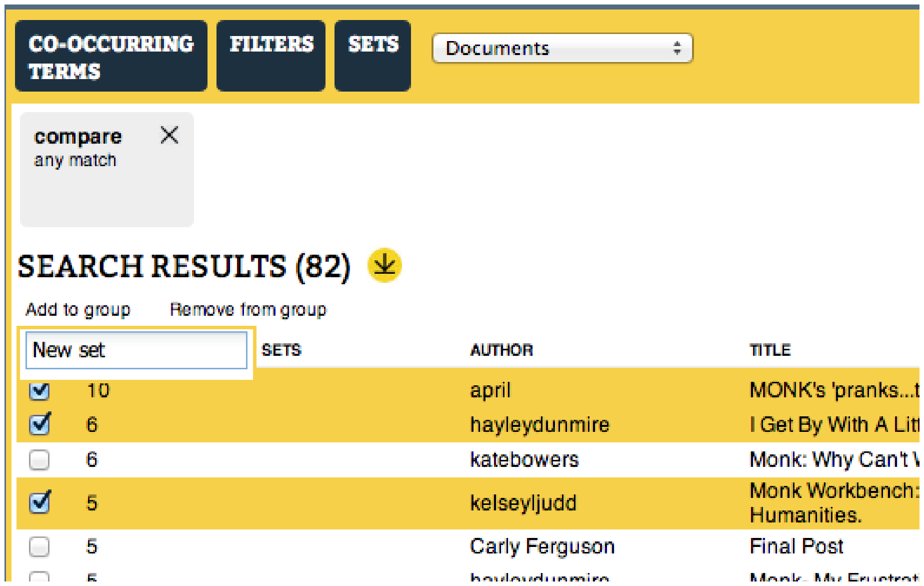
WordHoard doesn't like contractions, because it refused to
I searched the other words, I noticed an interesting trend. T
ng to the Danes. The other ti
g. From this evidence, I can d
he king have a very close relat
son relationship. I'm going to
once I have searched up Cla
ar correlation and what conclu
searched: gone, pray, rue. W
irst, When I clicked 'complete
'Goneril' (who is a character
' and adding '(v)' after it to mi
s, I got this message: This wa
'gone' did appear. Then I rer
WordHoard searches lemmas, and tried the word 'go' instea

✓	When I searched the other words, I noticed an interesting trend.
	The rest of the searches were easy, and I got the following results: When I look at these results, I notice two things.
✓	Did you notice, however, that almost every character makes an appearance somewhe in act 4.
✓	The first thing I noticed immediately after separating their lines, is that it looked like Laertes was saying a lot more words and h longer lines than Ophelia did.

(a) Sentences can be collected while reading using the Word Menu.

(b) Sentences can be collected from search results using the 'Add to group' button in the table header.

Figure 5.24: Adding sentences to Sentence Sets.



	SETS	AUTHOR	TITLE
✓	10	april	MONK's 'pranks...t
✓	6	hayleydunmire	I Get By With A Litt
	6	katebowers	Monk: Why Can't \
✓	5	kelseyljudd	Monk Workbench: Humanities.
	5	Carly Ferguson	Final Post
	5	hayleydunmire	Monk: My Frustrat

Figure 5.25: Documents can be collected from the Documents view.

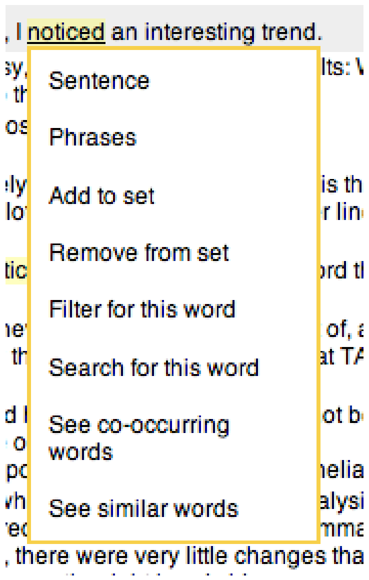


Figure 5.26: The Word Menu that appears when a word in a sentence is clicked. Here, we see the menu for the word `noticed`.

which a word occurs, seeing co-occurring words and words that occur in similar contexts, and performing a new search for the word.

The Phrases option (Figure 5.27) appears if the word appears in a sentence. It lists the 2, 3, and 4-word sequences starting at that word, and gives options to search for them or add them to Phrase Sets.

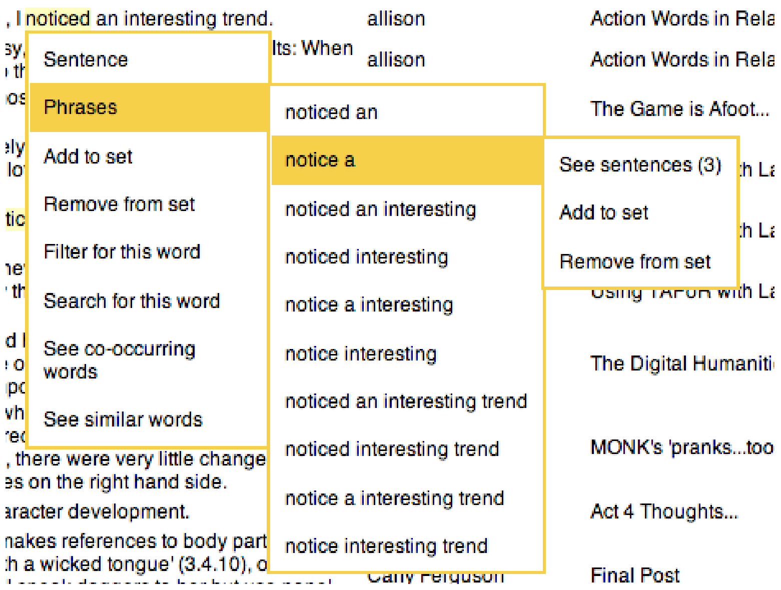
The 'Co-occurring Words' option (Figure 5.28) shows the nouns, verbs, adjectives and phrases that co-occur most frequently with the clicked-on word. Each of these related words can be clicked in turn, opening up a new Word Menu. These menus have the additional option to 'See co-occurences'. Selecting that option opens up a new view showing just the sentences in which the two words appear together.

The 'Similar Words' option (Figure 5.29) shows words that occur in similar contexts, computed according to the contextual similarity algorithm described by Lin (1997).

The 'Search for this word' option (Figure 5.30) shows the other words that have grammatical relations with the clicked-on word. Clicking on any of the options opens up a Grammatical Search for that specific relation.

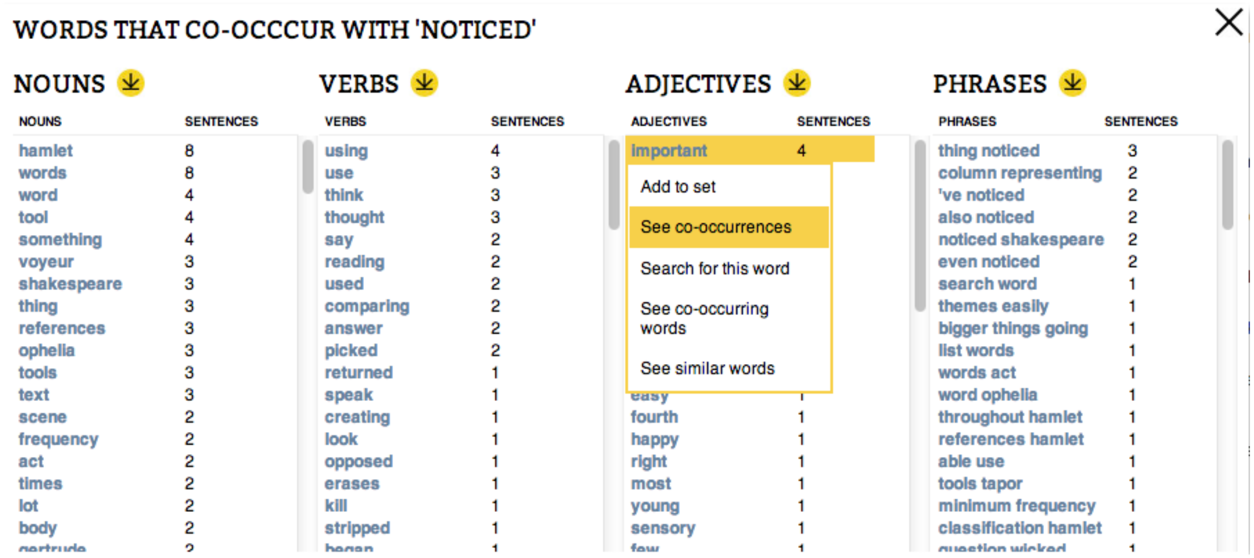
5.4 Example: The Importance of Comparison

We now illustrate the features described above with an example analysis of our own data: the 180 blog posts that the students in Case Study 2 posted while learning to analyze Hamlet with text analysis tools. We show how we used WordSeer 3 to discover three reasons why it was important to improve interface support for comparison. These were a) it was a



, there were very little change
es on the right hand side.
aracter development.
nakes references to body part
h a wicked tongue' (3.4.10), o

Figure 5.27: The Phrases option shows the phrases next to the clicked-on word along with their counts and options to search for them or add them to sets.



WORDS THAT CO-OCCCUR WITH 'NOTICED'

NOUNS

NOUNS	SENTENCES
hamlet	8
words	8
word	4
tool	4
something	4
voyeur	3
shakespeare	3
thing	3
references	3
ophelia	3
tools	3
text	3
scene	2
frequency	2
act	2
times	2
lot	2
body	2
gertrude	2

VERBS

VERBS	SENTENCES
using	4
use	3
think	3
thought	3
say	2
reading	2
used	2
comparing	2
answer	2
picked	2
returned	1
speak	1
creating	1
look	1
opposed	1
erases	1
kill	1
stripped	1
began	1

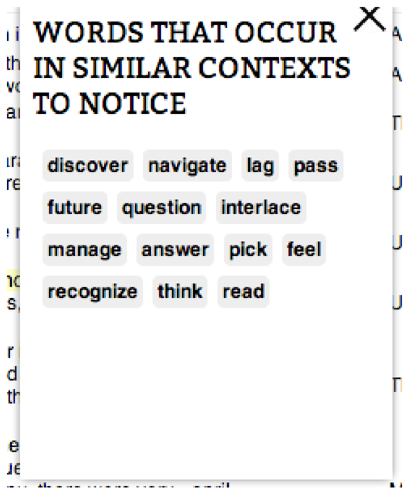
ADJECTIVES

ADJECTIVES	SENTENCES
important	4
easy	1
fourth	1
happy	1
right	1
most	1
young	1
sensory	1
few	1

PHRASES

PHRASES	SENTENCES
thing noticed	3
column representing	2
've noticed	2
also noticed	2
noticed shakespeare	2
even noticed	2
search word	1
themes easily	1
bigger things going	1
list words	1
words act	1
word ophelia	1
throughout hamlet	1
references hamlet	1
able use	1
tools tapor	1
minimum frequency	1
classification hamlet	1
question wicked	1

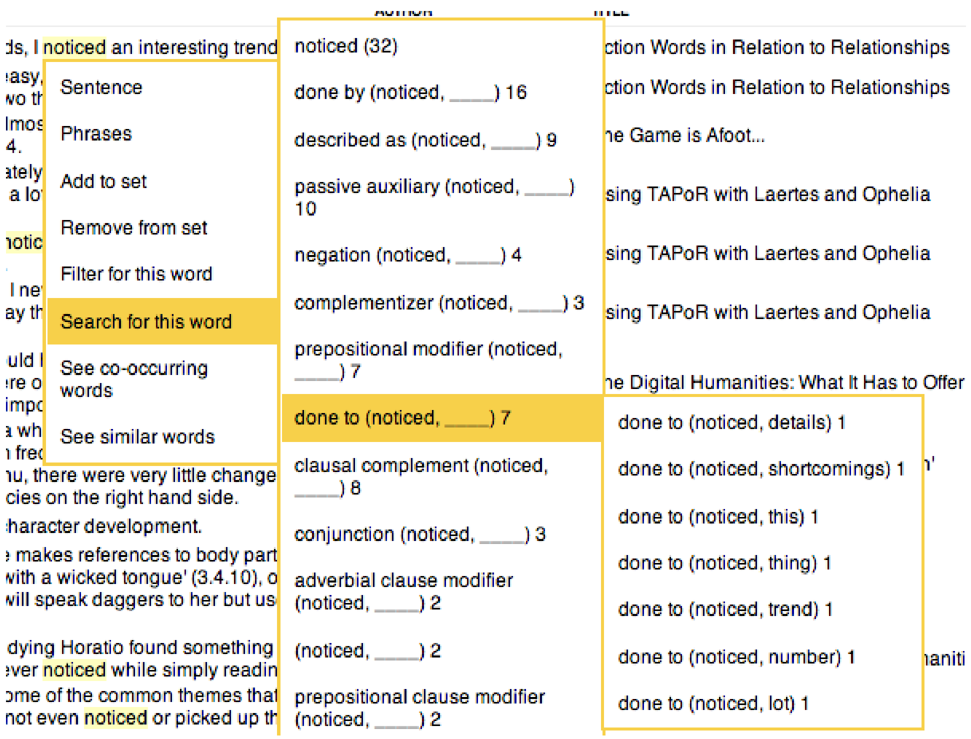
Figure 5.28: The 'co-occurring words' option shows the nouns, verbs, and adjectives that co-occur with the clicked-on word. The words in this list have an additional option in their word menu – 'see co-occurrences'. The user can click this option to see the sentences in which the words co-occur.



WORDS THAT OCCUR IN SIMILAR CONTEXTS TO NOTICE

discover navigate lag pass
future question interlace
manage answer pick feel
recognize think read

Figure 5.29: The 'Similar Words' option shows words that occur in similar contexts to the clicked-on word.



Sentence
Phrases
Add to set
Remove from set
Filter for this word
Search for this word
See co-occurring words
See similar words

noticed (32)
done by (noticed, ____) 16
described as (noticed, ____) 9
passive auxiliary (noticed, ____) 10
negation (noticed, ____) 4
complementizer (noticed, ____) 3
prepositional modifier (noticed, ____) 7
done to (noticed, ____) 7
clausal complement (noticed, ____) 8
conjunction (noticed, ____) 3
adverbial clause modifier (noticed, ____) 2
(noticed, ____) 2
prepositional clause modifier (noticed, ____) 2

done to (noticed, details) 1
done to (noticed, shortcomings) 1
done to (noticed, this) 1
done to (noticed, thing) 1
done to (noticed, trend) 1
done to (noticed, number) 1
done to (noticed, lot) 1

Figure 5.30: The search option lists the grammatical relations that the word enters into. It shows the count of each relation, under which it shows the counts of the other words that enter into that relation.

common activity, b) the comparisons that students were attempting could be more easily performed with computational assistance, and c) that it was useful because it led to interesting discoveries.

5.4.1 Prevalence of Comparative Activity

The initial overview of the collection of blog posts gave us the first indication of the prevalence of comparison. The overview was shown in Figure 5.1 earlier in this chapter, but the relevant section is reproduced in Figure 5.31. It reveals that `compare` (highlighted in yellow due to a mouse hover) was one of the most common words in the collection, with 83 matching sentences as a verb.

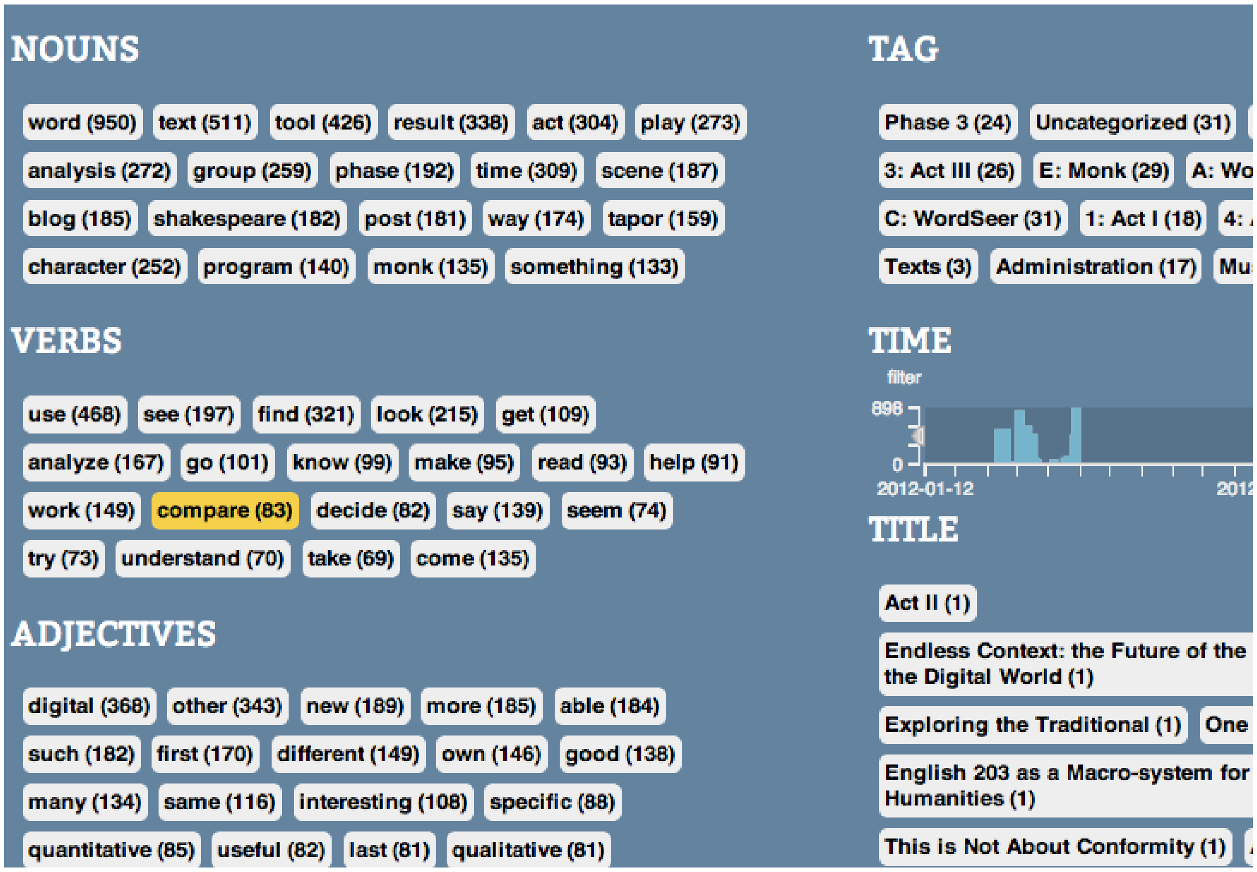


Figure 5.31: The initial view of the collection of blog posts in Case Study 2 (Chapter 3.3) shows that `compare` is a frequent verb, with 83 occurrences.

To investigate further, we decided to analyze a slice of sentences mentioning comparative activities. We did this by issuing a search for the Word Tree of sentences that contained any form of `compare` (the 'with stemming' option in the search box) (Figure 5.32).

However, we first needed to establish whether the high frequency of the verb `compare`

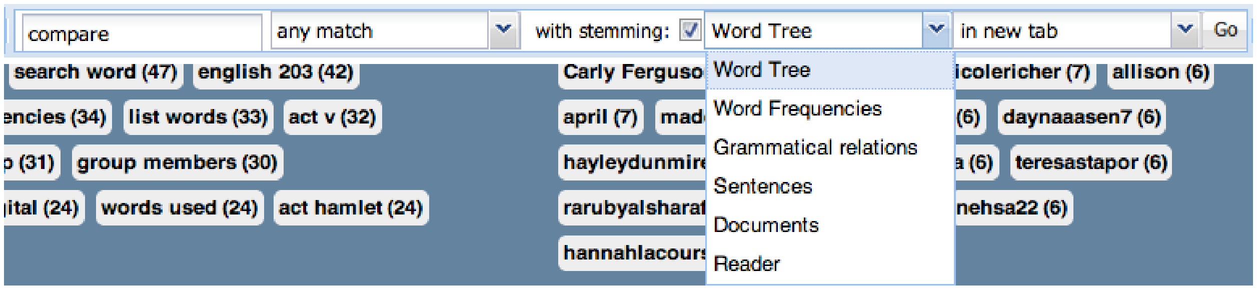


Figure 5.32: Using the search bar to open a Word Tree of all word forms of `compare`, in order to investigate its usage further.

that we had seen in the main overview (Figure 5.31) was due to comparison being a truly prevalent activity, or due to frequent mentions by just a few students. In order to do this, we looked at the Filters overview for that slice (Figure 5.33). The Filters overview revealed that comparison was truly prevalent. 21 of the 24 students mentioned some form of the word, in 80 of the 180 blog posts. Having established that comparison was a prevalent activity, we moved on to the next question: what were students comparing, and would they benefit from computational assistance in doing so?

5.4.2 Items Being Compared

Exploring the Word Tree (Figure 5.34) briefly, we saw that students were comparing word frequencies, as well as sections of *Hamlet* to other sections.

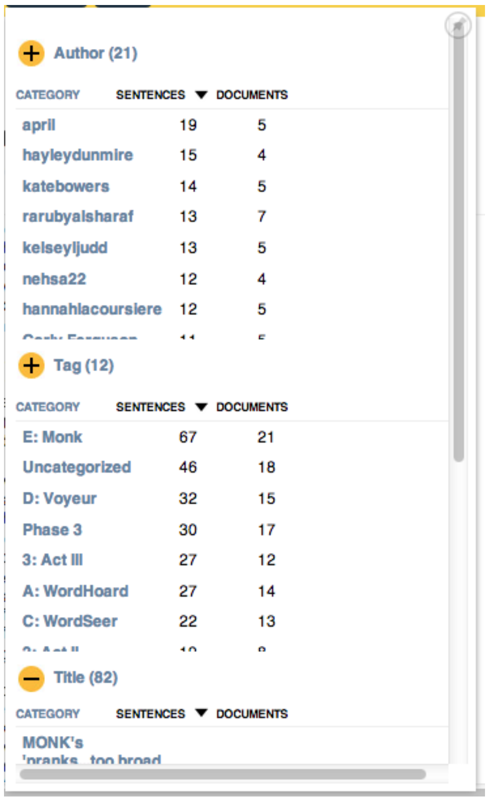
We used the Co-Occurring terms overview to get a more complete sense of the language around comparison (Figure 5.35). The overview showed that the most frequent noun was `word`. To investigate, we clicked it and selected the 'Filter for this word' option. The resulting overview (Figure 5.36) was even more informative. The frequent phrases showed that students were comparing `word frequencies` (5 sentences). The Frequent nouns showed that they were comparing these between `acts` (9) and `characters` (11).

The discovery that students were comparing word frequencies across sub-units of the play answered the question of whether comparison would benefit from computational assistance. Differences in word frequency can be assessed much more easily by comparing lists of words than by reading and keeping track, and sub-units of the play (especially speeches by a particular character) are time-consuming to isolate manually.

5.4.3 The Utility of Comparison

Our final question was whether comparison led to useful outcomes – meaning interesting discoveries. Our approach was to use WordSeer to assemble a set of sentences mentioning an interesting discovery, and to see whether some of them also mentioned comparison.

To gather a set of sentences mentioning interesting discoveries, we built up a set of terms that indicated noticing something interesting. We started with the word `notice` and added



Author (21)

CATEGORY	SENTENCES ▼	DOCUMENTS
april	19	5
hayleydunmire	15	4
katebowers	14	5
rarubyalsharaf	13	7
kelseyljudd	13	5
nehsa22	12	4
hannahlacoursiere	12	5

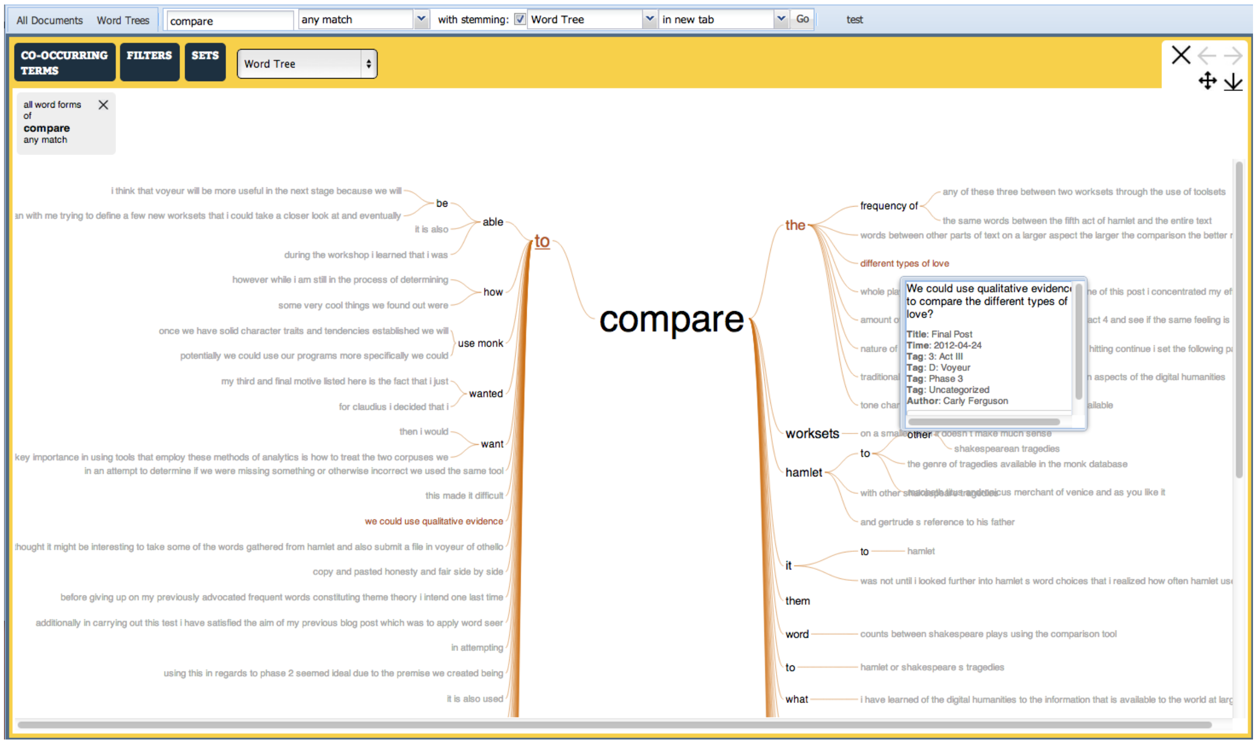
Tag (12)

CATEGORY	SENTENCES ▼	DOCUMENTS
E: Monk	67	21
Uncategorized	46	18
D: Voyeur	32	15
Phase 3	30	17
3: Act III	27	12
A: WordHoard	27	14
C: WordSeer	22	13

Title (82)

CATEGORY	SENTENCES ▼	DOCUMENTS
MONK's		

Figure 5.33: The Filters pane summarizes the metadata property values in sentences matching `compare` (all word forms). It shows the widespread usage of the term. There are posts from 21 different authors (out of 24), and 82 posts (out of 180). There are also 12 different tags. The most common tag is 'E: Monk', with 21 posts, Not surprising considering that Monk is primarily a comparison and classification tool (Clement *et al.* 2009).


All Documents Word Trees | compare | any match | with stemming: ☑ | Word Tree | in new tab | Go | test

CO-OCCURRING TERMS | FILTERS | SETS | Word Tree

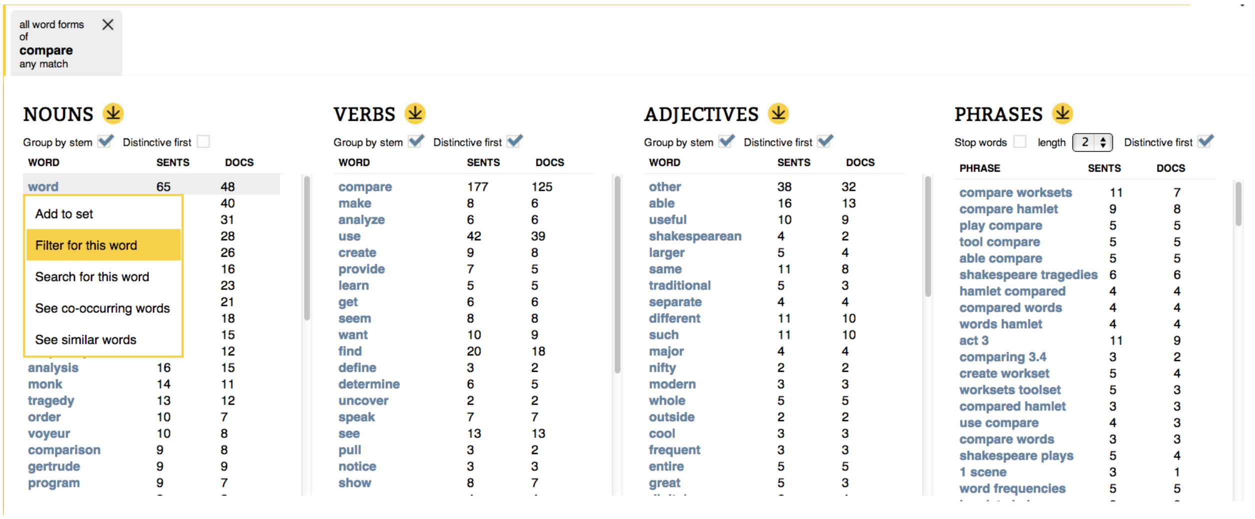
all word forms of **compare** any match

We could use qualitative evidence to compare the different types of love?

Title: Final Post
Time: 2012-04-24
Tag: 3: Act III
Tag: D: Voyeur
Tag: Phase 3
Tag: Uncategorized
Author: Carly Ferguson

compare
to
able
be
i think that voyeur will be more useful in the next stage because we will
an with me trying to define a few new worksets that i could take a closer look at and eventually
it is also
during the workshop i learned that i was
how
however while i am still in the process of determining
some very cool things we found out were
use monk
once we have solid character traits and tendencies established we will
potentially we could use our programs more specifically we could
wanted
my third and final motive listed here is the fact that i just
for claudius i decided that i
want
then i would
key importance in using tools that employ these methods of analytics is how to treat the two corpuses we
in an attempt to determine if we were missing something or otherwise incorrect we used the same tool
this made it difficult
we could use qualitative evidence
thought it might be interesting to take some of the words gathered from hamlet and also submit a file in voyeur of othello
copy and pasted honesty and fair side by side
before giving up on my previously advocated frequent words constituting theme theory i intend one last time
additionally in carrying out this test i have satisfied the aim of my previous blog post which was to apply word seer
in attempting
using this in regards to phase 2 seemed ideal due to the premise we created being
it is also used
the
frequency of
any of these three between two worksets through the use of toolsets
the same words between the fifth act of hamlet and the entire text
words between other parts of text on a larger aspect the larger the comparison the better
different types of love
whole pla
amount o
nature of
traditional
tone char
ne of this post i concentrated my ef
act 4 and see if the same feeling is
hitting continue i set the following pa
n aspects of the digital humanities
ailable
worksets
on a smalle
it doesn t make much sense
hamlet
to
other
shakespearean tragedies
the genre of tragedies available in the monk database
macbeth titus andronicus merchant of venice and as you like it
with other shakespeare tragedies
and gertrude s reference to his father
it
to
hamlet
was not until i looked further into hamlet s word choices that i realized how often hamlet use
them
word
counts between shakespeare plays using the comparison tool
to
hamlet or shakespeare s tragedies
what
i have learned of the digital humanities to the information that is available to the world at larg

Figure 5.34: This figure shows the Word Tree of the slice produced by the search for all word forms of `compare`. Here, we have clicked on the prefix 'to', meaning that we see the tree for sentences containing only `'to compare'`. We have also hovered over a grey 'leaf' of the tree – indicating a single sentence. The Word Tree shows that students talk about comparing word frequencies, and parts of the play *Hamlet* to other parts.



NOUNS

Group by stem ✓ Distinctive first ☐

WORD	SENTS	DOCS
word	65	48
		40
		31
		28
		26
		16
		23
		21
		18
		15
		12
analysis	16	15
monk	14	11
tragedy	13	12
order	10	7
voyeur	10	8
comparison	9	8
gertrude	9	9
program	9	7

VERBS

Group by stem ✓ Distinctive first ✓

WORD	SENTS	DOCS
compare	177	125
make	8	6
analyze	6	6
use	42	39
create	9	8
provide	7	5
learn	5	5
get	6	6
seem	8	8
want	10	9
find	20	18
define	3	2
determine	6	5
uncover	2	2
speak	7	7
see	13	13
pull	3	2
notice	3	3
show	8	7

ADJECTIVES

Group by stem ✓ Distinctive first ✓

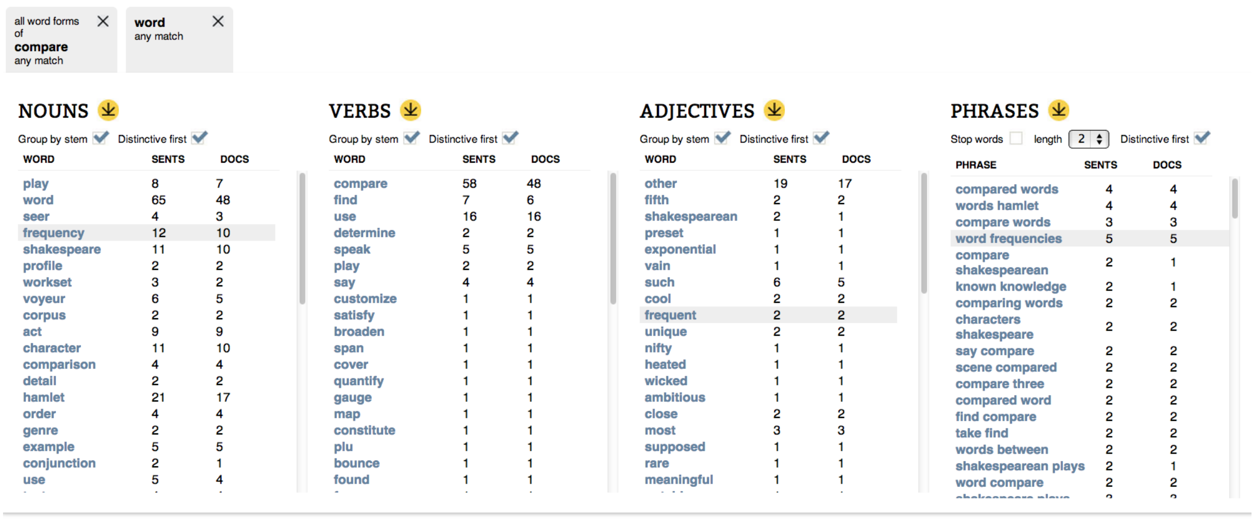
WORD	SENTS	DOCS
other	38	32
able	16	13
useful	10	9
shakespearean	4	2
larger	5	4
same	11	8
traditional	5	3
separate	4	4
different	11	10
such	11	10
major	4	4
nifty	2	2
modern	3	3
whole	5	5
outside	2	2
cool	3	3
frequent	3	3
entire	5	5
great	5	3

PHRASES

Stop words ☐ length 2 Distinctive first ✓

PHRASE	SENTS	DOCS
compare worksets	11	7
compare hamlet	9	8
play compare	5	5
tool compare	5	5
able compare	5	5
shakespeare tragedies	6	6
hamlet compared	4	4
compared words	4	4
words hamlet	4	4
act 3	11	9
comparing 3.4	3	2
create workset	5	4
worksets toolset	5	3
compared hamlet	3	3
use compare	4	3
compare words	3	3
shakespeare plays	5	4
1 scene	3	1
word frequencies	5	5

Figure 5.35: The Co-Occurring Terms overview shows that the most frequent noun that co-occurs with `compare` is `word`. Upon discovering this, we filtered the slice further, using the contextual menu, to get the sentences containing both.



NOUNS

Group by stem ✓ Distinctive first ✓

WORD	SENTS	DOCS
play	8	7
word	65	48
seer	4	3
frequency	12	10
shakespeare	11	10
profile	2	2
workset	3	2
voyeur	6	5
corpus	2	2
act	9	9
character	11	10
comparison	4	4
detail	2	2
hamlet	21	17
order	4	4
genre	2	2
example	5	5
conjunction	2	1
use	5	4

VERBS

Group by stem ✓ Distinctive first ✓

WORD	SENTS	DOCS
compare	58	48
find	7	6
use	16	16
determine	2	2
speak	5	5
play	2	2
say	4	4
customize	1	1
satisfy	1	1
broaden	1	1
span	1	1
cover	1	1
quantify	1	1
gauge	1	1
map	1	1
constitute	1	1
plu	1	1
bounce	1	1
found	1	1

ADJECTIVES

Group by stem ✓ Distinctive first ✓

WORD	SENTS	DOCS
other	19	17
fifth	2	2
shakespearean	2	1
preset	1	1
exponential	1	1
vain	1	1
such	6	5
cool	2	2
frequent	2	2
unique	2	2
nifty	1	1
heated	1	1
wicked	1	1
ambitious	1	1
close	2	2
most	3	3
supposed	1	1
rare	1	1
meaningful	1	1

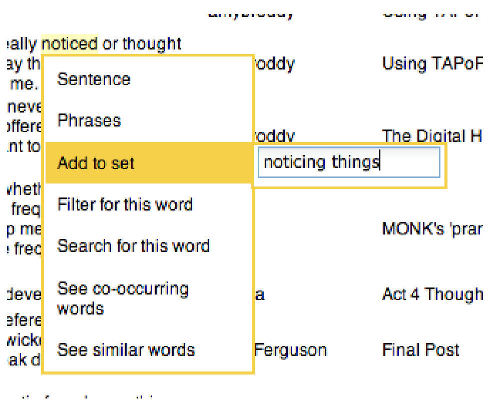
PHRASES

Stop words ☐ length 2 Distinctive first ✓

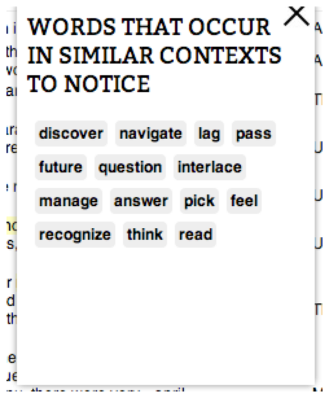
PHRASE	SENTS	DOCS
compared words	4	4
words hamlet	4	4
compare words	3	3
word frequencies	5	5
compare shakespearean	2	1
known knowledge	2	1
comparing words	2	2
characters shakespeare	2	2
say compare	2	2
scene compared	2	2
compare three	2	2
compared word	2	2
find compare	2	2
take find	2	2
words between	2	2
shakespearean plays	2	1
word compare	2	2

Figure 5.36: When we filtered the search for `compare` by the term `word`. The Nouns in the Co-Occurring terms overview showed that students compared word frequencies and word usage across `acts` (9 sentences) and `characters` (11 sentences), focusing on `word frequencies` (5 sentences) and differences between who `speaks` them (5 sentences).

it to a Phrase Set (Figure 5.37a). To expand the search to other ways of phrasing this, we looked at the 'similar words' to `notice` (Figure 5.37b), which are words that occur in similar contexts (see Chapter 3.1) and added the related words `realized` and `discovered`. Eventually, we settled on the set `{notice, realize, discover, recognize}` and all their word forms.

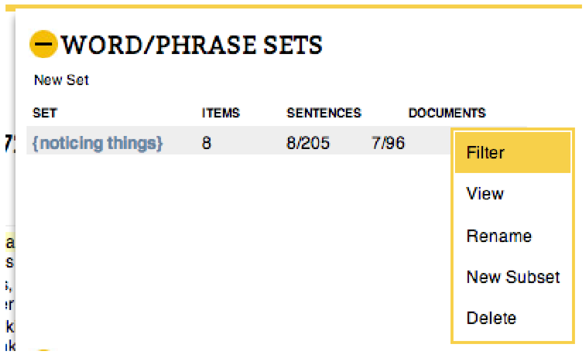


(a) Building up a set of terms that indicate noticing something interesting, starting with the word 'notice'.



(b) Expanding the set by examining words that occur in similar contexts to 'notice'. The words 'discover' and 'recognize' seemed promising.

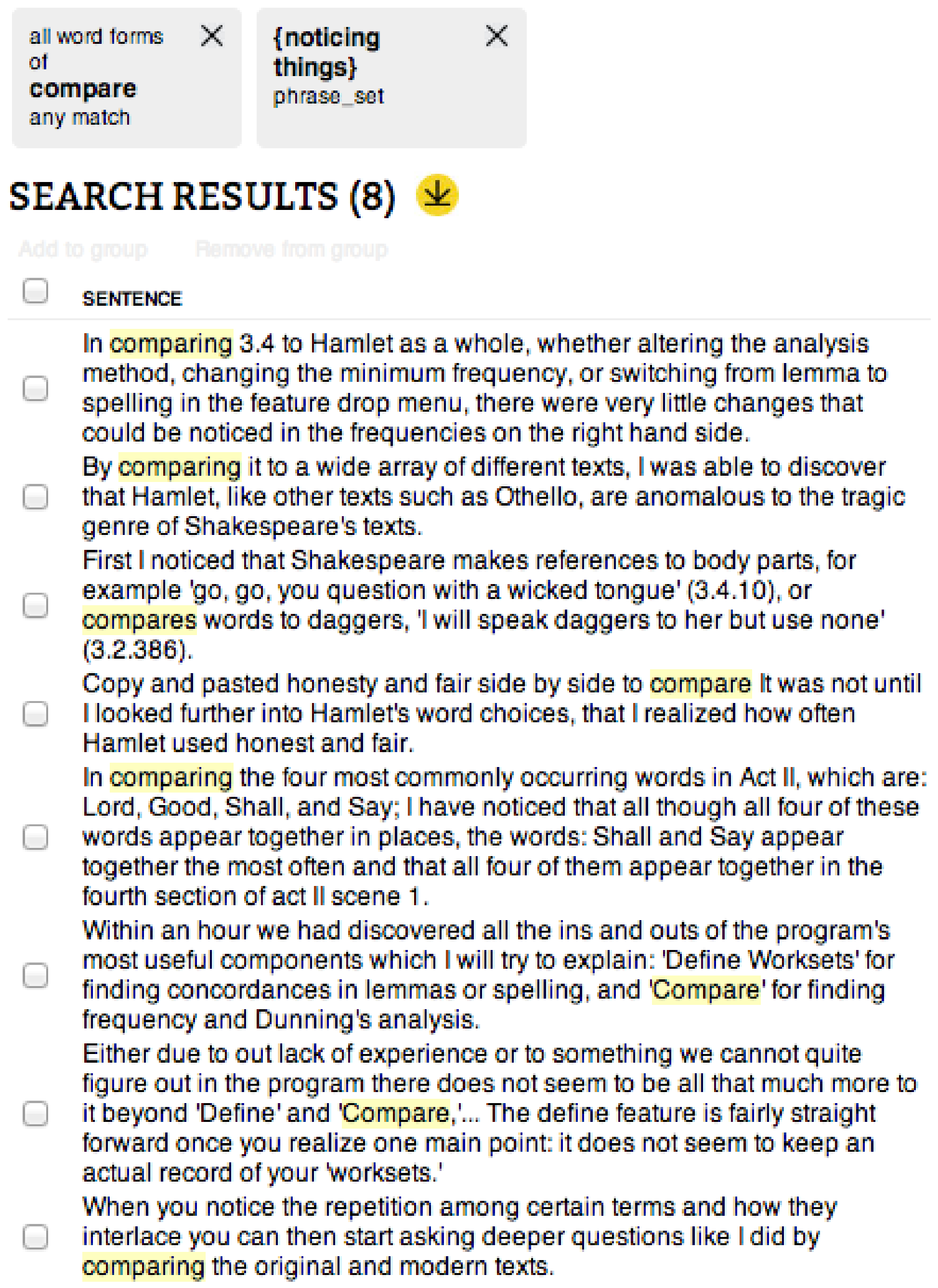
To perform the intersection between this set of terms and comparison, we searched for all word forms of `compare`, and the Sets overview showed that there were 8 sentences that were also in the `{noticing things}` set (Figure 5.38).



SET	ITEMS	SENTENCES		DOCUMENTS
{noticing things}	8	8/205	7/96	

Figure 5.37: The Sets Overview showed that there were 8 sentences that mentioned both `compare` and a word from the `{noticing things}` set. Here, we are filtering the slice by the Phrase Set `{noticing things}` in order to see those sentences.

We checked these sentences (Figure 5.38), and found that six of them were mentions of comparisons directly leading to an interesting discovery. We also found that these sentences particularly referenced side-by-side comparison between two results. We therefore had the answers we were looking for about comparison: a) it was prevalent, b) the kinds



SEARCH RESULTS (8)

SENTENCE

In comparing 3.4 to Hamlet as a whole, whether altering the analysis method, changing the minimum frequency, or switching from lemma to spelling in the feature drop menu, there were very little changes that could be noticed in the frequencies on the right hand side.

By comparing it to a wide array of different texts, I was able to discover that Hamlet, like other texts such as Othello, are anomalous to the tragic genre of Shakespeare's texts.

First I noticed that Shakespeare makes references to body parts, for example 'go, go, you question with a wicked tongue' (3.4.10), or compares words to daggers, 'I will speak daggers to her but use none' (3.2.386).

Copy and pasted honesty and fair side by side to compare It was not until I looked further into Hamlet's word choices, that I realized how often Hamlet used honest and fair.

In comparing the four most commonly occurring words in Act II, which are: Lord, Good, Shall, and Say; I have noticed that all though all four of these words appear together in places, the words: Shall and Say appear together the most often and that all four of them appear together in the fourth section of act II scene 1.

Within an hour we had discovered all the ins and outs of the program's most useful components which I will try to explain: 'Define Worksets' for finding concordances in lemmas or spelling, and 'Compare' for finding frequency and Dunning's analysis.

Either due to out lack of experience or to something we cannot quite figure out in the program there does not seem to be all that much more to it beyond 'Define' and 'Compare,'... The define feature is fairly straight forward once you realize one main point: it does not seem to keep an actual record of your 'worksets.'

When you notice the repetition among certain terms and how they interlace you can then start asking deeper questions like I did by comparing the original and modern texts.

Figure 5.38: The 8 sentences that mention some form of the word `compare` and also a term from the `{noticing things}` set (which includes all word forms of `notice`, `discover`, `realize` and `recognize`.

of comparisons students were doing could benefit from computational assistance, and c) comparisons (particularly side-by-side comparisons) had led to interesting discoveries.

5.5 Summary

In this chapter, we distilled the difficulties and common workflows identified in the previous two chapters into a concrete set of design requirements for text analysis systems. We then proposed a set of design guidelines, encapsulated by the principles of 'text sliding' to satisfy these requirements. We described WordSeer 3, which has been designed according to text sliding principles, and showed how we used it to analyze the value of supporting comparative analysis, based on the blog posts that that students posted while analyzing Hamlet in Case Study 2 (Chapter 3.3).

In the next chapter, we validate the redesign with three case studies. We show how scholars were able to use the system to access otherwise important knowledge that was useful to their existing research questions.

Chapter 6

Case Studies

The lessons learned from Chapters 3.3 and 4 led to a complete redesign of the WordSeer interface. The resulting system, described in the previous chapter, helped scholars make new discoveries about the world and create new understanding from text collections; these are discoveries that most likely would not have been able to make with any other existing tool.

This chapter describes the three case studies. The first was with a scholar studying U.S.-China relations through thousands of *New York Times* articles[1] . The second was with a writing instructor re-examining his students written assignments. The third study was with scholars studying the transition from religious to secular connotations for certain words in early American fiction.

The screenshots we show in this chapter are from the orignial WordSeer 3. After the case studies were completed, we redesigned the interface and the back end to improve the visual design and responsiveness of the interface. The previous chapter uses the redesigned interface.

6.1 U.S. Perceptions of China and Japan

Christoper Fan (C.F.) is a Ph.D. candidate at UC Berkeley's English department. Literary scholars typically allow their claims to rest on observations made by field experts like historians and sociologists, or on their own inductive reasoning, but C.F. wanted to verify some of those observations by gathering as much empirical evidence for them as possible.

Using the Lexis/Nexis database, C.F. collected *New York Times* editorials from 1980 to 2012, limiting his collection to editorials tagged with subjects "China" or "Japan." This left a set of 5,715 editorials, which we imported to WordSeer. Each editorial was associated with three pieces of metadata: year, month, and country (either China or Japan).

After a few weeks of face-to-face meetings with us, during which C.F. learned the system, and we made small adjustments to the interface to meet his analysis needs, C.F. gradually became comfortable using WordSeer on his own. He used WordSeer to find evidence for

[1]A condensed report on this case study study was included in Muralidharan *et al.* (2013).

discussion of the rise of China as described above, and for three other hypotheses, two of which are described below.

6.1.1 Initial Verification

One of C.F.'s first goals was to verify a well-known shift in the way China was percieved during the '80s, '90s and '00s: it stopped being seen primarily as a communist nation and Russian ally, and began to be regarded as an economic and political world power. To do this, C.F. assembled three slices, one for each decade, by starting with a *search* for "China" (Figure 6.1) and then filtering the 'Year' category to range over each ten-year period (Figure 6.2a).

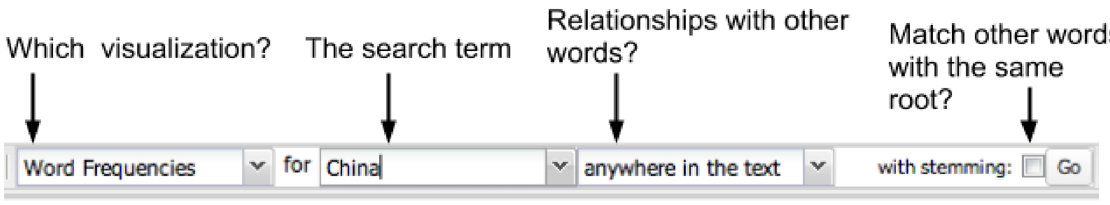


Figure 6.1: Searching for sentences matching "China".

The increasing frequencies of growth-related verbs confirmed the increasing discussion of China's rise, as shown by the frequencies per decade below:

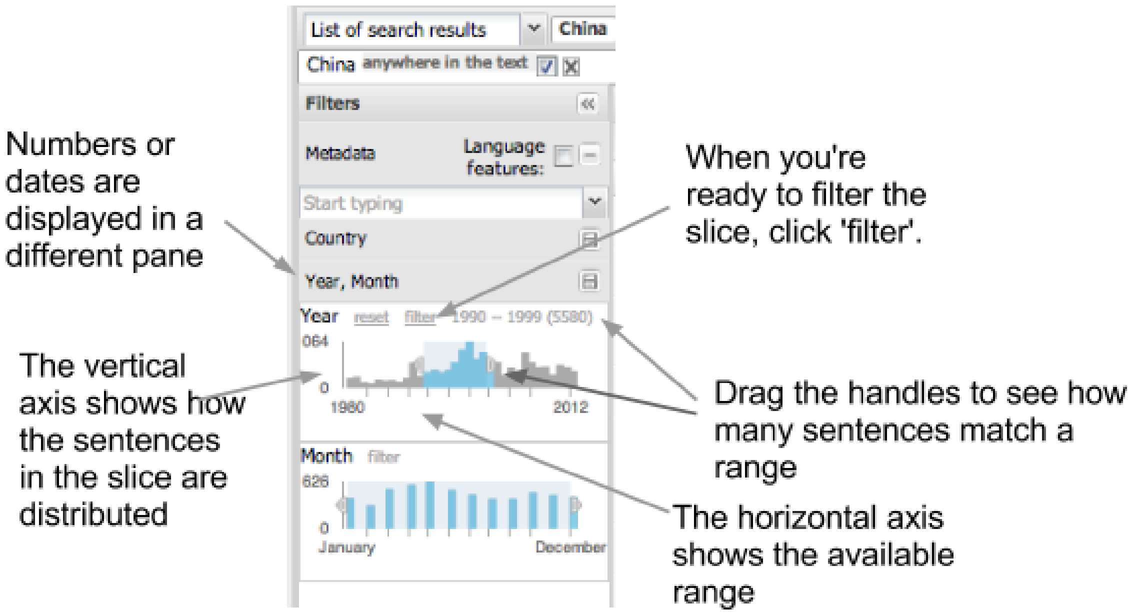
- "grow, growing": 294, 232, 421
- "rise, rising": 101, 134, 249
- "develop, developing": 274, 404, 476

Another way he checked this was by creating a phrase set of the words "growing, develop, developing, grow, rise, rising". The result was a Set representing a new slice of sentences, those containing at least one of those words.

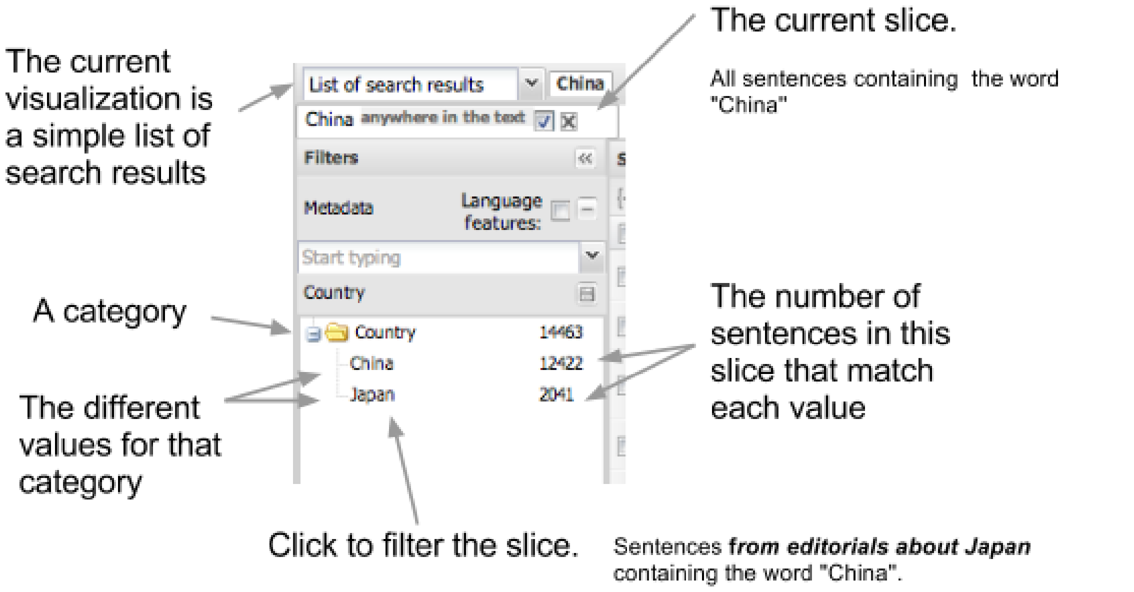
To verify that China was indeed described as rising, C.F. first filtered to just the editorials about China with the `country = China` filter, and then opened up a Word Frequencies view with a search on the `{growth}` Phrase Set. To avoid effects due to differing numbers of articles in different years, he normalized the graph by switching to percentages instead of raw frequencies. The resulting visualization is Figure 6.3, which shows almost a doubling of the frequency of these words in editorials about China over the 30-year period from 1980 to 2012. Satisfied that WordSeer was capable of reproducing this widely accepted fact, he was able to move on to deeper questions.

6.1.2 1980's: China Insignificant Except for Cold War Strategy

While comparing the most frequent adjectives for the three decades, C.F noticed a drop-off in cold war words. "Soviet" went from a count of 1029 in the 1980's to only 80 in the 2000's,


List of search results
China
China anywhere in the text
Filters
Metadata
Language features:
Start typing
Country
Year, Month
Year reset filter 1990 -- 1999 (5580)
064
0
1980
2012
Month filter
626
0
January
December
Numbers or dates are displayed in a different pane
When you're ready to filter the slice, click 'filter'.
The vertical axis shows how the sentences in the slice are distributed
Drag the handles to see how many sentences match a range
The horizontal axis shows the available range

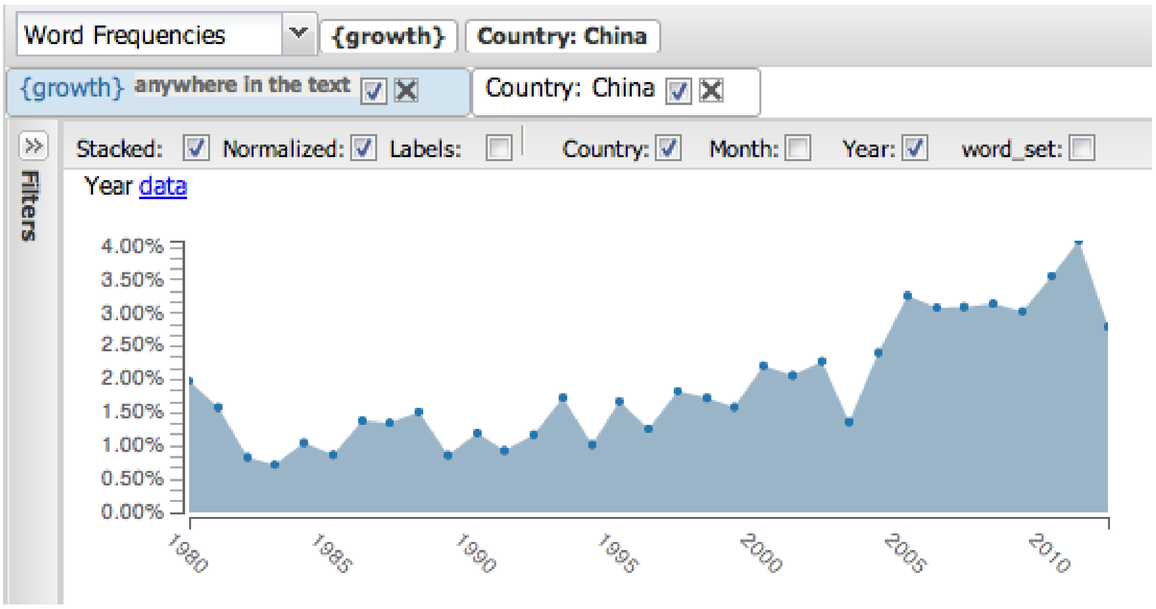
(a) C.F. used the date-range overview to select the slice "all sentences from the 1990's matching *China*"


List of search results
China
China anywhere in the text
Filters
Metadata
Language features:
Start typing
Country
Country 14463
China 12422
Japan 2041
The current visualization is a simple list of search results
The current slice.
All sentences containing the word "China"
A category
The different values for that category
The number of sentences in this slice that match each value
Click to filter the slice.
Sentences from editorials about Japan containing the word "China".

(b) Clicking on a value will filter the slice to match. For example, clicking on 'China' would create the slice "all sentences containing *China* from editorials about China".

Figure 6.2: Both types of overviews double as filters .

and "communist" went from 284 to 112. To investigate the drop-off in more detail, he used the Word Menu to open up word frequency plots for these two words over time, and filtered



Stacked: ✓ Normalized: ✓ Labels: ☐ Country: ✓ Month: ☐ Year: ✓ word_set: ☐

Figure 6.3: The **{growth}** Phrase Set used as a Word Frequencies query. The plot shows the percentage of sentences in editorials in the `country = China` category that contain words in that set.

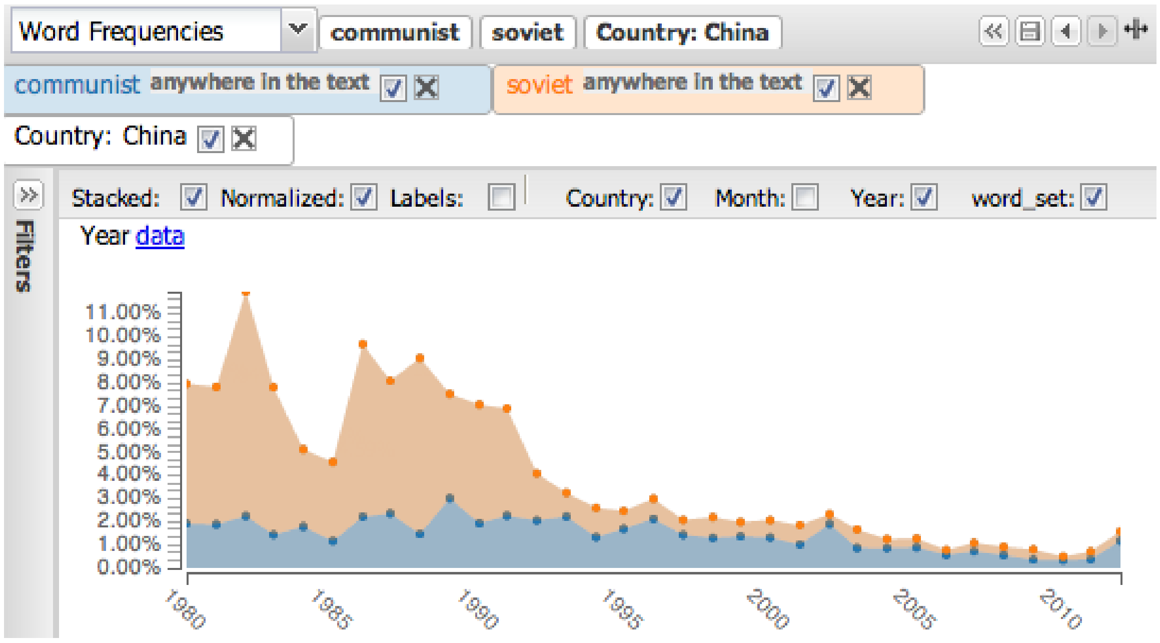
them to just the editorials about China.

Plotting the terms together (Figure 6.4) added depth to his initial calculations. The plot shows the dominance, and equally dramatic drop-off, in cold war mentions over this time period. In the early 1980's, almost 11% – more than 1 in 10 sentences in those editorials – mentions one of these the cold war terms. By the 1990's, however, this association is down to a trickle.

An exploration of the "grammatical neighborhood" around 'China' helped C.F. find yet more evidence for this idea. This time, it was through a distinctive rhetorical device. As Figure 6.5 shows, clicking on the word "China" opens up a menu with search options. These options act as previews of different grammatically-related slices: they show the frequencies with which "China" occurs in different grammatical constructions with different words. The noun compound relationship revealed the construction, "China card," which appeared 21 times. C.F. had never encountered this construction before, and found it odd.

He explored its usage by opening up the list of sentences in which the noun compound "China card" occurred. When C.F. read the sentences, an interpretation suggested itself. Summarizing his findings, he said:

> [Reading] the sentence search results reveals that phrase is used to refer to the China's strategic value in Cold War geopolitics. Of the four post-2000 instances, only one uses the phrase to describe a contemporary political situation; the others use it to describe Cold War politics. Reducing the vastness of China to



Stacked: [✓] Normalized: [✓] Labels: [ ] Country: [✓] Month: [ ] Year: [✓] word_set: [✓]

Figure 6.4: The percentage of sentences containing `communist` (blue) and `soviet` (orange) in editorials in the `country = China` category over time.

.

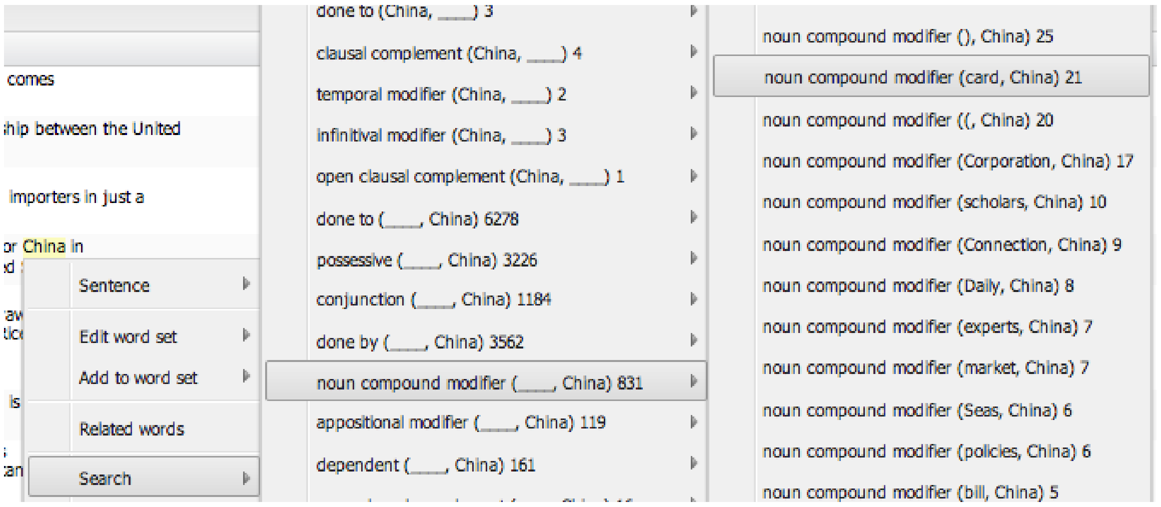

comes
ship between the United
importers in just a
or China in
Sentence
Edit word set
Add to word set
Related words
Search
done to (China, ____) 3
clausal complement (China, ____) 4
temporal modifier (China, ____) 2
infinitival modifier (China, ____) 3
open clausal complement (China, ____) 1
done to (____, China) 6278
possessive (____, China) 3226
conjunction (____, China) 1184
done by (____, China) 3562
noun compound modifier (____, China) 831
appositional modifier (____, China) 119
dependent (____, China) 161
noun compound modifier (), China) 25
noun compound modifier (card, China) 21
noun compound modifier ((, China) 20
noun compound modifier (Corporation, China) 17
noun compound modifier (scholars, China) 10
noun compound modifier (Connection, China) 9
noun compound modifier (Daily, China) 8
noun compound modifier (experts, China) 7
noun compound modifier (market, China) 7
noun compound modifier (Seas, China) 6
noun compound modifier (policies, China) 6
noun compound modifier (bill, China) 5

Figure 6.5: The word menu for `China` showing the noun compounds in which it participates. C.F. noticed an odd construction `China card`.

.

> a disposable "card," indicates a degree of U.S. self-confidence, not to mention condescension, that disappears after 1989.

The Word Frequencies view of the same data allowed him to make a temporal claim. Using the drop-down menu at the top of the panel, he opened up the word frequencies view alongside his list of search results. Because the "year" metadata was attached to each article, this view displayed a graph of how frequent sentences containing "China Card" were over time (Figure 6.6). The graph shows that "China card" was used until 1989, but only 4 times after

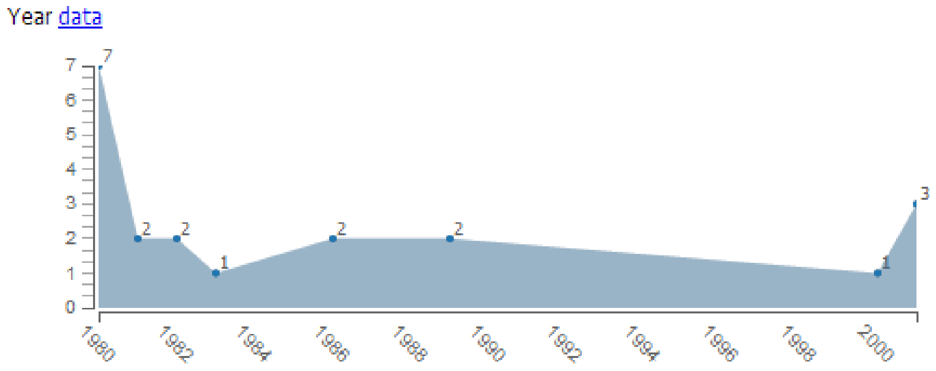


Figure 6.6: The number of sentences matching "China card" over time.

that, all in the 2000's. Upon investigating the occurrences individually by opening up a list of sentences, C.F. found that that in the 80s, the term was a rhetorical figure signifying cold-war strategic value, and the 2000's instances were historical references to those earlier attitudes. This evidence supported his claim that, up until the late 80's, China had a Cold-war strategic significance to the U.S. that later dissipated.

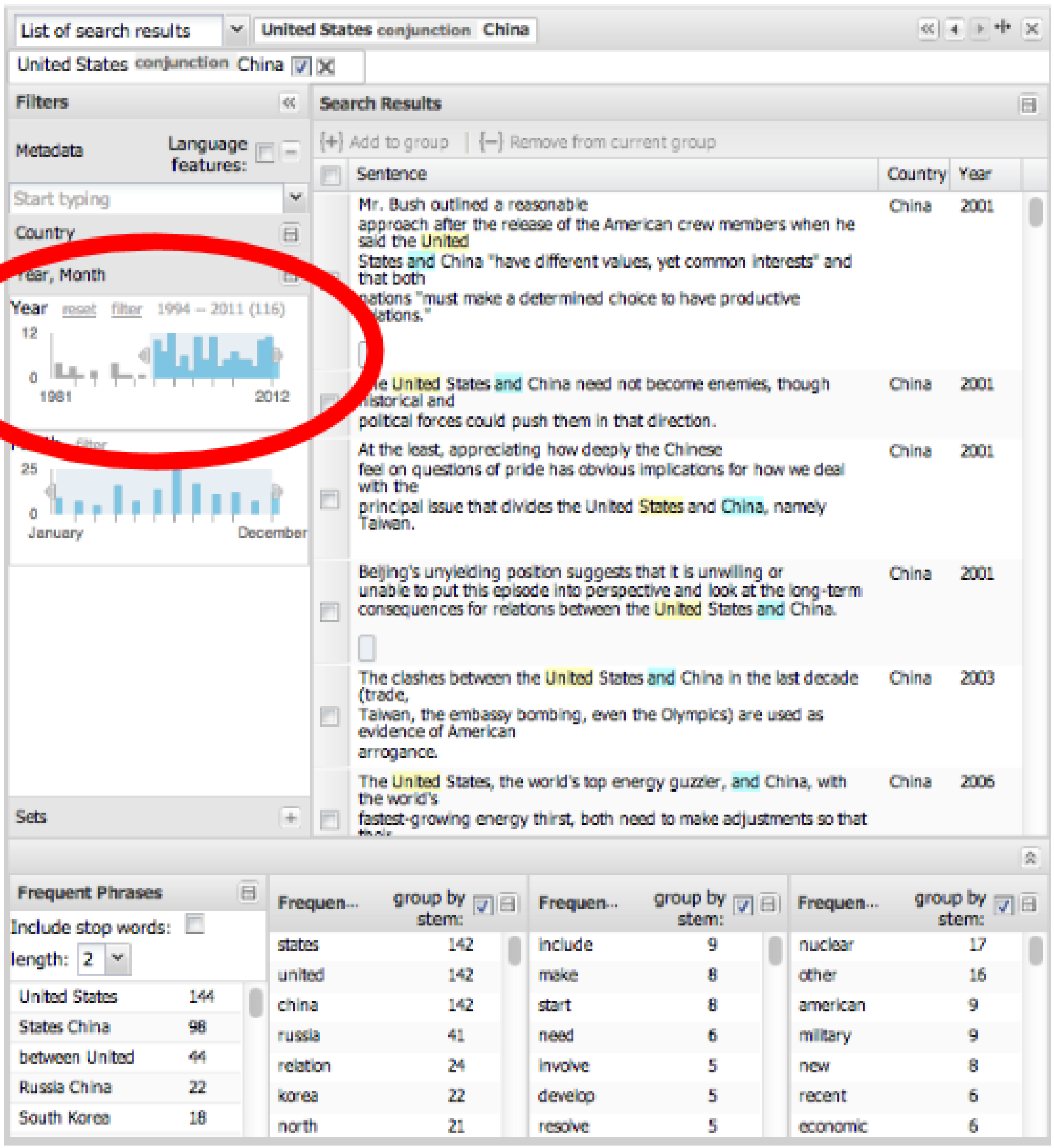
6.1.3 Mid 1990's Onwards: China's Ties With the U.S. Strengthen

A major event in Chinese international relations occurred in 2001, when China joined the World Trade Organization (WTO). Although attention had been shifting towards China since the 1990's, it is during this period that U.S.-China relations are thought to have become more interdependent, and central to global politics.

To find evidence for this, C.F. started with a grammatical structure he thought might make a good proxy for U.S.-China relations: the *conjunction*, which is an *and* relationship between two concepts. He needed a grammatical search because of the possibility of constructions like

> "The United States, the world's top energy guzzler, and China, with the world's fastest-growing energy thirst . . . " (April 2006).

This fragment places China and the United States in clear conjunction with each other, but an exact-phrase search for `'United States and China'` would miss it.



Sentence	Country	Year
Mr. Bush outlined a reasonable approach after the release of the American crew members when he said the United States and China "have different values, yet common interests" and that both nations "must make a determined choice to have productive relations."	China	2001
The United States and China need not become enemies, though historical and political forces could push them in that direction.	China	2001
At the least, appreciating how deeply the Chinese feel on questions of pride has obvious implications for how we deal with the principal issue that divides the United States and China, namely Taiwan.	China	2001
Beijing's unyielding position suggests that it is unwilling or unable to put this episode into perspective and look at the long-term consequences for relations between the United States and China.	China	2001
The clashes between the United States and China in the last decade (trade, Taiwan, the embassy bombing, even the Olympics) are used as evidence of American arrogance.	China	2003
The United States, the world's top energy guzzler, and China, with the world's fastest-growing energy thirst, both need to make adjustments so that	China	2006

Frequent Phrases	
United States	144
States China	98
between United	44
Russia China	22
South Korea	18

Frequen...	
states	142
united	142
china	142
russia	41
relation	24
korea	22
north	21

Frequen...	
include	9
make	8
start	8
need	6
involve	5
develop	5
resolve	5

Frequen...	
nuclear	17
other	16
american	9
military	9
new	8
recent	6
economic	6

Figure 6.7: A *List of Search Results* view of the `conjunction(United States, China)` sentences.

He obtained a list of search results containing a total of 142 sentences (Figure 6.7). A quick look at the distribution of articles over the Year category (circled in red) confirmed that 1994 was the year in which frequent mentions began. These were much more frequent from the mid-nineties onward. In fact, out of the 142 occurrences, 116 (81%) occurred after 1994.

He began skimming over the results, but soon noticed a problem. While these these sentences did indeed contain many references to U.S.-China relations, not all of them were relevant to him. For example, there were sentences like, "While maintaining cautious vigilance against rival powers, the United States, Soviet Union, and China separately welcomed this trend.", and sentences involving other parties, like, "In turn, the nuclear five – the United States, Britain, Russia, France and China – committed themselves to sign a comprehensive test ban by next year."

WordSeer's sentence sets are designed to allow the user to create their own semantic categories by grouping semantically similar sentences together according to the user's own definition of meaning. C.F. recognized this, and put them to use. He manually inspected each of the 134 results, and created a sentence set out of the ones that specifically dealt with U.S.-China relations. He called this set `{US-China relations}`. Sentences were sometimes ambiguous, but the ease of looking up the sentence within the editorial made it easy to check whether or not it should be included. To quote:

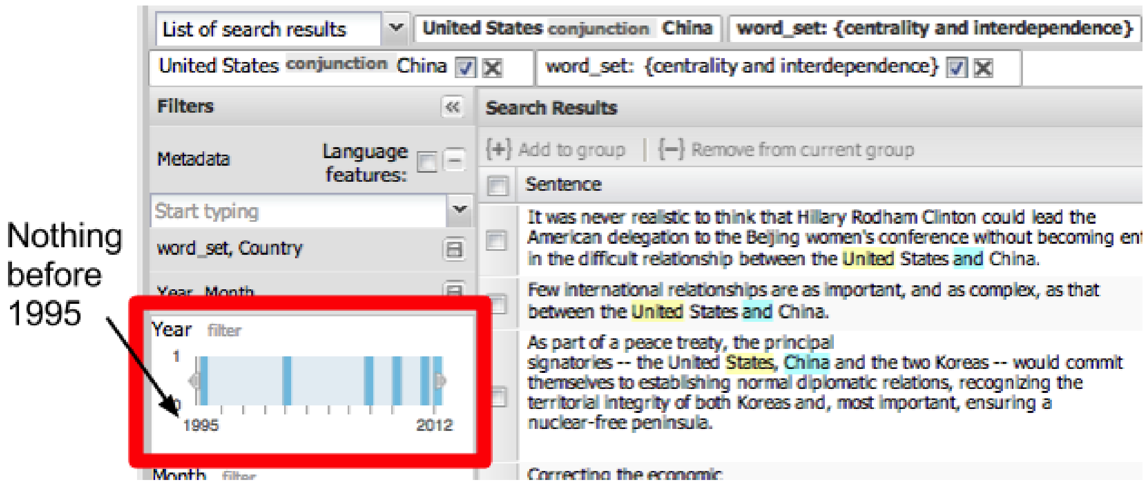
> [The filtering process] was significantly aided by the ability to click on the sentence and immediately see it located in context within the full editorial.

He eliminated about half the sentences from consideration, leaving him with a set of 79 sentences. A Word Frequencies view of this smaller slice made the picture much clearer. Of the 79 sentences, the overwhelming majority (86%) occurred after 1994.

C.F. noticed a subtle point: as he moved from the 1990's to the 2000's, the relationship between the U.S. and China seemed to be increasingly 'central' and 'inter-dependent'. To quote:

> Phrases and words like the following begin to appear: "21st century's most important relationship," "co-dependent," "interlocking," etc. But I achieved a more precise sense of the themes of *interdependence* and *centrality* more precisely using a Word Set of thematically-related adjectives drawn from the Frequent Adjectives overview.

C.F. composed a Word Set of words from the overview, consisting of: "most, central, important, indispensable, dependent, interdependent, dependency, entangled, interlocking, interdependency". Figure 6.8 shows the distribution of centrality and interdependence adjectives detected automatically by the tool, intersected with the conjoined U.S.-China relationship sentences curated by C.F. The year-overview for the filtered set immediately shows that it is not until 1995 that the U.S.-China relationship is characterized by these themes.



It was never realistic to think that Hillary Rodham Clinton could lead the American delegation to the Beijing women's conference without becoming en
in the difficult relationship between the United States and China.

Few international relationships are as important, and as complex, as that between the United States and China.

As part of a peace treaty, the principal signatories -- the United States, China and the two Koreas -- would commit themselves to establishing normal diplomatic relations, recognizing the territorial integrity of both Koreas and, most important, ensuring a nuclear-free peninsula.

Correcting the economic

Figure 6.8: The distribution of the `{centrality and interdependence}` Word Set within the `conjunction(United States, China)` slice. There are no occurrences before 1995.

6.1.4 Discussion

Prior to using WordSeer, C.F. had general ideas about the points he wanted to make based on years of reading, but had no way to quantify or codify those ideas based on data. With WordSeer, he found changes in language use that reflected historians' understanding of the rise of China (its transformation from a 'card' played by American diplomats during the cold war, to its present-day status as a 'central' and 'entangled' partner of the United States), but with more precision (the steady increase in growth-related verbs over the last 30 years, the sudden jump in phrases like "US and China" after 1994) and with more nuance (the emergence of words relating to the 'centrality' and 'interdependence' of their relationship in the 2000s).

6.2 Literacy Autobiographies

6.2.1 Introduction

As a college English Composition instructor, Rex Ganding (R.G.) a PhD student at UC Berkeley's School of Education asks students to write often and in a variety of situations. He examined ways in which WordSeer could assist him in the teaching and learning of writing. He was particularly interested in the literacy autobiography, in which students describe significant experiences they have had with literacy and reflect upon the importance of literacy to their lives.

R.G. analyzed the content of approximately 140 literacy autobiographies written by students from courses that he has taught over the past two years, each about 1,500 to 2,000 words long. He is familiar with the collection, having previously read and commented on each

Course	College	Level
Engl 1A (n=29)	[anonymized]	First-year
Engl 201AB (n=28)	[anonymized]	Pre-transfer
Engl 5 (n=40)	[anonymized]	Transfer level
Edu 140 (n=40)	[anonymized]	Upper division

Table 6.1: The four different courses, each at a different proficiency level, from which literacy autobiographies were analyzed by R.G.

of the essays. Table 6.1 shows the different courses from which literacy autobiographies were analyzed. The courses were at different colleges and taught at different proficiency levels.

Among the questions that guided his inquiry were the following. In this example, we show how the tool helped him get answers:

1. What can a distant reading of student literacy autobiographies tell him about students that close readings cannot?

2. What patterns exist in student literacy autobiographies at different course levels and institutions?

Here, *distant reading* is a term coined by Moretti (2005), for algorithmic and visualization-based approches to literary analysis. He compared it with the traditional activity of *close reading*, in which a scholar analyzes texts by reading them carefully and interpreting them.

6.2.2 A New Take on Students' Experiences

R.G. does not usually consider the frequencies of words unless a student repeats one to the point of distraction. However, the WordSeer's very first overview (automatically generated for the whole collection on startup) prompted him to consider the significance of individual words and their repeated use. As he states:

> From the moment I opened WordSeer, 'distance' immediately provided new insight and areas of exploration as I was intrigued by unexpected high word frequencies. The most obvious example is the frequent use of the word `time` which is not only the most frequent noun but also the most frequent word overall. As opposed to `literacy`, `language`, and any noun or verb forms of `read` and `write`, the Word Tree for `time` appears before any term is placed in the keyword search. I found similar surprises in the adjective and verb frequencies, and these surprises would guide my decision-making and analysis.

Faced with unexpected discoveries in every part of speech, R.G. decided to explore the adjectives. The top three adjectives were not surprising - `other` (474),`first` (379), and `new` (384). However, `able` (314) and `own` (282) were words that he did not expect to be used frequently.

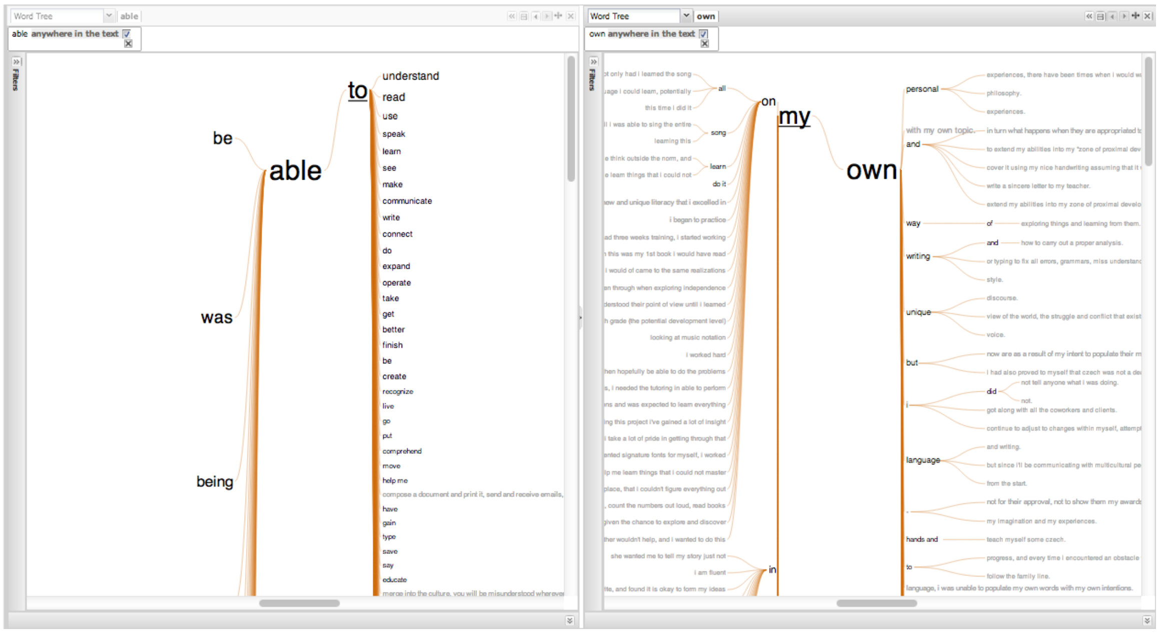


Figure 6.9: Side-by-side Word Trees (with overview panels collapsed) for `able` and `own`, created from the Word Menu.

R.G. now wanted to understand the unexpectedly high frequencies of `able` and `own`. WordSeer made this exploration easier. Instead of having to open a new window and type new search queries, clicking on the words themselves opened up Word Menus which allowed him to create Word Trees centered on those words.

The Word Tree for `able` (Figure 6.9 left) revealed that the most common use of the adjective able was: `[form of the verb 'to be'] + able + to [action verb]`. To get a sense of how common this usage was overall, R.G. hovered over each branch (which displayed the number of sentences under that branch), and adding up the numbers for branches matching `to be`. He found that the overwhelming majority, 280 out of 314 occurrences, fell into this pattern, with the most common `abilities` being literacy-related: read, understand, communicate, learn, speak, etc. For `own` (Figure 6.9 right), the most common construction was: `[preposition] + my + own`, with around a third: 100 out of 291 occurrences. He also used the Word Tree to zoom in and read the individual sentences.

As a result, R.G. came away with a new insight about his students:

> Student writers articulate their experiences from some first encounter with the unfamiliar, followed by a process of being `able` to act within that which is becoming less unfamiliar. Moving into literacy requires these writers to develop their `own` abilities as necessary to survive and prosper in that context.

6.2.3 Writing Proficiency Across Courses

The different courses R.G. taught were taken by students with differing amounts of college education. He was especially interested in differences in sentence construction, hypothesizing that more proficient students would use advanced structures more frequently. To initiate the analyses, he performed word searches on terms such as `though`, `while`, `although`, and `however`, which indicate a complex structure called a `concessive`. He was expecting concessives to be increasingly frequent as student experience increased from English 1A, to English 201, to English 5, to Ed140.

To follow up R.G. created a Word Frequencies visualization of this terms. This visualization shows how often terms occur across different metadata categories (and can show the counts stacked or grouped, normalized or raw). R.G. added all the concessive terms to the same view, which created a comparative visualization (Figure 6.10). The increasing frequencies across the

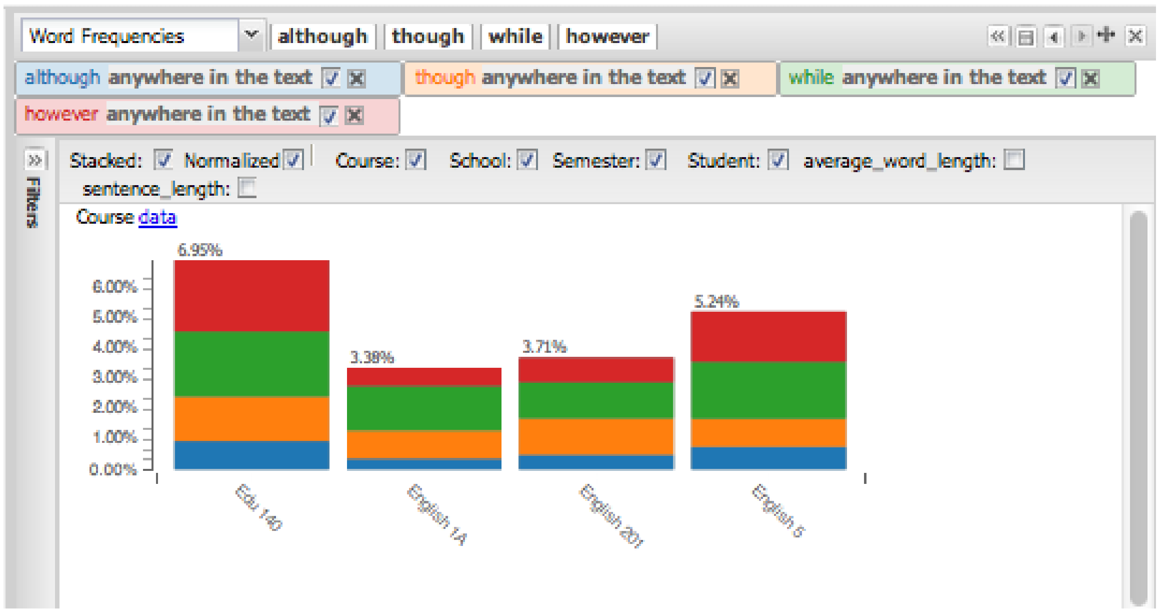

Word Frequencies
although
though
while
however
although anywhere in the text
though anywhere in the text
while anywhere in the text
however anywhere in the text
Filters
Stacked: ✓ Normalized ✓ Course: ✓ School: ✓ Semester: ✓ Student: ✓ average_word_length: ☐
sentence_length: ☐
Course data
6.95%
3.38%
3.71%
5.24%
6.00%
5.00%
4.00%
3.00%
2.00%
1.00%
0.00%
Edu 140
English 1A
English 201
English 5

Figure 6.10: The frequencies of different concessive words across the Course category: 'although' (blue), 'though' (orange) 'while' (green) and 'however' (red).

course levels confirmed his hunch that as students become more experienced and comfortable with the written word, they used more complex sentence structures more frequently.

6.3 Belief: From Religious to Secular in Early American Fiction

This case study was done by Matthew Ramirez, a PhD student in UC Berkeley's English department, along with undergraduates Kathleen Miller and Joyce Liu. They had been working with Professor Donald McQuade of UC Berkeley's History department, who is interested in 19th Century visual culture, especially advertisements for patent medicine, and how this reflected a shift in how print culture shifted from religious discourse to more secular topics over the course of the 19th Century. Although there is already a great deal of historical work on this shift, the students wished to examine how this trend manifested itself in the ways that particular terms relating to belief were used.

Their hypothesis was that words indicating belief, such as the words 'belief', 'faith', 'confidence', and 'trust', occurred mainly in the context of *religious* belief in the early 19th Century, but gradually gained secular meanings, particularly relating to medicine. To this end, they used WordSeer to analyze a sample of early American fiction ranging from 1789 to 1875. Their results suggest the presence of this shift. The words 'faith' and 'trust' occurred exclusively in religious contexts before the 1820s, but began to be used in medical contexts at a steady rate staring in the 1830s and after.

6.3.1 Texts

They accessed the texts, digitized by the University of Virginia Library, through ProQuest (`www.proquest.com/products_pq/descriptions/early_am_fiction.shtmlâĂİ`. The collection was chosen because it states in its editorial policy that the Early American Fiction Collection situates works of canonical writers of the time alongside novels and short stories that were popular during that time period but are now largely forgotten and inaccessible in print form. They considered this to be a fairly random sample of texts, as the literary genres are mixed.

Their collection consisted of of 463 documents from 52 authors, some well-known, such as James Fenimore Cooper and Edgar Allan Poe, and others more obscure, like Rufus Dawes.

To obtain the texts, they used Python and Beautiful Soup to scrape the texts from the Chadwyck site and also to remove the HTML markup. They gave the texts to us, and we processed them into XML format, and uploaded them into WordSeer.

6.3.2 Word Relationships

The terms of belief the students chose to investigate were the following:

belief, faith, trust, confidence, and miracle

To avoid having to process the entire collection, we filtered the texts and extracted only the sentences containing the words relating to belief and their variants (such as `believe`, and `beliefs` as well as just `belief`).

The underlying collection was therefore composed only of sentences containing the belief-related word-stems `belie*`, `confid*`, `faith`, `mirac*`, `trust`. When we processed the texts they gave us, we treated these as metadata categories. For example, the collection could be filtered to `term = belie` to restrict it to just the sentences in which a word with the stem `belie*` occurred.

The word `miracle` was something of a test. It is a non-belief-related term that occurred in many contexts, religious and secular. Their reason for including it (as described in an unpublished report, quoted here with permission) was that:

> If the co-occurrences between miracle and religious and secular terms is similar to the terms of belief, it may be worthwhile to conduct further work on co-occurrences of non-terms of belief within religious and secular contexts over the course of the nineteenth century.

6.3.3 Decision to Use WordSeer

Prior to using WordSeer they experimented with Jigsaw Stasko *et al.* (2008), another platform that provides sensemaking support. However, they decided to use WordSeer instead:

> However, as Jigsaw works best with shorter documents, we found it unsuitable for our corpus. Additionally, Jigsaw focuses on entities (names, places, dates), which also was not a central concern in our project. While Jigsaw has impressive visualizations, we found that WordSeer allows for more precision in terms of locating contexts in which our target words occur and allows us to interact more with our corpus. WordSeer also provides rich exploratory tools through its numerous grammatical search options.

6.3.4 Collecting Religious and Secular Terms

Their first task was to refine their conceptions of 'religion contexts' and 'medical contexts'. They did this by collecting groups of religious and secular medical terms. There were many words that would have fit in any of these categories, but they had to discover them and select only those that were strongly associated with one or the other.

In this phase, they started with terms like 'god' and 'holy' and examined similar-context and co-occurring words. One exploratory exercise they did was to create a sentence set of all the instances of 'belief' that were in verb form, to see the different things that were being "believed in". They explored this in several ways.

The grammatical search for the `prep_in(believe, ____)` relation gave a list of the things that were "believed in" (Figure 6.11).

Another exploratory exercise was to examine the Word Tree for 'believe' (Figure 6.12).

The overviews of frequent phrases also came in handy. Among the words most frequently used with the phrase `believe in` were words like `God`, `existence`, `witchcraft`, `Providence` (which upon investigation turned out to be divine providence) and `Soul`.

At each stage, they collected more words related to religion, ending up with the final list:

> God, holy, sacred, redemption, prayer, church, soul, heaven, lord, religion, priest, and sin

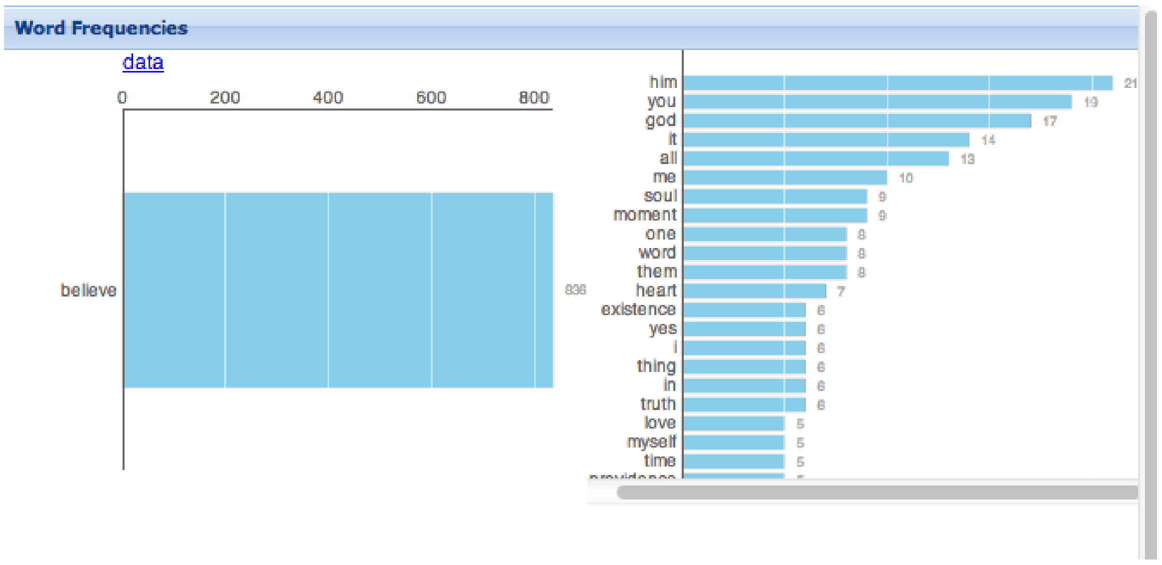


Figure 6.11: Grammatical search results for `prep_in(believe, ____)`.

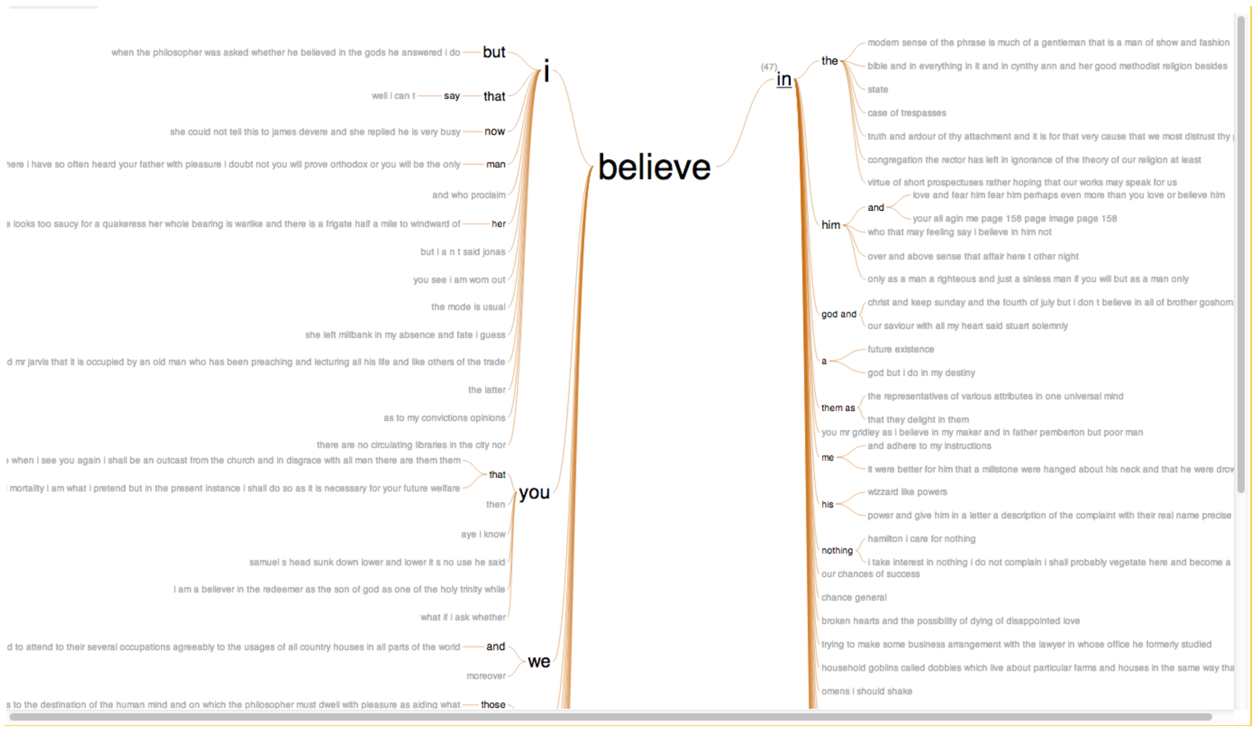

believe
i
but
that
say
now
man
her
you
we
and
those
(47)
in
the
him
and
god and
a
them as
me
his
nothing
when the philosopher was asked whether he believed in the gods he answered i do
well i can t
she could not tell this to james devere and she replied he is very busy
here i have so often heard your father with pleasure i doubt not you will prove orthodox or you will be the only
and who proclaim
looks too saucy for a quakeress her whole bearing is warlike and there is a frigate half a mile to windward of
but i a n t said jonas
you see i am worn out
the mode is usual
she left millbank in my absence and fate i guess
d mr jarvis that it is occupied by an old man who has been preaching and lecturing all his life and like others of the trade
the latter
as to my convictions opinions
there are no circulating libraries in the city nor
when i see you again i shall be an outcast from the church and in disgrace with all men there are them them
mortality i am what i pretend but in the present instance i shall do so as it is necessary for your future welfare
then
aye i know
samuel s head sunk down lower and lower it s no use he said
i am a believer in the redeemer as the son of god as one of the holy trinity while
what if i ask whether
d to attend to their several occupations agreeably to the usages of all country houses in all parts of the world
moreover
s to the destination of the human mind and on which the philosopher must dwell with pleasure as aiding what
modern sense of the phrase is much of a gentleman that is a man of show and fashion
bible and in everything in it and in cynthy ann and her good methodist religion besides
state
case of trespasses
truth and ardour of thy attachment and it is for that very cause that we most distrust thy
congregation the rector has left in ignorance of the theory of our religion at least
virtue of short prospectuses rather hoping that our works may speak for us
love and fear him fear him perhaps even more than you love or believe him
your all agin me page 158 page image page 158
who that may feeling say i believe in him not
over and above sense that affair here t other night
only as a man a righteous and just a sinless man if you will but as a man only
christ and keep sunday and the fourth of july but i don t believe in all of brother goshom
our saviour with all my heart said stuart solemnly
future existence
god but i do in my destiny
the representatives of various attributes in one universal mind
that they delight in them
you mr gridley as i believe in my maker and in father pemberton but poor man
and adhere to my instructions
it were better for him that a millstone were hanged about his neck and that he were drov
wizzard like powers
power and give him in a letter a description of the complaint with their real name precise
hamilton i care for nothing
i take interest in nothing i do not complain i shall probably vegetate here and become a
our chances of success
chance general
broken hearts and the possibility of dying of disappointed love
trying to make some business arrangement with the lawyer in whose office he formerly studied
household goblins called dobbies which live about particular farms and houses in the same way tha
omens i should shake

Figure 6.12: Word Tree for `believe`, for the branch `believe in`.

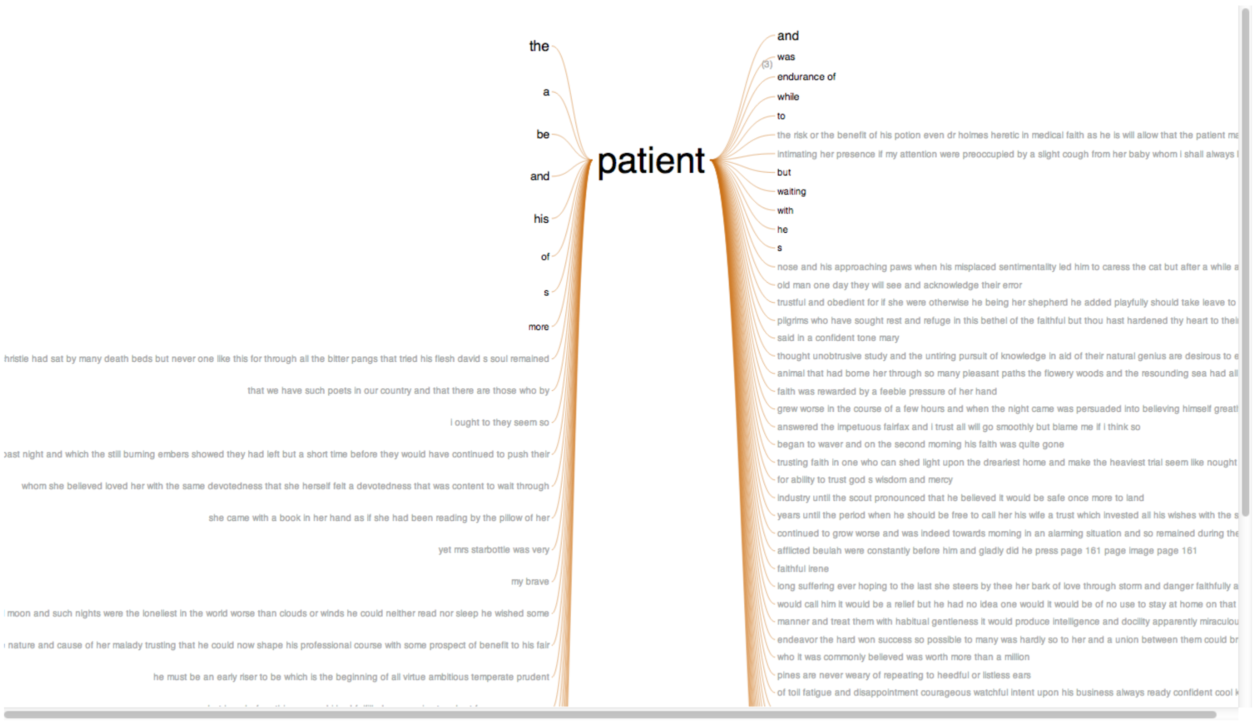

the
a
be
and
his
of
s
more
hristie had sat by many death beds but never one like this for through all the bitter pangs that tried his flesh david s soul remained
that we have such poets in our country and that there are those who by
i ought to they seem so
past night and which the still burning embers showed they had left but a short time before they would have continued to push their
whom she believed loved her with the same devotedness that she herself felt a devotedness that was content to wait through
she came with a book in her hand as if she had been reading by the pillow of her
yet mrs starbottle was very
my brave
moon and such nights were the loneliest in the world worse than clouds or winds he could neither read nor sleep he wished some
nature and cause of her malady trusting that he could now shape his professional course with some prospect of benefit to his fair
he must be an early riser to be which is the beginning of all virtue ambitious temperate prudent
patient
and
was
(3)
endurance of
while
to
the risk or the benefit of his potion even dr holmes heretic in medical faith as he is will allow that the patient ma
intimating her presence if my attention were preoccupied by a slight cough from her baby whom i shall always
but
waiting
with
he
s
nose and his approaching paws when his misplaced sentimentality led him to caress the cat but after a while a
old man one day they will see and acknowledge their error
trustful and obedient for if she were otherwise he being her shepherd he added playfully should take leave to
pilgrims who have sought rest and refuge in this bethel of the faithful but thou hast hardened thy heart to thei
said in a confident tone mary
thought unobtrusive study and the untiring pursuit of knowledge in aid of their natural genius are desirous to e
animal that had borne her through so many pleasant paths the flowery woods and the resounding sea had all
faith was rewarded by a feeble pressure of her hand
grew worse in the course of a few hours and when the night came was persuaded into believing himself great
answered the impetuous fairfax and i trust all will go smoothly but blame me if i think so
began to waver and on the second morning his faith was quite gone
trusting faith in one who can shed light upon the dreariest home and make the heaviest trial seem like nought
for ability to trust god s wisdom and mercy
industry until the scout pronounced that he believed it would be safe once more to land
years until the period when he should be free to call her his wife a trust which invested all his wishes with the s
continued to grow worse and was indeed towards morning in an alarming situation and so remained during the
afflicted beulah were constantly before him and gladly did he press page 161 page image page 161
faithful irene
long suffering ever hoping to the last she steers by thee her bark of love through storm and danger faithfully a
would call him it would be a relief but he had no idea one would it would be of no use to stay at home on that
manner and treat them with habitual gentleness it would produce intelligence and docility apparently miraculou
endeavor the hard won success so possible to many was hardly so to her and a union between them could br
who it was commonly believed was worth more than a million
pines are never weary of repeating to heedful or listless ears
of toil fatigue and disappointment courageous watchful intent upon his business always ready confident cool

Figure 6.13: Word Tree for the word `patient` showing its usage as a virtue, and not just as the person treated by a doctor.

A sample of sentences containing each of these words was examined, and these words were chosen because their usage was almost exclusively religious in meaning.

A similar process was followed for the medical terms. This time, starting with the word `medicine`, and examining frequently co-occurring words, which included `cure`, `patient`, and `doctor`, and exploring the chain of co-occurring words from there on. They chose the words:

> medicine, balm, pill, herb, doctor, dose, tonic and cure

The term 'patient' would have been a good candidate, but was often used in the sense of endurance or patience, as shown by the Word Tree (Figure 6.13).

6.3.5 Results: Faith and Trust Become More Secular

After the religious and secular word sets had been created, the results were straightforward to obtain by comparing time trends for different words over time. The Word Frequencies graph for both `{religious terms}` and secular `{medical terms}` is relateively flat over time, although religious terms are overwhelmingly more popular (Figure 6.14).

Filtering the visualization to just the belief-words `faith` (Figure 6.15) and `trust` (Figure 6.16), however, suggested that although these words occured in purely religious contexts before the 1920s, they increasingly started to appear in medical contexts after that. In particular, the graph for `trust` showed that in the late 1820s and the late early 1860s, trust actually appears *more often* in medical contexts than religious contexts. The complete absence of `faith` and `trust` in medical contexts before the 1820s made (Figure 6.17) this shift even more interesting.

6.4 Discussion

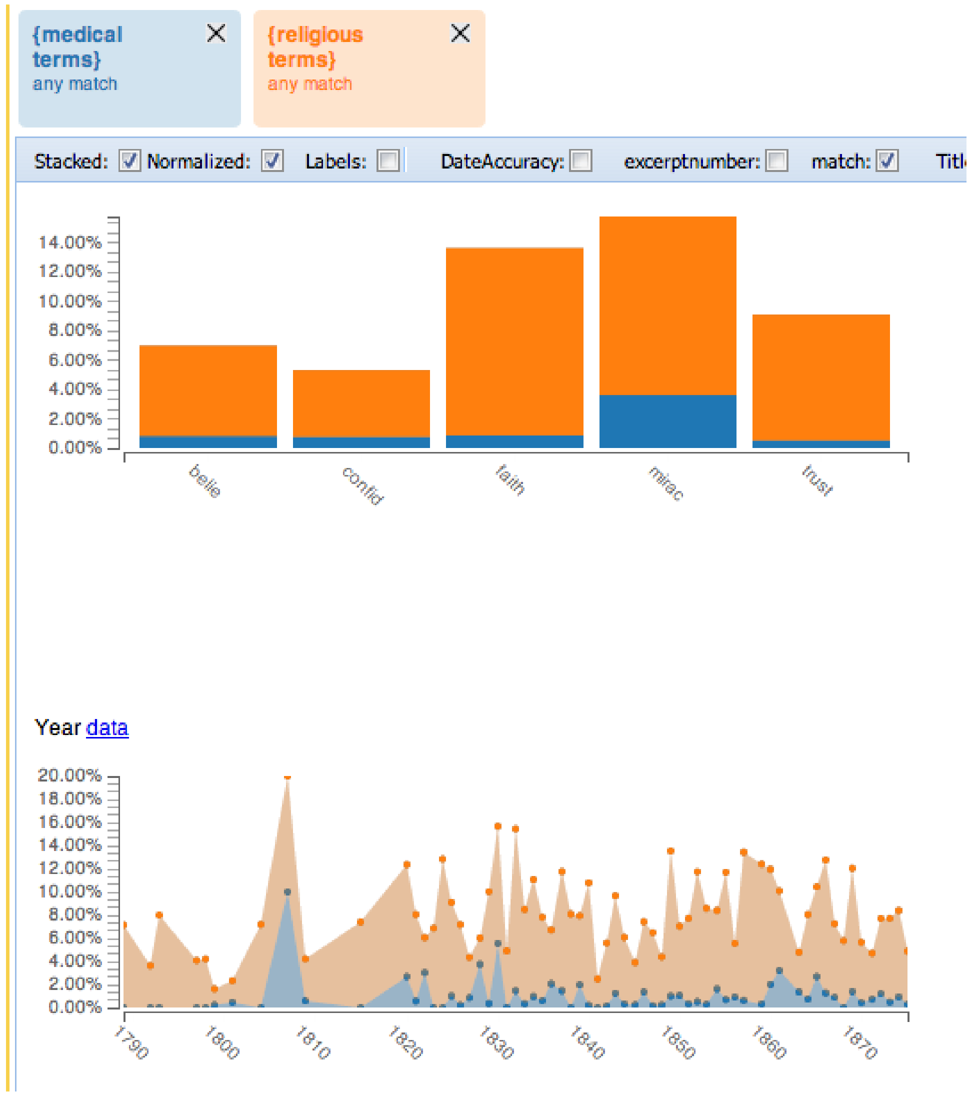
These case studies show how WordSeer helped scholars make new discoveries about the world and create new understanding from text collections; these are discoveries that most likely would not have been able to make with any other existing tool.

Prior to using WordSeer, C.F. had general ideas about the points he wanted to make based on years of reading, but had no way to quantify or codify those ideas based on data. In just a few weeks, WordSeer allowed him to find changes in use in language that reflected historians' understanding of the rise of China (its transformation from a 'card' played by American diplomats during the cold war, to its present-day status as a 'central' and 'entangled' partner of the United States), but with more precision (the steady increase in growth-related verbs over the last 30 years, the sudden jump in phrases like "US and China" after 1994) and with more nuance (the emergence of words relating to the 'centrality' and 'interdependence' of their relationship in the 2000s).

In the second case study, R.G. had spent hours reading and thinking about the literacy essays, but had not looked at them in terms of word frequency or syntactic structure. Using WordSeer, he was able to see the documents in a new light and form new understanding of the general trends in his students' process of learning to write. He was also able to formulate and find evidence for a hypothesis about an increase in proficiency with advanced writing structures as they moved to successively advanced educational levels. He used a complex combination of searching for grammatical structures, side-by-side comparisons of sets of sentences according to adjectives and their use in context, selecting a subset of words and comparing their frequency across a metadata classification.

In the third case study, WordSeer enabled the scholars to iteratively refine their sets of medical and religious terms. They could explore individual words, such as `patient`, and examine the degree to which they were used in the desired contexts, down to the level of individual sentences. The Related Words menus also allowed them to discover other words to include. Discovering related words in this way would not have been easy by traditional reading (their collection had hundreds of novels) or without implementing an algorithm, which would have required programming knowledge.

A key component of the use of the tool in these examples is the freedom it affords the user to pivot on words, on words' relations to other words, to create groups of words and cross them with arbitrary metadata categories, and to be able to immediately view the context of their original sentences and documents, thus allowing "close reading" as part of the text analysis process.


{medical terms} any match

{religious terms} any match

Stacked: ✓ Normalized: ✓ Labels: ☐ DateAccuracy: ☐ excerptnumber: ☐ match: ✓ Titl

Year data

Figure 6.14: The Word Frequencies graph of the religious and medical terms. The underlying collection was composed of sentences containing the following word-stems related to belief: `belie*`, `confid*`, `faith`, `mirac*`, `trust`. These stems appear as categories in the top graph. For each stem, the blue bars show the percentage of sentences containing that stem that also contain `{medical terms}`, and the orange show the percentage of those sentences that contain `{religious terms}`. The bottom graph shows a time series visualization of the percentages of all the sentences in that year that contained `{medical terms}` (blue) and `{religious terms}` set (orange). For both sets, the graphs are relatively flat over time, although `{religious terms}` are overwhelmingly more popular.

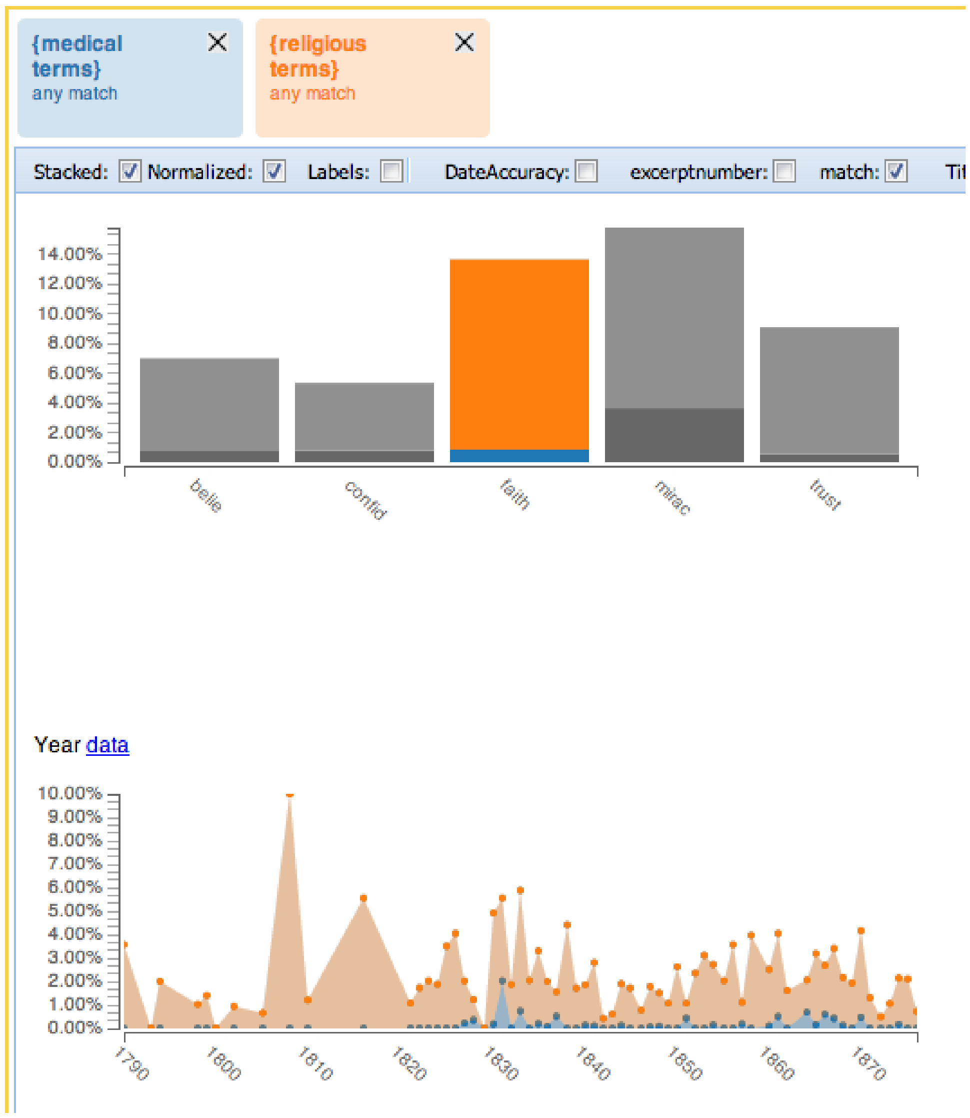


Figure 6.15: The visualization filtered to just the sentences for the belief-word `faith`, by clicking the top chart's bars for `faith`. The time series now shows the trends for `{medical terms}` (blue) and `{religious terms}` (orange) in just the sentences with `faith` over time. The `{medical terms}` do not appear in the `faith` sentences before the 1820s, but appear at a small, steady rate after that.

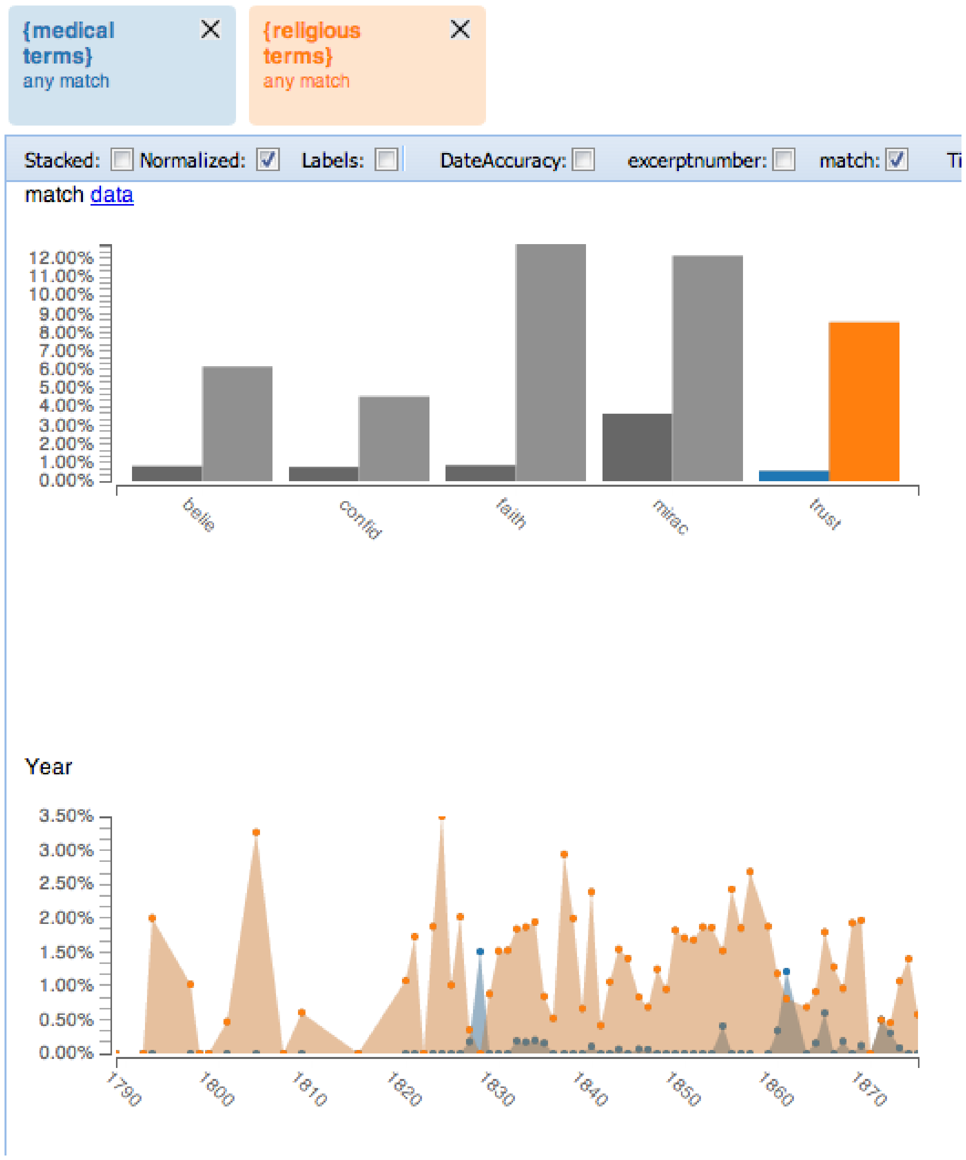


Figure 6.16: The visualization filtered to just the sentences for the belief-word `trust`, by clicking the top chart's bars for `trust`. The time series now shows the trends for `{medical terms}` (blue) and `{religious terms}` (orange) in just the sentences with `trust` over time. The `{medical terms}` do not appear in the `trust` sentences before the 1820s, but appear at a small, steady rate after that, including two years during which the appear more frequently than `{religious terms}`.

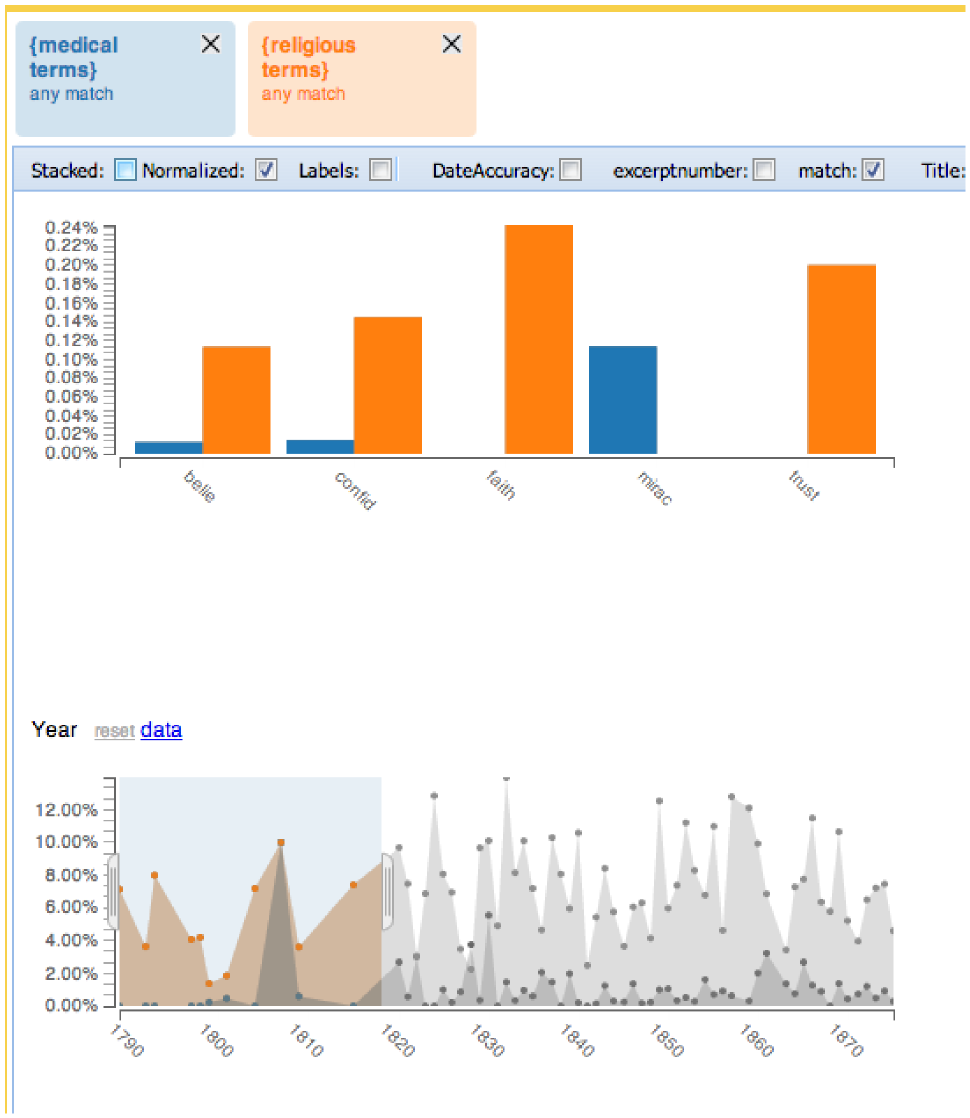


Figure 6.17: Filtering the time-series graph visualization alters the top bar chart to show only data from filtered time period. The top chart has bars for each of the different belief-related stems. For each stem, the blue bars show the percentage of sentences that also contain `{medical terms}`, and the orange bars show the percentage that contain `{religious terms}`. However, when the data set is filtered to the pre-1820s, there are no blue bars for `faith` and `trust`, indicating an absence of `{medical terms}` sentences containing those words.

6.5 New Problems

The three case studies described here were a third round of formative evaluation for WordSeer. They revealed two more text analysis tasks that were not well supported. In the next chapter, we describe the investigations we conducted into possible solutions.

C.F. and R.G. both wanted to search for sentence structures beyond word-to-word relationships. C.F. was interested in sentences with a comparative structure, specifically comparisons between Chinese and Japanese things. As a composition instructor, R.G. was interested in applications of a “sentence-structure search” to teaching. If there was a way to summarize students’ most frequent sentence-structure mistakes, or to find overall commonalities in the ways students structured their sentences, instructors could focus on helping the students improve in those areas. In Chapter 7.2, we describe experiments we conducted on making syntactic search more useful by improving the presentation and recognizability of syntactic relationships between words.

C.F. and R.G. also wanted to automatically create groups based on a set of examples, and to fine-tune the results by giving feedback. For concepts that did not map onto keyword or grammatical searches (such as C.F. analysis of the changing tone of U.S.-China relations), they found it laborious to hand-pick individual sentences for Sentence Sets. As a possible approach to this problem, we investigated relevance feedback (Salton and Buckley 1997) a technique for a gathering a relevant set of search results by using users’ feedback on previously-returned items. These experiments are described in Chapter 7.1.

The case studies also made the need for improved speed and responsiveness clear.Although WordSeer provides results in under a second in some cases, in other cases the wait is longer, depending on the size of the collection and the operation requested. As a result, the visual design and back-end implementation were optimized for speed, leading to the system described in the previous chapter.

Chapter 7

Empirical Investigations

This chapter describes two experiments we conducted on specific text analysis features of the WordSeer system. In previously-described case studies (Chapters 3 and 6), we evaluated the system end to end to seek qualitative feedback to guide future development. Here, we used controlled experiments to test specific user interface changes and algorithms in isolation.

The questions we investigated were motivated by two text analysis problems identified in the previous chapter: collecting a group of thematically-related sentences, and searching over the syntactic structures of sentences. The following two sections describe our experiments.

7.1 Finding Literary Themes with Relevance Feedback

7.1.1 Introduction

Conceptually-linked passages of text are central to text analysis. For example, journalists and intelligence analysts might seek quotes, excerpts, and mentions of events. Legal scholars may want to find evidence of particular treatments given to issues. In the humanities, literary scholars may search for examples of themes, imagery, particular kinds of word usage, occurrences of tropes or stereotypes, and other patterns of language use (Don *et al.* 2007).

In our own Case Study 1 (Chapter 3.1), one problem we faced was that some stereotypes were hard to characterize in terms of grammatical relationships. For example, one of the stereotypical conventions was a description of a white father. However, this was usually conveyed over multiple lines of text, and interspersed with other events from the narrator's childhood. It was rarely the case that the adjective "white" was directly used to describe "father", but it was often understood that the father was white because, for example, he was also the slave owner.

In current information retrieval systems, such passages of text are retrieved through keyword search. In systems such as Google Books, the searcher types in a query, and receives a ranked list of matching excerpts from various books.

For passages with a common conceptual link, generating representative keyword queries is problematic. The reason is familiar: the vocabulary problem (Furnas *et al.* 1987). This refers to the phenomenon that the same concept can often be expressed with many different words, and that different people are unlikely to choose the same way of describing the same thing. For example, take the Shakespearean theme that seeing an event first-hand is more credible than hearing about it from another person (discussed by Robert B. Heilman in (Heilman 1956) pp.58–64). This theme surfaces in two very different sentences, which have no words in common:

> I would not take this from report; it is,
> And my heart breaks at it.
> (King Lear Act 4, Scene 6 Lines 136–137)
>
> Then have you lost a sight, which was to be seen, cannot be spoken of.
> (The Winter's Tale, Act 5 Scene 2 Line 47)

Thus, relying on a set of search terms generated by a single person could lead to missing examples and non-representative results. This is a problem if the scholars seeking these passages are doing so as part of a sensemaking process (Pirolli and Card 2005). Incomplete or non-representative examples compromise the integrity of arguments made using them.

A better approach would rely less on the query formulation process. Relevance feedback (Rocchio 1971; Salton and Buckley 1997) offers such an approach. In a relevance feedback system, the searcher can give the system feedback on the relevance of the results. The system uses the feedback to adapt its model of the user's information need, and return results that are "more like" the ones marked relevant. So, instead of having to formulate and re-formulate queries in search of relevant results, the searcher has the option to mark relevant and irrelevant ones, leaving the re-formulation to the system. This approach takes advantage of *recognition over recall*. This is the psychological finding that it is usually easier for a person to recognize something by looking for it than it is to think up how to describe that thing (Hearst 2009a).

In the case study on U.S.-China relations (Chapter 6.1), C.F. could have used this approach to identify sentences describing the complexities of the US-China relationship, and R.G. could have used it to automatically catalog the different senses of the word "get": becoming ('I got ready to . . . '), and acquisition ('I got a lot of practice . . . '). Instead of having to manually construct them by searching, reading, and individually selecting sentences.

Our goal was to investigate whether relevance feedback is an effective solution to this problem. Due to our previous collaboration on Shakespeare's works with Professor Michael Ullyot from Case Study 2 (Chapter 3.3), and his interest in this approach, we focused on finding examples of literary themes. Specifically, our question was whether searchers equipped with relevance feedback are more effective at finding examples of themes than searchers without.

We implemented a simple relevance feedback system for sentence-length text and compared it with a keyword-search-only system in a user study. In the study, participants were asked to find examples of a pre-selected theme by searching over the complete works of Shakespeare, and given five minutes in which to do so. The theme was described in words, and the participants were also given two example sentences that illustrated the theme. Participants used either the relevance feedback system or the keyword-search-only system to complete the task.

After all of the participants submitted their responses, the sentences they chose were rated as "relevant" or "not relevant" by Professor Ullyot. We then compared participants' scores across the two interfaces to determine whether relevance feedback produced improvements over keyword search.

7.1.2 System Description

We implemented the Rocchio algorithm for relevance feedback (Rocchio 1971), as described by Salton and Buckley in (Salton and Buckley 1997).The only difference was that our "documents" were in fact sentences. Our system used the vector space model of information retrieval. Each search query was translated into a feature vector Q, and sentences retrieved by ranking their feature vectors in order of their inner products with the query vector.

We used a bag of words feature vector to represent sentences. A sentence s would be translated into feature-space vector S:

$$S = (w_1, w_2, \ldots, w_n), \tag{7.1}$$

where w_i is the weight of word i in the sentence. Word weights were computed using tf-idf (Salton and Buckley 1988) with the "documents" being sentences:

$$w_i = TF(i, s) \times IDF(i), \tag{7.2}$$

where

$$TF(i, s) = \frac{\#\text{ times word } i \text{ appears in } s}{\#\text{ words in } s} \tag{7.3}$$

and

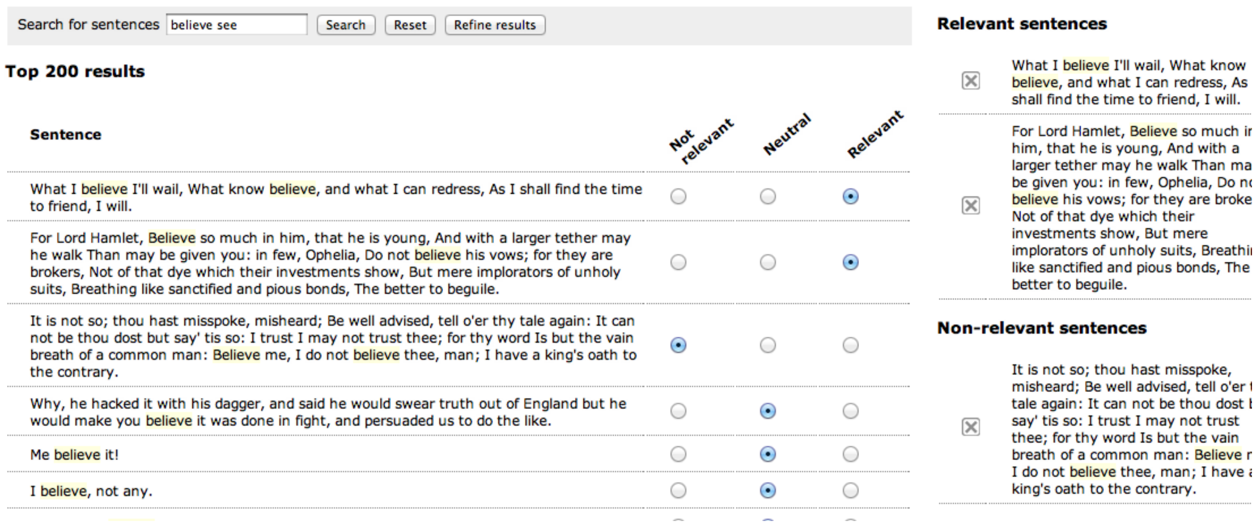
$$IDF(i) = \log\left(\frac{\text{total }\#\text{ sentences}}{\#\text{ sentences in which word } i \text{ appears}}\right). \tag{7.4}$$

Since we had access to part-of-speech tagged text, we were able distinguished between some words that shared the same lexical form. For example, compare the sentences, "I lost my rings." and "The bell rings." Because of part of speech tagging, the feature corresponding to "rings" in the first sentence (a noun) would be different in our system from the feature corresponding to "rings" in the second sentence (a verb).

The query vector Q was:

$$Q = (q_1, q_2, \ldots, q_n) \tag{7.5}$$

Where $q_i = 1/(\#\text{ non-zero features})$ if word i was in the query, or 0 if word i was not in the query. Since the query was not part-of-speech tagged, the vector contained equal weights for all part-of-speech variants of the query words.



Top 200 results

Sentence	Not relevant	Neutral	Relevant
What I believe I'll wail, What know believe, and what I can redress, As I shall find the time to friend, I will.	○	○	●
For Lord Hamlet, Believe so much in him, that he is young, And with a larger tether may he walk Than may be given you: in few, Ophelia, Do not believe his vows; for they are brokers, Not of that dye which their investments show, But mere implorators of unholy suits, Breathing like sanctified and pious bonds, The better to beguile.	○	○	●
It is not so; thou hast misspoke, misheard; Be well advised, tell o'er thy tale again: It can not be thou dost but say' tis so: I trust I may not trust thee; for thy word Is but the vain breath of a common man: Believe me, I do not believe thee, man; I have a king's oath to the contrary.	●	○	○
Why, he hacked it with his dagger, and said he would swear truth out of England but he would make you believe it was done in fight, and persuaded us to do the like.	○	●	○
Me believe it!	○	●	○
I believe, not any.	○	●	○

Relevant sentences

What I believe I'll wail, What know believe, and what I can redress, As shall find the time to friend, I will.

For Lord Hamlet, Believe so much ir him, that he is young, And with a larger tether may he walk Than ma be given you: in few, Ophelia, Do nc believe his vows; for they are broke Not of that dye which their investments show, But mere implorators of unholy suits, Breathi like sanctified and pious bonds, The better to beguile.

Non-relevant sentences

It is not so; thou hast misspoke, misheard; Be well advised, tell o'er t tale again: It can not be thou dost t say' tis so: I trust I may not trust thee; for thy word Is but the vain breath of a common man: Believe r I do not believe thee, man; I have a king's oath to the contrary.

Figure 7.1: The relevance-feedback user interface. Participants could mark results of a search query either relevant or not relevant (or leave them as neutral). The marked sentences appeared on the right side of the screen. If a non-zero number of sentences was marked, the "Refine Results" button would appear. Participants could click it to refine their results. If not, they could re-formulate their search query and Search with the Search button

When sentences were marked relevant or not relevant, the query vector Q was adjusted toward the relevant sentences, and away from the irrelevant sentences, according to the following weights:

$$Q' = Q + \frac{1}{n_1} \sum_{\text{relevant}} \frac{S_i}{|S_i|} - \frac{1}{n_2} \sum_{\text{not relevant}} \frac{S_i}{|S_i|} \tag{7.6}$$

where n_1 was the number of relevant sentences and n_2 was the number of non-relevant sentences. Q' was then used to retrieve more sentences.

7.1.3 User Interface

We implemented a minimal user interface for the relevance feedback system, shown in Figure 7.1. The system was initialized either with a set of examples or a search query. Users were given the following controls: a search box, a "Search" button, a "Reset" button, and a "Refine results" button.

Searching or Refining produced a list of search results in a table. For each result, participants could mark one of three relevance-feedback radio buttons: relevant or not relevant (or leave the default choice, neutral). If a sentence was marked anything other than neutral, it would appear in a list on the right side of the screen. Participants could undo their choices by clicking the "X" button next to a sentence in a list or by changing their choice in the radio buttons.

After sentences were marked, clicking the Refine Results button displayed a new set of results produced through relevance feedback. The participant then marked more sentences and repeated the process. They could also perform a search instead.

If a search was performed after a relevance feedback step, a two-step process would ensue. First, any marked sentences not already integrated into the relevance feedback model were integrated into the query. Second, the resulting query vector was added with equal weight to the new search query. This ensured that the relevance feedback already issued was not lost when the user typed in a new search query.

7.1.4 Study Design

Our goal was to determine whether relevance feedback was better able to support finding literary themes than keyword search alone. We decided upon a between-participants study design with a single theme from Shakespeare. Participants were randomly assigned to either the relevance feedback system or the search-only system. Both systems retrieved results from the same collection: the complete works of Shakespeare, published as electronic texts by the Internet Shakespeare Editions (`http://internetshakespeare.uvic.ca`).

The search-only system used the same vector space retrieval model as the relevance feedback system, except that feedback was disabled – there was no "Refine Results" button, and the system did not adjust the query vector in the direction of sentences marked relevant.

Participants were shown an explanation of the theme, with two examples. Then, they were asked to find as many more examples of the same theme as they could within five minutes. When the time limit expired, the sentences that the participants had marked relevant were logged.

Finally, the participants were taken to a self-evaluation questionnaire. On a scale of 1 to 5, they were asked to rate their understanding of the task, their understanding of the theme, their perception of how easy or difficult the task was, and their perception of how well they performed. Lower numbers were worse, and higher numbers were better.

We instrumented the study so that people could take it remotely (using cookies to guard against repeat participation). We logged the number of searches, number of 'refine results' cycles, and the numbers and ID's of sentences marked relevant and not relevant.

Since we were dealing with literary themes, we decided to restrict participation to people with at least college-level backgrounds in English language or literature. Before starting the study, participants were asked to self-report their level of experience in English language or literature. Calls for participation were tweeted by the authors, and sent to mailing lists at the English departments at Berkeley and Stanford.

The theme in the study was chosen by our domain-expert collaborator, Dr. Michael Ullyot at the University of Calgary. Dr. Ullyot is a Professor in the English department, and regularly teaches Shakespeare courses. The theme he chose was "the world as a stage", in which imagery relating to actors, acting, or the theater is used in relation to life, or real-world events. The two examples he selected to illustrate the theme were:

> All the world's a stage,
> And all the men and women merely players:

They have their exits and their entrances;
And one man in his time plays many parts,
His acts being seven ages.
(As You Like It, Act 2 Scene 7 Lines 139–143)

Life's but a walking shadow, a poor player
That struts and frets his hour upon the stage
And then is heard no more: it is a tale
Told by an idiot, full of sound and fury,
Signifying nothing.
(Macbeth Act 5 Scene 5 Lines 23–27)

In the relevance feedback condition, the query vectors were automatically adjusted to include the two sentences above.

Expert Evaluation

Once all participants had finished, we submitted their sentences to our Shakespeare scholar, Dr. Ullyot. He marked each sentence either relevant or not relevant to the "world as a stage" theme. Using these scores, we were able to derive the set of 46 sentences identified as relevant across all participants and the number of relevant sentences found by each participant. We were thus able to compute precision and recall for each participant, with recall defined against the union over all the participants of sentences judged relevant by our expert.

7.1.5 Results

23 participants with the requisite background completed the study. Of these, 11 had PhDs, 3 had Master's degrees, 6 had bachelors' degrees, and 3 were current undergraduates in English language or literature. The search-only condition received 12 participants, and the relevance feedback condition received 11.

We were interested in differences between the following observables across the two systems:

- Number of sentences found,
- Number of relevant sentences found,
- Precision,
- Recall,
- Perceived satisfaction with own performance,
- Perceived difficulty of task
- Number of searches performed

Our hypothesis was that relevance feedback would be more effective than keyword search-only at helping find examples of the theme. In terms of our observables, this implied the

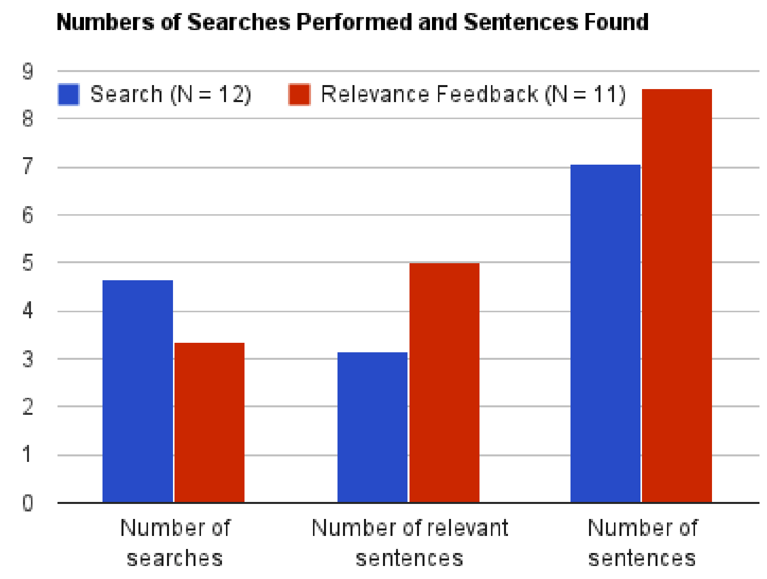


Figure 7.2: A comparison of the average number of searches performed, number of sentences found, and the number of relevant sentences found across the two conditions, relevance feedback (red) and keyword-search only (blue). The differences are consistent with our hypothesis that relevance feedback (red) makes it easier to find relevant sentences. With relevance feedback, there are fewer searches, but more sentences, and more relevant sentences.

following changes: more sentences, and more relevant sentences, higher precision and recall, higher task satisfaction (with no increase in perceived difficulty), and fewer searches.

To test our hypothesis, we performed a two-sided Wilcoxson rank sum test on each of the above observables across the two conditions. We chose this test because we did not have paired samples, and we could not assume that the observations were normally distributed.

Our results were suggestive, but not statistically significant. For all observables, the average values in the relevance-feedback condition differed from the average values in the search condition in the direction consistent with our hypothesis. On average, participants in the relevance feedback condition found more sentences, of which more were relevant, and performed fewer searches in order to do so (Figure 7.2). This resulted in higher precision and recall (Figure 7.3). They also had higher task satisfaction, with no change in perceived difficulty (Figure 7.4).

However, none of the differences were statistically significant, and the standard deviations on the differences (Table 7.1) were so broad as to render the study inconclusive. Larger sample sizes would have reduced this uncertainty.

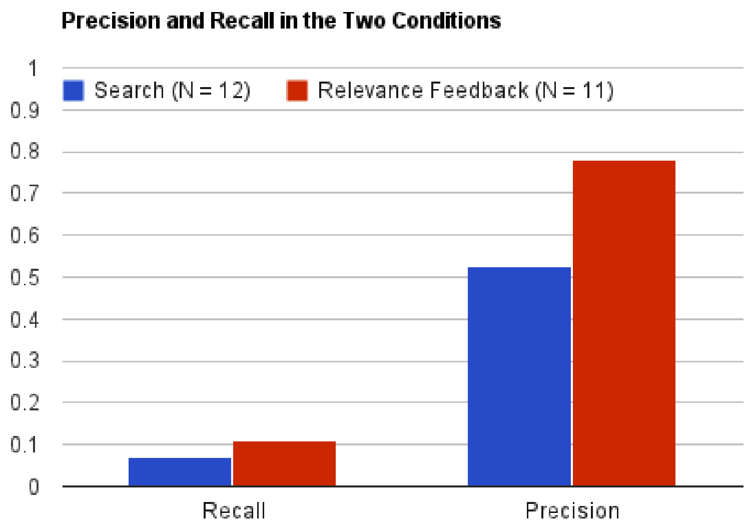


Figure 7.3: A comparison of average precision and recall across the two conditions, relevance feedback (red) and keyword-search only (blue). The increased precision and recall in the relevance feedback condition (red) is consistent with our hypothesis that relevance feedback is an effective aid for finding literary themes.

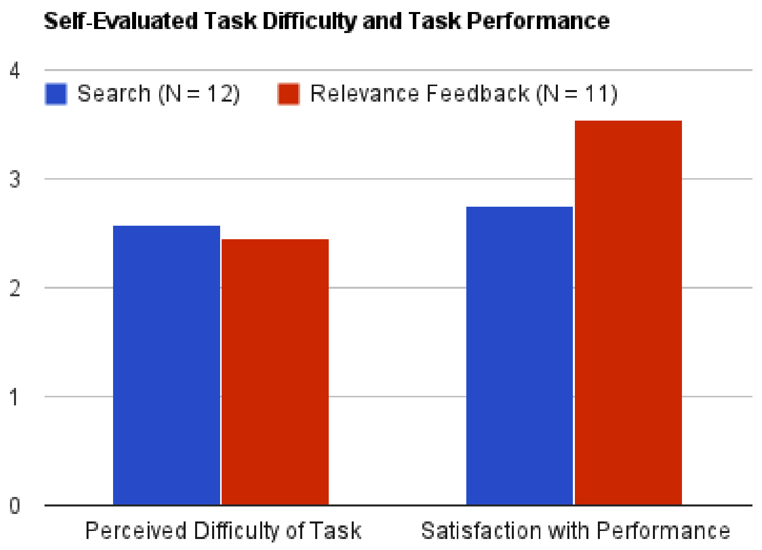


Figure 7.4: A comparison of the average self-evaluated task difficulty and performance satisfaction across the two conditions, relevance feedback (red) and keyword-search only (blue). Participants in the relevance feedback condition (red) were more satisfied with their performance, consistent with our hypothesis that relevance feedback makes it easier to find relevant sentences.

	Relevance Feedback (N=11)		**Search(N=12)**	
	Average	**Std. Dev.**	**Average**	**Std. Dev.**
Number of Searches	3.36	1.80	4.67	4.64
Number of Relevant Sentences	7.00	3.22	4.83	3.56
Number of Sentences	8.64	3.29	7.08	3.92
Self-reported difficulty	2.45	1.04	2.58	1.16
Self-reported performance	3.55	1.29	2.75	1.14
Precision	0.91	0.12	0.78	0.33
Recall	0.17	0.08	0.12	0.09

Table 7.1: Results from the relevance feedback experiment.

7.2 Displaying Syntactic Relations in a Recognizable Way

7.2.1 Introduction

The ability to search over grammatical relationships between words is useful in many fields. For example, a social scientist trying to characterize different perspectives on immigration might ask how adjectives applying to 'immigrant' have changed in the last 30 years. A scholar interested in gender might search a collection to find out whether different nouns enter into possessive relationships with 'his' and 'her' (Muralidharan and Hearst 2013).In other fields, grammatical queries can be used to develop patterns for recognizing entities in text, such as medical terms (Hirschman *et al.* 2005; MacLean and Heer 2013), and products and organizations (Culotta and McCallum 2005), and for coding qualitative data such as survey results.

In the case studies described in this dissertation (Chapters 3 and 6), grammatical search was used to discover stereotypes in the North American slave narratives, (Chapter 3.1), textual evidence of the U.S.'s perception of China as a cold-war strategy 'card' during the 1980s (Chapter 6.1), and lists of concepts that were 'believed in' in Early American fiction (Chapter 6.3).

However, the scholars we worked with did not use WordSeer's grammatical search capabilities as much as they would have liked. As the group of students in the final Case Study 3 (Chapter 6.3) said:

> WordSeer provides a rich array of syntactical and relational offerings when it comes to slicing up a text, which we admit we still have not probably utilized to the fullest.

In part, this is because they were unfamiliar with the formal linguistic terms we used to indicate relationships (such as 'clausal complement' and 'participle modifier').

Most existing interfaces for *syntactic search* (querying over grammatical and syntactic structures) require complex program-like syntax. For example, the popular Stanford Parser includes Tregex, which allows for sophisticated regular expression search over syntactic tree structures and Tsurgeon, which allows for manipulation of the trees extracted with Tregex (Levy and Andrew 2006). The Finite Structure Query tool requires its queries to be stated in first order logic (Kepser 2003). In the Corpus Query Language (Jakubicek *et al.* 2010), a query is a pattern of attribute-value pairs.

However, most potential users of syntactic search do not know how to program. One survey found that even though linguists wished to make very technical linguistic queries, 55% of them did not know how to program (Soehn *et al.* 2008). According to another survey (Gibbs and Owens 2012), humanities scholars and social scientists are frequently skeptical of digital tools, because they are often difficult to use. This reduces the likelihood that existing structured-query tools for syntactic search will be usable by non-programmers.

A related approach is the query-by-example work seen in the past in interfaces to database systems (Androutsopoulos *et al.* 1995). For instance, the Linguist's Search Engine (Resnik *et al.* 2005) uses a query-by-example strategy in which a user types in an initial sentence in English, and the system produces a graphical view of a parse tree as output, which the user can alter. According to Shneiderman and Plaisant (Shneiderman and Plaisant 2010), query-by-example has largely fallen out of favor as a user interface design approach. A downside of QBE is that the user must manipulate an example to arrive at the desired generalization.

At the same time, a related technique, auto-suggest, has become a widely-used approach in search user interfaces with strong support in terms of its usability (Hearst 2009c). A list of selectable options is shown under the search bar, filtered to be relevant as the searcher types. Searchers can recognize and select the option that matches their information need, without having to generate the query themselves.

The success of auto-suggest depends upon showing users options they can recognize. However, we know of no prior work applying auto-suggest to syntactic search. In our own case studies (Chapter 6), the scholars who used WordSeer did not make significant use of the syntactic search functionality. This leaves open the question of how syntactic relationships should be presented to make them more recognizable. WordSeer's current presentation (Chapter 3.1) (not used with auto-suggest) is to name the relation and show blanks where the words that satisfy it would appear as in *X is the subject of Y* (Muralidharan and Hearst 2013); this was the baseline condition. Following the principle of recognition over recall, we hypothesized that showing contextualized usage examples would make the relations more recognizable.

We gave participants a series of identification tasks. In each task, they were shown a list of sentences containing a particular syntactic relationship between highlighted words. They were asked to identify the relationship type from a list of four options. We presented the options in three different ways, and compared the accuracy.

We chose Amazon's Mechanical Turk (MTurk) crowdsourcing platform as a source of study participants. The wide range of backgrounds provided by MTurk is desirable because our goal is to find a representation that is understandable to most people, not just linguistic experts or programmers.

Our results confirm that showing examples in the form of words or phrases significantly improves the accuracy with which grammatical relationships are recognized over a standard baseline. Our findings also showed that clausal relationships, which span longer distances in sentences, benefited significantly more from example phrases than either of the other treatments.

These findings suggest that a query interface in which a user enters a word of interest and the system shows candidate grammatical relations augmented with examples from the text will be more successful than the baseline of simply naming the relation and showing gaps where the participating words appear.

7.2.2 Experiment

Our hypothesis was:

> Grammatical relations are identified more accurately when shown with examples of contextualizing words or phrases than without.

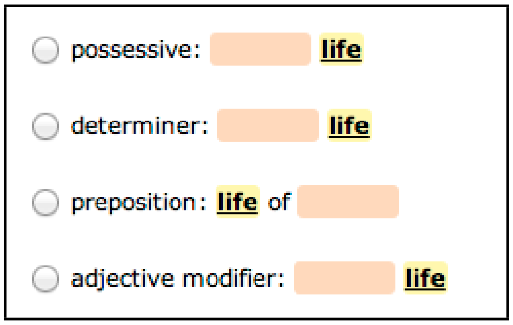
To test it, participants were given a series of identification tasks. In each task, they were shown a list of 8 sentences, each containing a particular relationship between highlighted words. They were asked to identify the relationship from a list of 4 choices. Additionally, one word was chosen as a *focus word* that was present in all the sentences, to make the relationship more recognizable ("life" in Figure 7.5).

The choices were displayed in 3 different ways (Figure 7.5). The **baseline** presentation (Figure 7.5a) named the linguistic relation and showed a blank space with a pink background for the varying word in the relationship, the focus word highlighted in yellow and underlined, and any necessary additional words necessary to convey the relationship (such as "of" for the prepositional relationship "of", the third option).

The **words** presentation showed the baseline design, and in addition beneath was the word "Examples:" followed by a list of 4 example words that could fill in the pink blank slot (Figure 7.5b). The **phrases** presentation again showed the baseline design, beneath which was the phrase "Patterns like:" and a list of 4 example phrases in which fragments of text including both the pink and the yellow highlighted portions of the relationship appeared (Figure 7.5c).

We used a between-subjects design. The task order and the choice order were not varied: the only variation between participants was the presentation of the choices. To avoid the possibility of guessing the right answer by pattern-matching, we ensured that there was no overlap between the list of sentences shown, and the examples shown in the choices as words or phrases. Not every relationship shown in the distractors was tested, and distractors were chosen to be ones intuitively most likely to be mistaken for the relationship shown.

The tasks were generated using the Stanford Dependency Parser (De Marneffe *et al.* 2006) on the text of *Moby Dick* by Herman Melville. We tested the 12 most common grammatical relationships in the novel in order to cover the most content and to be able to provide as many real examples as possible. These relationships fell into two categories, listed below with examples.



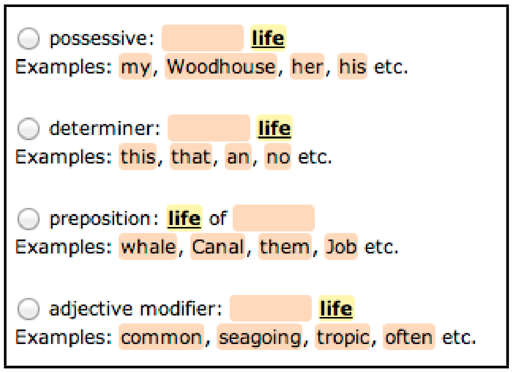
○ possessive: ___ **life**

○ determiner: ___ **life**

○ preposition: **life** of ___

○ adjective modifier: ___ **life**

(a) The options as they appear in the *baseline* condition.



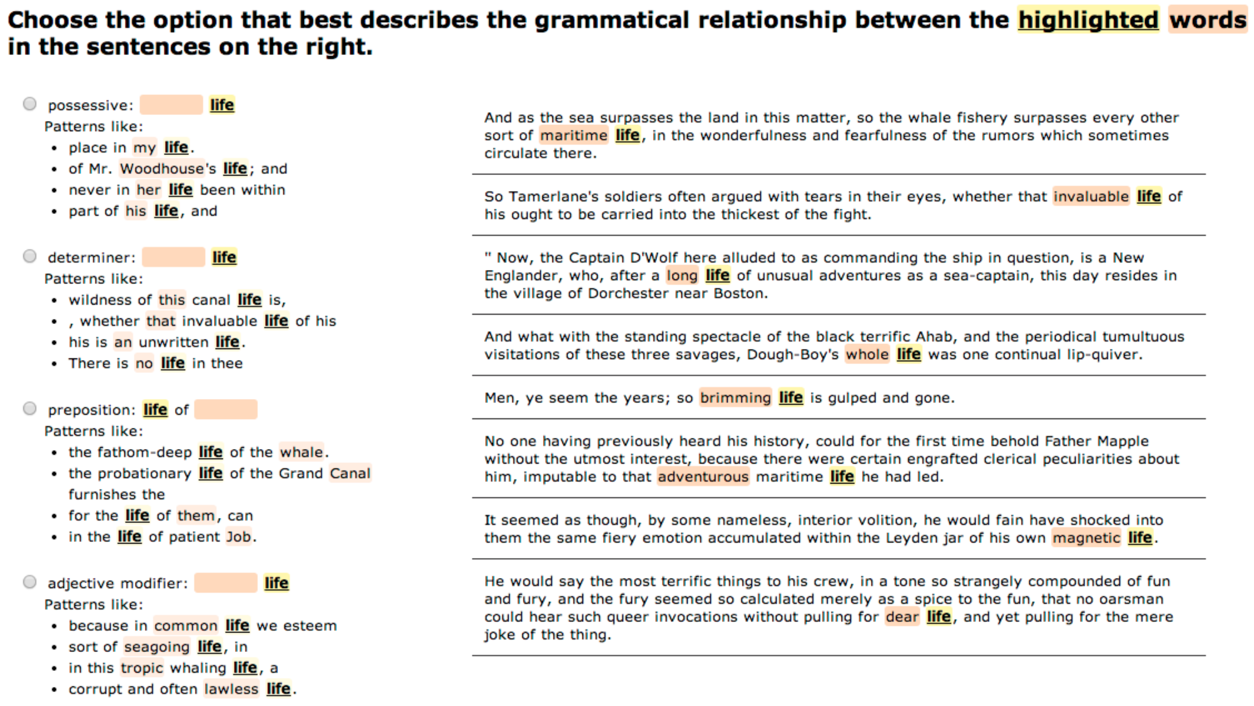
○ possessive: ___ **life**
Examples: my, Woodhouse, her, his etc.

○ determiner: ___ **life**
Examples: this, that, an, no etc.

○ preposition: **life** of ___
Examples: whale, Canal, them, Job etc.

○ adjective modifier: ___ **life**
Examples: common, seagoing, tropic, often etc.

(b) The same options as they appear in the *words* condition.



Choose the option that best describes the grammatical relationship between the highlighted words in the sentences on the right.

○ possessive: ___ **life**
Patterns like:
- place in my **life**.
- of Mr. Woodhouse's **life**; and
- never in her **life** been within
- part of his **life**, and

○ determiner: ___ **life**
Patterns like:
- wildness of this canal **life** is,
- , whether that invaluable **life** of his
- his is an unwritten **life**.
- There is no **life** in thee

○ preposition: **life** of ___
Patterns like:
- the fathom-deep **life** of the whale.
- the probationary **life** of the Grand Canal furnishes the
- for the **life** of them, can
- in the **life** of patient Job.

○ adjective modifier: ___ **life**
Patterns like:
- because in common **life** we esteem
- sort of seagoing **life**, in
- in this tropic whaling **life**, a
- corrupt and often lawless **life**.

And as the sea surpasses the land in this matter, so the whale fishery surpasses every other sort of maritime **life**, in the wonderfulness and fearfulness of the rumors which sometimes circulate there.

So Tamerlane's soldiers often argued with tears in their eyes, whether that invaluable **life** of his ought to be carried into the thickest of the fight.

" Now, the Captain D'Wolf here alluded to as commanding the ship in question, is a New Englander, who, after a long **life** of unusual adventures as a sea-captain, this day resides in the village of Dorchester near Boston.

And what with the standing spectacle of the black terrific Ahab, and the periodical tumultuous visitations of these three savages, Dough-Boy's whole **life** was one continual lip-quiver.

Men, ye seem the years; so brimming **life** is gulped and gone.

No one having previously heard his history, could for the first time behold Father Mapple without the utmost interest, because there were certain engrafted clerical peculiarities about him, imputable to that adventurous maritime **life** he had led.

It seemed as though, by some nameless, interior volition, he would fain have shocked into them the same fiery emotion accumulated within the Leyden jar of his own magnetic **life**.

He would say the most terrific things to his crew, in a tone so strangely compounded of fun and fury, and the fury seemed so calculated merely as a spice to the fun, that no oarsman could hear such queer invocations without pulling for dear **life**, and yet pulling for the mere joke of the thing.

(c) The same options in the *phrases* condition, shown as they appeared in an identification task for the relationship `amod(life, ___)` (where different adjectives modify the noun 'life'). The correct answer is 'adjective modifier' (4th option), and the remaining 3 options are distractors.

Figure 7.5: The appearance of the choices shown in the three experiment conditions.

Clausal or long-distance relations:

– Adverbial clause: *I **walk** while **talking***
– Open clausal complement: *I **love** to **sing***
– Clausal complement: *he **saw** us **leave***
– Relative clause modifier: *the **letter** I **wrote** reached*

Non-clausal relations:

– Subject of verb: ***he threw** the ball*
– Object of verb: *he **threw** the **ball***
– Adjective modifier ***red ball***
– Preposition (in): *a **hole** in a **bucket***
– Preposition (of): *the **piece** of **cheese***
– Conjunction (and) ***mind** and **body***
– Adverb modifier: *we **walk slowly***
– Noun compound: ***Mr. Brown***

We tested each of these relations 4 times, with 2 different focus words in each role. For example, the *Subject of Verb* relation was tested in the following forms:

– `(Ahab, ___)`: the sentences each contained 'Ahab', highlighted in yellow, as the subject of different verbs highlighted in pink.
– `(captain, ___)`
– `(___, said)`: the sentences each contained the verb 'said', highlighted in yellow, but with different subjects, highlighted in pink.
– `(___, stood)`

To maximize coverage, yet keep the total task time reasonable (average 6.8 minutes), we divided the relations above into 4 task sets of 3 relations each. Each relation was tested with 4 different words, making a total of 12 tasks per participant.

Participants

400 participants completed the study distributed randomly over the 4 task sets and the 3 presentations. Participants were paid 50c (U.S.) for completing the study, with an additional 50c bonus if they correctly identified 10 or more of the 12 relationships. They were informed of the possibility of the bonus before starting.

To gauge their syntactic familiarity, we also asked them to rate how familiar they were with the terms 'adjective' (88% claimed they could define it), 'infinitive' (43%), and 'clausal complement' (18%). To help ensure the quality of effort from participants, we included a multiple-choice screening question, "What is the third word of this sentence?" Those that answered incorrectly were eliminated.

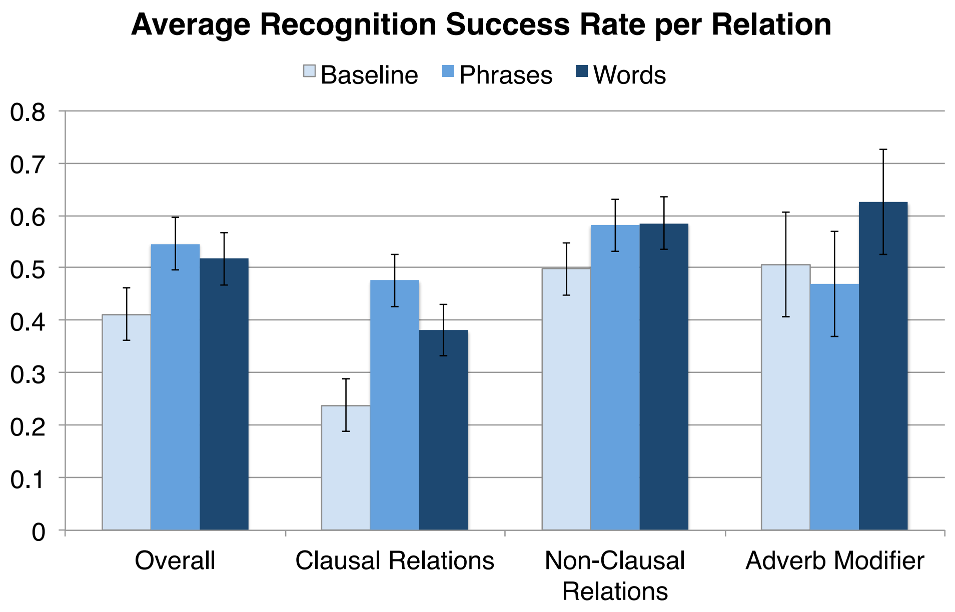


Figure 7.6: Recognition rates for different types of relations under the 3 experiment conditions, with 95% confidence intervals.

Results

The results (Figure 7.6) confirm our hypothesis. Participants in conditions that showed examples (**phrases** and **words**) were significantly more accurate at identifying the relations than participants in the **baseline** condition. We used the Wilcoxson signed-rank test, an alternative to the standard T-test that does not assume samples are normally distributed. The average success rate in the **baseline** condition was 41%, which is significantly less accurate than **words**: 52%, (p=0.00019, W=6136), and **phrases**: 55%, (p=0.00014, W=5546.5).

Clausal relations operate over longer distances in sentences, and so it is to be expected that showing longer stretches of context would perform better in these cases; that is indeed what the results showed. Phrases significantly outperformed words and baseline for clausal relations. The average success rate was 48% for **phrases**, which is significantly more than **words**: 38%, (p=0.017 W=6976.5) and **baseline**: 24%, ($p=1.9\times10^{-9}$ W=4399.0), which was indistinguishable from random guessing (25%). This is a strong improvement, given that only 18% of participants reported being able to define 'clausal complement'.

For the non-clausal relations, there was no significant difference between **phrases** and **words**, although they were both overall significantly better than the baseline (words: p=0.0063 W=6740, phrases: p=0.023 W=6418.5). Among these relations, adverb modifiers stood out (Figure 7.6), because evidence suggested that **words** (63% success) made the relation more recognizable than **phrases** (47% success, p=0.056, W=574.0) – but the difference was only almost significant, due to the smaller sample size (only 96 participants encountered this relation). This may be because the words are the most salient piece of information in an adverbial relation – adverbs usually end in ‘ly’ – and in the phrases condition the additional information distracts from recognition of this pattern.

7.2.3 Discussion

The results imply that auto-suggest interfaces for syntactic search should show syntactic relationships augmented with a list of phrases in which they occur. A list of phrases is the most recognizable presentation for clausal relationships (34% better than the baseline), and is as good as a list of words for the other types of relations.

Chapter 8

Future Work and Contributions

8.1 Future Work

Future directions for the WordSeer project fall into three categories: improving search, adding more analytical tools, and improving the user interface.

8.1.1 Search

At present, WordSeer supports keyword and grammatical search. In the previous chapter, we reported on two investigations aimed at improving the search experience. In the first investigation, we found that relevance feedback had promise as a solution to the problem of finding conceptually-linked passages of text that do not necessarily match precise keywords or grammatical relationships. In the second, we examined a potential way to improve WordSeer's syntactic search by using the surrounding context to make grammatical relations more recognizable. This section discusses future investigations along those lines, as well as for search over more complex syntactic structures and search-result clustering.

Relevance Feedback

The investigation into relevance feedback (Chapter 7.1) suggested that relevance feedback was more effective than keyword search alone for finding examples literary themes. A continuation of this research would be to conduct a larger study to establish statistical significance. In this second study, statistical power could be increased by collecting paired samples in which each participant does two different theme finding tasks, one on the relevance feedback and one on the search-only system.

There are also improvements that can be made to the relevance feedback system. The first is the size of the retrieved units. When we described the system to our literary-scholar collaborators, a frequent objection was that sentences were an unnatural unit when looking for themes. However, paragraphs might be too big or too small, depending on context. To address this problem, we could segment the text into consecutive topically-coherent units using an approach such as TextTiling (Hearst 1997). Instead of retrieving sentences or paragraphs, we could instead retrieve these units.

Another area for improvement is the feature-space representations of the units. With syntactic parsing, the subject-object and dependent-modifier relationships between words could be extracted and incorporated into the feature vectors. Synonymy could also be employed as a way to broaden the query. The presence of a word in a sentence feature could "activate" (with some diminished weight) the features for all the synonyms of that word. The larger-scale study with the above improvements to the algorithm could then inform the incorporation of relevance feedback into WordSeer.

Autosuggest for Syntactic Search

Our experiment on improving the recognizability of grammatical relations showed that contextual information in the form of a surrounding phrase makes the relations more recognizable. This result could be applied to improve WordSeer's syntactic search capabilities with autosuggest.

A mockup of such an autosuggest interface is shown in Figure 8.1. Selecting the choice will return all sentences that contain the search term and match the relation.

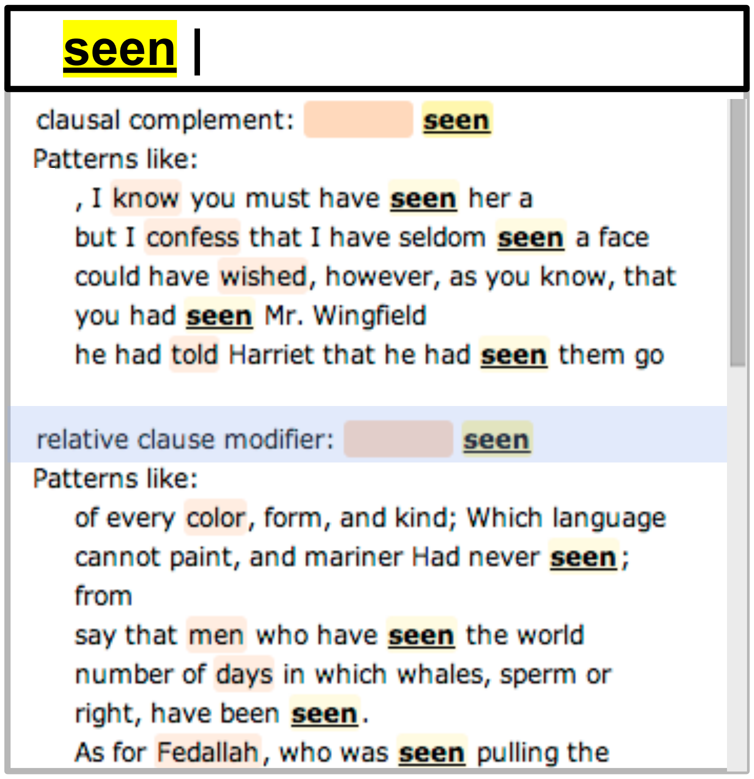
There is a tradeoff between recognizability and space required for scrolling through the choices, although it is important to keep in mind that because the suggestions are populated with phrases from the collection itself, they are informative. Further, the suggestions can be ordered by frequency of occurrence in the collection, or by an interestingness measure given the search word. As the user becomes more familiar with a given relation, it may be expedient to shorten the cues shown, and then re-introduce them if a relation has not been selected after some period of time as elapsed.

However, even the best strategy we found, **phrases**, had an overall success rate of only 55%. There is room for improvement, even though our intended user base may have more familiarity with grammatical relations than the study participants did, and therefore may perform better in practice. Nonetheless, it may be that additional visual cues, such as some kind of bracketing, will improve results. Furthermore, the experiment did not test three-word relationships or more complex combinations of structures, and those may require improvements to the design.

A follow-up to this experiment could be conducted in which different phrase- and word-based presentations are tested in a more realistic setting. Instead of testing recognition, like this experiment, the goal would be to discover a presentation that functions effectively as the user types into an autosuggest box.

Search over Sentence Structures

At present, WordSeer only allows search and exploration of dependency relations between words. However, in our case studies, we found the need for exploration over other syntactic structures. For example, C.F. in the final Case Study on U.S.-China relations (Chapter


seen

clausal complement: seen

Patterns like:

, I know you must have seen her a

but I confess that I have seldom seen a face

could have wished, however, as you know, that you had seen Mr. Wingfield

he had told Harriet that he had seen them go

relative clause modifier: seen

Patterns like:

of every color, form, and kind; Which language cannot paint, and mariner Had never seen; from

say that men who have seen the world

number of days in which whales, sperm or right, have been seen.

As for Fedallah, who was seen pulling the

Figure 8.1: Mockup of autosuggest for syntactic search on the word 'seen', showing clausal relations with example phrases.

6.1), wanted to collect examples of comparative sentences. He was interested in the types of Chinese and Japanese concepts that were compared and contrasted with each other. Sentences of the form "In [Chinese location] ___, but in [Japanese location], ___." would have been especially interesting to him. Similarly, when R.G. was analyzing the literacy autobiographies (Chapter 6.2), he compared the relative frequencies of 'concessive' structures across different class levels. He was able to do this indirectly by using words like `although` and `however`, which can indicate the presence of the structure, but a syntactic-structure search would have made his analysis more precise.

One problem that stands in the way of implementing this functionality is that syntactic parses are not easily retrievable from WordSeer's MySQL databases. They are stored as strings, which means that their sub-structures are not easily traversable using database calls. In collaboration with Carolynn Jimenez, an undergraduate student at the University of California, San Diego, we developed a way of storing traversable tree structures in a MySQL database in order to index syntactic information such as dominancy and headship. The solution allows the database to store the sentence structures in a way that can be retrieved without incurring recursive calls, enabling quick search over parse trees.

Future work could take advantage of this solution by incorporating syntactic structure search into WordSeer's user interface. However, there are user interfaces challenges. In Chapter 7.2, we investigated ways to make dependency relationships more recognizable. We needed a more recognizable presentation than the unfamiliar linguistic terminology we were using to display relationships between words. Searching over syntactic structures, however, is even more complex than searching over dependency relationships. For example, the Tregex syntax for querying parse structures (Levy and Andrew 2006) allows an arbitrary number words and wild-cards, as well as adjacency and hierarchy relationships.

A hybrid of relevance feedback and query by example (Zloof 1975) approaches could be used. Instead of typing in a search expression, a user could collect a few sentences and highlight the portions of each that matched their query. For example, to collect comparative sentences, C.F. could highlight the following sentence:

> "In comparison to Beijing, a northern Chinese city which has close to 20 million people , Tokyo's population is a mere 12 million people."

The system would then examine the sentence sub-structures that had been highlighted, and suggest matches based on distinctive structures that were highlighted more than once. Relevance feedback could be used to refine the search to just the structures that the user had in mind, and then those structures used to return results.

Clustering Search Results

Currently, search results in WordSeer are displayed as a flat list. However, grouping results based on frequent terms, and showing them in clusters has been found to improve the outcomes of exploratory search (Käki and Aula 2005). There are many algorithms for extracting frequent or important terms from sentences Chuang *et al.* (2012); Kummamuru *et* *al.* (2004); Smadja (1993), and grouping sentences according to similarity (Hatzivassiloglou *et al.* 1999; Corley and Mihalcea 2005; Metzler *et al.* 2007). One direction for future work would be evaluate some alternative approaches side by side and use the results to incorporate clustered search into WordSeer.

8.1.2 User Interface

There are currently several user interface problems. The most pressing problem is the rigid left-to-right arrangement of panels, which makes having more than two or three simultaneous views impractical on most displays. A more adaptive, customizable layout would make managing and navigating between views easier. Next, the user's history of actions should be made more accessible. At present history is presented as a linear sequence of actions. The system should instead give the user a sense of the different sensemaking paths they have taken, and allow them to replay their history, and branch off from it. The third problem is that users cannot customize their views according to their needs. The same three overviews are always present: the metadata, and the most frequent phrases, nouns, verbs, and adjectives. The user should be able to choose different overviews.

Visualizations

At present, WordSeer only incorporates two data visualizations: the Word Tree (Wattenberg and Viegas 2008) and the bar charts and time-series Word Frequencies visualization. There are other visualizations that could be incorporated, such as Phrase Nets (Van Ham *et al.* 2009), which show the prevalence of n-gram-based patterns (such as the pattern '`___ and ___`'), network diagrams showing links between documents based on search terms, and term co-occurrence heat-maps.

Additionally, visualizations of different slices can be viewed side by side, but cannot be compared within a single visualization. While analyzing belief in early American fiction (Chapter 6.3) the students found the Word Frequencies visualization useful, but wanted more direct comparison:

> The ability to overlay one graph onto another would have further amplified our lens for comparison, rather than toggling back and forth between results.

This could be implemented with drag and drop between visualization panels. Users could open up different panels with the slices they wanted to compare, and then drag one onto the other in order to see comparative visualizations.

8.1.3 Analytical Features

Natural Language Processing

WordSeer uses off-the-shelf natural language parsers to perform syntactic processing. There are other more high-level, but similarly off-the-shelf techniques that could be incorporated for extracting named entities (stanford parser), topic modeling, and sentiment analysis. These could easily be added to the processing pipeline, which could extract the information from the text during pre-processing and store it database tables. The information could then be indexed according to the sentences in which the entities, topics, or sentiments appeared, in the same way that words, sequences, grammatical relationships and metadata are currently indexed. Then, in addition to providing overviews of frequent words and phrases, the interface could display frequent entities or topics. Such extraction and indexing would also allow browsable categories to be added for each entity or topic (just like WordSeer's current metadata categories), which would enable discovery and filtering over the extracted information.

Hypothesis Testing and Discovery

As an exploratory data analysis (EDA) system, WordSeer aids in the formulation of hypotheses, and the accumulation of evidence in favor or in refutation of hypothesis. A significant improvement to the tool would be a way to help users try to disprove any hypothesis that they think the tool has helped them to find. For example, assessments of literacy essays could be compared to learning outcomes and grades. This would require the development of a way to upload numerical or categorical data into WordSeer, as an addition to the data accompanying an existing collection.

In a similar vein, simple statistical tests could be added. A user could select two slices to compare, and the system could indicate whether any differences in the frequencies of particular words were statistically significant. With built-in metadata, such as time or various property values, the system could automatically compute whether there were any statistically significant differences in the distributions of words or phrases across those categories. If any were found, the system could call attention to them. Such a system would draw the user's attention if, for example, the character of Gertrude spoke a certain word significantly more or less often than Ophelia in Shakespeare's *Hamlet.*

8.2 Contributions

The contributions of this dissertation are the design and source code of the WordSeer exploratory text analysis system, a descriptive model of how humanities and social science scholars undertake exploratory text analysis during the course of their work, and design guidelines for future systems.

We identified an unmet need for systems tha support exploratory text analysis in the humanities and social sciences. Other current systems for this audience support algorithmic and visual analysis, but not collecting evidence, isolating and analyzing sub-sets of the collection, making comparisons based on collected items, and exploring a new idea without interrupting the current task.

In order to develop a better system, we adopted a user-centered design approach based on case studies. We conducted the first case study with Professor Bryan Wagner at UC Berkeley, who was interested in analyzing narrative stereotypes in a collection of North American slave narratives. We developed grammatical search to help identify instances of stereotypes, and a newspaper-strip visualization to visualize their prevalence. We were able to identify a few conventions, but the experience was ultimately unsatisfying for Professor Wagner. His feeback made it clear that the system, WordSeer, needed to support more than cut-and-dry hypothesis testing; it needed to support sensemaking: a cycle involving exploration, collection, curation, and comparison. In response, we added collection and exploration features such as sets of documents, which could be used as inputs to visualizations, and popups showing related words, which could be accessed through a contextual menu without interrupting the current task.

In the second case study, the redesigned WordSeer was used by a semester-long class of undergraduate Shakespeare students at the University of Calgary. The students used WordSeer and four other similar text analysis tools to analyze Shakespeare's *Hamlet* and posted about their experiences and findings weekly on the class blog. Their accounts showed that sensemaking in the humanities and social sciences had important differences from other fields. They wanted to analyze text at the word, phrase, and sentence level, and to isolate and analyze 'slices' of the collection according to combinations of searches and within-document metadata properties. They also criticized user interface flows that made frequently-performed sequences of actions, such as comparisons, and exploring a new train of thought, cumbersome and disruptive to their train of thought.

To get a better understanding of the text analysis processes of humanities scholars and social scientists, we conducted informal open-ended conversations with English and History PhD students at UC Berkeley. We asked them about their work processes, the types of questions they analyzed, their research goals, the different activities they performed and the frustrations they experienced. By summarizing their responses, we developed a descriptive model of the their analysis processes. We were also able to identify specific pain points that could benefit from computational assistance.

To address the pain points, we developed principles for user interface interaction flows and information display and navigation and redesigned WordSeer to implement these principles. The redesigned system supports highly flexible slicing and dicing, as well as easier transitions than in other tool between visual analyses, drill-downs, lateral explorations and overviews of slices in a text collection. The source code for this version is available open-source.

To validate the redesign, we conducted a final set of three case studies at UC Berkeley. An English PhD student analyzed a collection of *New York Times* editorials about China and Japan to find evidence for shifts in the tone of the relationship between the U.S. and China. In the second study, a PhD student in the School of Education analyzed themes and writing styles in a collection of essays written by college students describing their own experiences with literacy. In the third study, a group of one English PhD student and two undergraduates analyzed the shifting connotations of certain words in early American fiction. As a measure of the system's success, all users were able to conduct analyses yielding otherwise inaccessible results useful to their research.

Finally, there were user interface and search features we investigated in isolation: grammatical search and query-by-example. In all our case studies, we found that users did not issue very many grammatical search queries in comparison to keyword search queries. Hypothesizing that this might be due to their lack of familiarity with the linguistic terminology we were using, we experimented with ways to make grammatical relations more recognizable. We found that putting the relationships into context using examples of phrases containing the relationships, as well as showing lists of words that entered into those relationships, improved recognition. Our investigation into query-by-example was motivated by several examples from the case studies in which users wanted to find passages of text that matched a concept, but could not put that concept into words as a search query. We investigated relevance feedback as a way to tackle this problem: users could mark examples of sentences that matched their query, and the system would use their feedback to improve its results. Our experiment indicated that relevance feedback had promise as a way to find examples of sentences matching a theme.

Bibliography

M. H. Abrams and G. G. Harpham. *A Glossary of Literary Terms.* Cengage Learning, January 2011.

A. Adler, A. Gujar, B. L. Harrison, K. O'Hara, and A. Sellen. A diary study of work-related reading: design implications for digital reading devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '98, pages 241–248, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

C. Ahlberg. Spotfire: an information exploration environment. *SIGMOD*, 25(4):25–29, 1996.

W. L. Andrews. *To Tell a Free Story: The First Century of Afro-American Autobiography, 1760-1865.* University of Illinois Press, 1988.

W. L. Andrews. An introduction to the slave narrative, 2004.

I. Androutsopoulos, G. Ritchie, and P. Thanisch. Natural language interfaces to databases–an introduction. *Natural Language Engineering*, 1(01):29–81, 1995.

R. Badi, S. Bae, J. M. Moore, K. Meintanis, A. Zacchi, H. Hsieh, F. Shipman, and C. C. Marshall. Recognizing user interest and document value from reading and organizing activities in document triage. In *Proceedings of the 11th international conference on Intelligent user interfaces*, pages 218–225, 2006.

P. Belanoff, P. Elbow, and S. I. Fontaine. Introduction. In *Nothing Begins with N: New Investigations of Freewriting.* SIU Press, January 1991.

M. S. Bernstein, B. Suh, L. Hong, J. Chen, S. Kairam, and E. H. Chi. Eddi. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology - UIST '10*, page 303, New York, New York, USA, 2010.

T. Blanke and M. Hedges. Scholarly primitives: Building institutional infrastructure for humanities e-science. *Future Generation Computer Systems*, 29(2):654–661, February 2013.

M. Bron, J. van Gorp, F. Nack, M. de Rijke, A. Vishneuski, and S. de Leeuw. A subjunctive exploratory search interface to support media studies researchers. In *SIGIR*, pages 425–434. ACM, 2012.

G. Buchanan and F. Loizides. Investigating document triage on paper and electronic media. In *Research and Advanced Technology for Digital Libraries*, pages 416–427. Springer, 2007.

D. H. Chau, A. Kittur, J. I. Hong, and C. Faloutsos. Apolo: making sense of large network data by combining rich user interaction and machine learning. In *CHI*, pages 167–176. ACM, 2011.

J. Chuang, C. D. Manning, and J. Heer. "Without the clutter of unimportant words": Descriptive keyphrases for text visualization. *ACM Trans. Comput.-Hum. Interact.*, 19(3):19:1–19:29, October 2012.

T. E. Clement, C. Plaisant, and R. Vuillemot. The story of one: Humanity scholarship with visualization and text analysis. *Relation*, 10:1–43, 2009.

T. E. Clement. ‘a thing not beginning and not ending’: using digital tools to distant-read gertrude stein’s the making of americans. *Literary and Linguistic Computing*, 23(3):361, 2008.

C. Corley and R. Mihalcea. Measuring the semantic similarity of texts. In *Proceedings of the Association for Computationsal Linguistics Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, EMSEE ’05, pages 13–18, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

P. Cowley, L. Nowell, and J. Scholtz. Glass box: An instrumented infrastructure for supporting human interaction with information. In *System Sciences, 2005. HICSS’05. Proceedings of the 38th Annual Hawaii International Conference on*, pages 296c–296c, 2005.

A. Culotta and A. McCallum. Reducing labeling effort for structured prediction tasks. In *AAAI*, pages 746–751, 2005.

M.-C. De Marneffe, B. MacCartney, C. D. Manning, et al. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454, 2006.

A. Don, E. Zheleva, M. Gregory, S. Tarkan, L. Auvil, T. E. Clement, B. Shneiderman, and C. Plaisant. Discovering interesting usage patterns in text collections: integrating text mining with visualization. In *Proc. ACM CIKM*, pages 213–222, Lisbon, Portugal, 2007. ACM.

K. Donn, C. Plaisant, and B. Shneiderman. Query previews in networked information systems. In *Advances in Digital Libraries ‘96.*, pages 120–129, 1996.

T. Eagleton. *Literary Theory: An Introduction.* John Wiley & Sons, November 2011.

S. C. Eick, J. L. Steffen, and E. E. Sumner Jr. Seesoft-a tool for visualizing line oriented software statistics. *Software Engineering, IEEE Transactions on*, 18(11):957–968, 1992.

P. Elbow. Teaching thinking by teaching writing. *Change: The Magazine of Higher Learning*, 15(6):37–40, 1983.

D. K. Elson, N. Dames, and K. R. McKeown. Extracting social networks from literary fiction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 138–147, 2010.

J. Fekete and N. Dufournaud. Compus: visualization and analysis of structured documents for understanding social life in the 16th century. In *Proc. ACM Digital libraries*, DL ’00. ACM, 2000. ACM ID: 336632.

B. Frischer, J. Unsworth, A. Dwyer, A. Jones, L. Lancaster, G. Rockwell, and R. Rosenzweig. Summit on digital tools for the humanities: Report on summit accomplishments. *Charlottesville, VA*, 23, 2006.

H. Froehlich. Independent women? representations of gender-specific possession in two shakespeare plays. In *Papers from the 7th Lancaster University Postgraduate Conference in Linguistics & Language Teaching 2012*, page 78, 2012.

G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, 1987.

F. Gibbs and T. Owens. Building better digital humanities tools. *DH Quarterly*, 6(2), 2012.

G. Golovchinsky, M. N. Price, and B. N. Schilit. From reading to retrieval: freeform ink annotations as queries. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 19–25, New York, NY, USA, 1999. ACM.

C. Gorg, Z. Liu, J. Kihm, J. Choo, H. Park, and J. Stasko. Combining computational analyses and interactive visualization for document exploration and sensemaking in jigsaw. *IEEE Visualization and Computer Graphics*, PP(99), December 2012.

V. Hatzivassiloglou, J. Klavans, and E. Eskin. Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 203–212, 1999.

M. A. Hearst and D. Degler. Sewing the seams of sensemaking. In *The 7th Annual Symposium on Human-Computer Interaction and Information Retrieval*, Vancouver, British Columbia, Canada, 2013.

M. A. Hearst. TextTiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.*, 23(1):33–64, March 1997.

M. A. Hearst. Chapter 3 models of the information seeking process. In *Search User Interfaces*, pages 64–90. Cambridge University Press, New York, NY, USA, 1st edition, September 2009.

M. A. Hearst. Chapter 7 supporting the search process. In *Search User Interfaces*, pages 157–173. Cambridge University Press, New York, NY, USA, 1st edition, September 2009.

M. A. Hearst. *Search user interfaces.* Cambridge University Press, 2009.

J. Heer and M. Agrawala. Software design patterns for information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):853–860, 2006.

R. B. Heilman. *Magic in the Web: Action and Language in Othello.* University of Kentucky, first edition edition, 1956.

L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia. Overview of biocreative: critical assessment of information extraction for biology. *BMC bioinformatics*, 6(Suppl 1):S1, 2005.

J. M. T. Hong. The informavore shopper: Analysis of information foraging, system design, and purchasing behavior in online retail stores. 2013.

M. Jakubicek, A. Kilgarriff, D. McCarthy, and P. Rychlỳ. Fast syntactic searching in very large corpora for many languages. In *PACLIC*, volume 24, pages 741–747, 2010.

M. Jockers. Detecting and characterizing national style in the 19th century novel. In *Digital Humanities 2011*, Stanford, CA, 2011. Stanford University Libraries.

K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972.

D. Jonker, W. Wright, D. Schroh, P. Proulx, and B. Cort. Information triage with trist. In *2005 International Conference on Intelligence Analysis*, pages 2–4, 2005.

D. Jurafsky and J. H. Martin. Chapter 13 syntactic parsing. In *Speech and language processing*, pages 427 — 459. Pearson Prentice Hall, 2nd ed. edition, 2009.

M. Käki and A. Aula. Findex: improving search result use through automatic filtering categories. *Interacting with Computers*, 17(2):187–206, March 2005.

S. Kepser. Finite structure query: A tool for querying syntactically annotated corpora. In *EACL*, pages 179–186, 2003.

M. Krstajic, E. Bertini, F. Mansmann, and D. A. Keim. Visual analysis of news streams with article threads. In *Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques*, StreamKDD '10, pages 39–46, New York, NY, USA, 2010. ACM. ACM ID: 1833286.

M. Krstajic, F. Mansmann, A. Stoffel, M. Atkinson, and D. Keim. Processing online news streams for large-scale semantic analysis. In *Data Engineering Workshops (ICDEW), 2010 IEEE 26th International Conference on*, pages 215–220, 2010.

B. Kules and R. Capra. Designing exploratory search tasks for user studies of information seeking support systems. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, pages 419–420, 2009.

K. Kummamuru, R. Lotlikar, S. Roy, K. Singal, and R. Krishnapuram. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 658–665, New York, NY, USA, 2004. ACM.

H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale. Empirical studies in information visualization: Seven scenarios. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1520–1536, 2012.

B. Lee, M. Czerwinski, G. Robertson, and B. B. Bederson. Understanding research trends in conferences using paperlens. In *CHI'05 extended abstracts*, pages 1969–1972. ACM, 2005.

J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506, 2009.

R. Levy and G. Andrew. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *Proc. 5th Conference on Language Resources and Evaluation*, pages 2231–2234, 2006.

R. B. Lewis and S. M. Maas. QDA miner 2.0: Mixed-model qualitative data analysis software. *Field Methods*, 19(1):87–108, 2007.

D. Lin. Using syntactic dependency as local context to resolve word sense ambiguity. In *Proc. ACL*, 1997.

X. Llora, B. Acs, L. S. Auvil, B. Capitanu, M. E. Welge, and D. E. Goldberg. Meandre: Semantic-driven data-intensive flows in the clouds. In *Proc. IEEE eScience*, pages 238–245, 2008.

L. Lloyd, D. Kechagias, and S. Skiena. Lydia: A system for large-scale news analysis. In M. Consens and G. Navarro, editors, *String Processing and Information Retrieval*, volume 3772, pages 161–166. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

D. L. MacLean and J. Heer. Identifying medical terms in patient-authored text: a crowdsourcing-based approach. *Journal of the American Medical Informatics Associ-*

ation, 2013.

T. W. Malone. How do people organize their desks?: Implications for the design of office information systems. *ACM Transactions on Information Systems (TOIS)*, 1(1):99–112, 1983.

R. Mander, G. Salomon, and Y. Y. Wong. A âĂIJpileâĂİ metaphor for supporting casual organization of information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 627–634, 1992.

C. C. Marshall and F. M. Shipman III. Spatial hypertext: designing for change. *Communications of the ACM*, 38(8):88–97, 1995.

C. C. Marshall, F. M. Shipman III, and J. H. Coombs. VIKI: spatial hypertext supporting emergent structure. In *Proceedings of the 1994 ACM European conference on Hypermedia technology*, pages 13–23, 1994.

R. Matignon. *Data mining using SAS enterprise miner*, volume 638. Wiley. com, 2007.

D. Metzler, S. Dumais, and C. Meek. Similarity measures for short segments of text. In G. Amati, C. Carpineto, and G. Romano, editors, *Advances in Information Retrieval*, volume 4425 of *Lecture Notes in Computer Science*, pages 16–27. Springer Berlin / Heidelberg, 2007.

L. G. Militello and R. J. Hutton. Applied cognitive task analysis (ACTA): a practitioner's toolkit for understanding cognitive task demands. *Ergonomics*, 41(11):1618–1641, 1998.

F. Moretti. *Graphs, Maps, Trees: Abstract models for a literary history.* Verso Books, 2005.

M. Mueller. Digital shakespeare, or towards a literary informatics. *Shakespeare*, 4(3):284–301, 2008.

M. Mueller. MorphAdorner. http://morphadorner.northwestern.edu/morphadorner, 2009. Accessed January 13th, 2012 2:10 PM.

A. Muralidharan and M. A. Hearst. Supporting exploratory text analysis in literature study. *Literary and Linguistic Computing*, 28(2):283–295, 2013.

A. Muralidharan, M. A. Hearst, and C. Fan. WordSeer: a knowledge synthesis environment for textual data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2533–2536, 2013.

B. A. Nardi. *A small matter of programming: perspectives on end user computing.* The MIT Press, 1993.

V. L. O'Day and R. Jeffries. Orienteering in an information landscape: how information seekers get from here to there. In *Proc. INTERACT and CHI*, CHI '93, Amsterdam, The Netherlands, 1993. Association for Computing Machinery. ACM ID: 169365.

J. Olney. "I was born": Slave narratives, their status as autobiography and as literature. *Callaloo*, 20:46–73, January 1984. ArticleType: research-article / Full publication date: Winter, 1984 / Copyright 1984 The Johns Hopkins University Press.

C. L. Palmer and M. H. Cragin. Scholarship and disciplinary practices. *Annual Review of Information Science and Technology*, 42(1):163–212, 2008.

C. L. Palmer, L. Teffeau, and C. Pirmann. Scholarly information practices in the online environment: Themes from the literature and implications for library service development. January 2009.

G. L. Peterson and A. R. Bryant. *Eliciting a Sensemaking Process from Verbal Protocols of Reverse Engineers*. AIR FORCE RESEARCH LAB WRIGHT-PATTERSON AFB OH HUMAN PERFORMANCE WING (711TH), 2012.

P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proc. International Conference on Intelligence Analysis*, volume 1, pages 2–4, MacLean, VA, USA, 2005.

C. Plaisant, J. Rose, B. Yu, L. Auvil, M. G. Kirschenbaum, M. N. Smith, T. E. Clement, and G. Lord. Exploring erotics in emily dickinson's correspondence with text mining and visual interfaces. In *Proc. ACM Digital Libraries*, pages 141–150, Chapel Hill, NC, USA, 2006. ACM.

Y. Qu and G. W. Furnas. Model-driven formative evaluation of exploratory search: A study under a sensemaking framework. *Information Processing & Management*, 44(2):534–555, March 2008.

R. Rao, S. K. Card, W. Johnson, L. Klotz, and R. H. Trigg. Protofoil: storing and finding the information worker's paper documents in an electronic file cabinet. In *CHI*, pages 180–185. ACM, 1994.

P. Resnik, A. Elkiss, E. Lau, and H. Taylor. The web in theoretical linguistics research: Two case studies using the linguist's search engine. In *Proc. 31st Mtg. Berkeley Linguistics Society*, pages 265–276, 2005.

M. Ringel-Morris, E. Cutrell, S. Dumais, and E. Horvitz. Milestones in time: The value of landmarks in retrieving information from personal stores. In *In Proceedings of Interact 2003*, pages 184–191, 2003.

G. Robertson, M. Czerwinski, K. Larson, D. C. Robbins, D. Thiel, and M. Van Dantzich. Data mountain: using spatial memory for document management. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*, pages 153–162, 1998.

J. J. Rocchio. Relevance feedback in information retrieval. *SMART Retrieval System Experiments in Automatic Document Processing*, 1971.

G. Rockwell, S. G. Sinclair, S. Ruecker, and P. Organisciak. Ubiquitous text analysis. *Poetess Archive Journal*, 2(1), 2010.

G. Rockwell. What is text analysis, really? *Literary and Linguistic Computing*, 18(2):209 –219, June 2003.

G. Rockwell. What is text analysis, really? *Literary and linguistic computing*, 18(2):209–219, 2003.

D. M. Russell, M. J. Stefik, P. Pirolli, and S. K. Card. The cost structure of sensemaking. In *Proc. INTERACT and CHI*, CHI '93, 1993.

D. M. Russell, M. Slaney, Y. Qu, and M. Houston. Being literate with large document collections: Observational studies and cost structure tradeoffs. In *Proc. HICSS'06.*, volume 3, pages 55–55, 2006.

D. M. Russell, R. Jeffries, and L. Irani. Sensemaking for the rest of us. In *Sensemaking Workshop at CHI*, 2008.

G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information*

Processing & Management, 24(5):513–523, 1988.

G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. In *Readings in information retrieval*. Morgan Kaufmann, 1997.

N. Schneider, R. Hwa, P. Gianfortoni, D. Das, M. Heilman, A. W. Black, F. L. Crabbe, and N. A. Smith. Visualizing topical quotations over time to understand news discourse. 2010.

S. Schreibman and A. Hanlon. Determining value for digital humanities tools: Report on a survey of tool developers. 2010.

B. Shneiderman and C. Plaisant. Strategies for evaluating information visualization tools: multi-dimensional in-depth long-term case studies. In *Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization*, BELIV '06, pages 1–7, New York, NY, USA, 2006. ACM.

B. Shneiderman and C. Plaisant. *Designing The User Interface: Strategies for Effective Human-Computer Interaction, 5/e (Fifth Edition)*. Addison Wesley, 2010.

F. Smadja. Retrieving collocations from text: Xtract. *Computational linguistics*, 19(1):143–177, 1993.

J.-P. Soehn, H. Zinsmeister, and G. Rehm. Requirements of a user-friendly, general-purpose corpus query interface. *Sustainability of Language Resources and Tools for Natural Language Processing*, 6:27, 2008.

J. Stasko, C. Görg, and Z. Liu. Jigsaw: supporting investigative analysis through interactive visualization. *Information visualization*, 7(2):118–132, 2008.

C. Stolte, D. Tang, and P. Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Visualization and Computer Graphics*, 8(1):52–65, 2002.

S. Stone. Humanities scholars: Information needs and uses. *Journal of Documentation*, 38(4):292–313, December 1982.

J. Teevan, C. Alvarado, M. S. Ackerman, and D. R. Karger. The perfect search engine is not enough: a study of orienteering behavior in directed search. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '04, pages 415–422, New York, NY, USA, 2004. ACM.

J. W. Tukey. Exploratory data analysis. *Reading, MA*, 231, 1977.

J. W. Tukey. We need both exploratory and confirmatory. *The American Statistician*, 34(1):23–25, 1980.

M. Ullyot. Hamlet in the humanities lab. http://ullyot.ucalgaryblogs.ca/teaching/hamlet/, 2012. Accessed January 13th, 2012 2:00 PM.

N. Uramoto, H. Matsuzawa, T. Nagano, A. Murakami, H. Takeuchi, and K. Takeda. A text-mining system for knowledge discovery from biomedical documents. *IBM Systems Journal*, 43(3):516–533, 2004.

F. Van Ham, M. Wattenberg, and F. B. Viegas. Mapping text with phrase nets. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):1169–1176, 2009.

M. G. Van Kleek, M. Bernstein, K. Panovich, G. G. Vargas, D. R. Karger, and M. Schraefel. Note to self: examining personal information keeping in a lightweight note-taking tool. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1477–1480, New York, NY, USA, 2009. ACM.

R. Vuillemot, T. E. Clement, C. Plaisant, and A. Kumar. What's being said near 'Martha'? exploring name entities in literary text collections. In *IEEE VAST*, pages 107–114, 2009.

R. Watson-Boone. The information needs and habits of humanities scholars. *RQ*, 34(2):203–215, December 1994. ArticleType: research-article / Full publication date: WINTER 1994 / Copyright 1994 American Library Association.

M. Wattenberg and F. B. Viegas. The word tree, an interactive visual concordance. *IEEE Visualization and Computer Graphics*, 14(6):1221–1228, 2008.

K. E. Weick. *The social psychology of organizing.* Random House ; McGraw-Hill, New York, N.Y.; New York [etc.], 1979.

K. E. Weick. *Sensemaking in Organizations.* SAGE, May 1995.

R. W. White, G. Marchionini, and G. Muresan. Evaluating exploratory search systems: Introduction to special topic issue of information processing and management. *Information Processing & Management*, 44(2):433–436, March 2008.

B. M. Wildemuth and L. Freund. Assigning search tasks designed to elicit exploratory search behaviors. In *Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval*, page 4, 2012.

W. Wright, D. Schroh, P. Proulx, A. Skaburskis, and B. Cort. The sandbox for analysis: concepts and methods. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 801–810, New York, NY, USA, 2006. ACM. ACM ID: 1124890.

M. M. Zloof. Query by example. In *Proceedings of the May 19-22, 1975, national computer conference and exposition*, pages 431–438, 1975.

Appendix A

Implementation

WordSeer has a client-server architecture. The user-facing client (which runs in a browser) uses HTML5, CSS3,SVG, and JavaScript to render the interface and to send search queries and other user actions to the server. In response to requests triggered by user actions (searches, filters and visualizations), the server runs PHP scripts that read data from a MySQL database, process it, and send it back to the client. Like many other modern web applications, WordSeer's user interface is built with an MVC (model-view-controller) architecture.

To create a WordSeer instance, a user uploads a set of XML files. Then, a processing pipeline written in Java processes and indexes those text files in many ways, and stores the results in a MySQL database.

The latest version of the source code for the front-end application is available from `https://bitbucket.org/silverasm/wordseer/get/new-ui.zip`. It is also bundled with this dissertation as supporting material.

A.1 Third-Party Libraries

WordSeer relies on the following third-party libraries to help create rich user interactions and interactive visualizations. These need to be downloaded separately and placed in the `lib/`, which has placeholders for these libraries.

- To create interactive visualizations, we use D3 (`http://d3js.org/`) which is a JavaScript visualization library which enables the easy creation of data visualizations backed by data models that can be shared across visualizations.

- For the user interface elements, Model-View-Controller framework, and communicating with server-side scripts, we use ExtJS (`http://docs.sencha.com/ext-js/`), a JavaScript library for developing interactive user interfaces for web applications.

- We also use JQuery (`http://jquery.com/`), a general-purpose JavaScript library that makes accessing objects in the HTML document easier.

A.2 Web Application

The source code for the web application is in the `wordseer` subdirectory of the project source code. Throughout this section, we will make references to JavaScript classes named using the format `WordSeer.namespace.namespace.ClassName`. This class-naming system helps keep the code organized, and maps onto the directory structure of the code. The prefix `WordSeer` maps onto `wordseer/src/js`, the main directory for the user interface code, and for the rest of the file, the middle namespaces map onto subdirectories. The final name is the name of the file. For example, the class with this name can be found in the directory `wordseer/src/js/namespace/namespace/ClassName.js`.

A user accesses wordseer by navigating to `wordseer/instances/<instance>/index.html`, where `<instance>` is the name of the collection that WordSeer is using, and determines the database that is accessed when data is requested from the server.

When the user accesses this URL, JavaScript loads the WordSeer application defined in `src/js/new-ui.js`. This file tells the ExtJS application framework the models, views and controllers that WordSeer uses, and call the `WordSeer.controller.UserController`'s initialization methods. These create the sign-in page that the user sees. Upon signing in, the user sees the overview of the collection, encapsulated by the `WordSeer.view.windowing.Viewport.LandingPage` view. Interactions with the overview are controlled by the `WordSeer.controller.WindowingController`.

WordSeer's user interface software architecture is dictated by the following requirements:

- Keep track of the user's history of actions.
- Be able to show multiple items side-by-side.
- Give the user an overview of the contentes of a slice.
- Make the same slice viewable through many visualizations and search tools.

The main concepts in WordSeer are Slices and Views. Slices are sets of sentences. In principle, these can be arbitrary sets. In practice, they are intersections of searches and filters. They can also be collected individually and placed into Sets, which can be used as filters. Views are visualizations of the data in a slice. In WordSeer, Slices are represented by the Model Class `WordSeer.model.FormValues` and views are represented by the View WordSeer.view.windowing.viewport.LayoutPanel.

As a result, there are the following basic concepts:

- History items, FormValues, and History (Section A.2.1)
- Layout panels (Section A.2.2)
- Overviews (Section A.2.4)
- Widgets (Section A.2.5)

A.2.1 History

A history item represents a search or browsing action. History items are represented by the `WordSeer.model.HistoryItemModel`. Every history item has a `WordSeer.model.FormValues` object representing the search query, and a field for a `WordSeer.view.windowing.viewport.LayoutPanel` ID, which is the ID of the layout panel showing the data in this history item.

Examples of actions that are represented by history items are:

- Issuing a new search query,
- Filtering a set of results,
- Opening a new document,
- Browsing a collections of words, or
- Making a new comment on a document

History items are similar to items in a web browser's history. When the user goes to a new page, it gets added to their history. Similarly, when a user opens up a new search or visualization in WordSeer, a new history item gets added to their WordSeer history.

A `WordSeer.model.FormValues` (henceforth `FormValues` for short) object represents a slice in WordSeer. A FormValues object represents a slice as an intersection of searches and filters. It has a field `searches` which is a list of search queries, and a field `metadata` which is a list of metadata property value filters. If user-created sets are used, they are added to the metadata list.

The FormValues are serialized into JSON and sent to the server, which translates the various searches and filters into a set of sentences by performing SQL queries with the given parameters.

A.2.2 Layout Panels

The data in a slice is visualized through a View called a `WordSeer.view.windowing.viewport.LayoutPanel` (henceforth, `LayoutPanel` for short). The user interface starts with an overview showing the metadata categories in the underlying collection. Users can add panels by performing searches and filters.

The interactions around performing searches are governed by the Controller class `WordSeer.controller.SearchController`, and the interactions around creating and destroying layout panels are governed by the Controller class `WordSeer.controller.WindowingController`.

A layout panel displays the data in a slice. More precisely, a layout panel displays the values returned when making data requests based on a `HistoryItem`'s `FormValues`. A layout panel's state keeps track of two things: 1. The history item it is currently showing. 2. The history items it has shown in the past.

A layout panel has two types of sub-components: overviews and widgets. Overviews give the user a sense of the contents and metadata properties of the sentences in the slice, widgets visualize the data in the slice. Every `LayoutPanel` has three overviews: Co-Occurring Terms, Filters, and Sets. The Co-occurring Terms overview shows the most frequent nouns, verbs, adjectives, and phrases in the slice along with their sentence and counts. Filters shows the metadata properties of the sentences in the slice along with their sentence and document counts, and Sets shows the user-created sets of items, and their overlap with the current slice.

A.2.3 Caching

When a new search or filter is performed, a new `LayoutPanel` is opened, and a new `HistoryItem` and `FormValues` are created. To display the three different overviews and the visualization, a number of requests to the server are necessary. However, all these requests are for the same `LayoutPanel`, so they all retrieve data about the same underlying slice of sentences.

To avoid repeatedly re-computing this set of sentences for each request, the server caches the sentences that match a particular `FormValues` object. When a new search or filter is performed, the `SearchController` sends the serialized `FormValues` for that panel to the server URL `src/php/caching.php`. This script calculates the set of matching sentences and stores them in a table called `CACHED_FILTERED_SENTENCE_IDS`. This table has the columns `sentence_id`, `document_id`, and `query_id`. For each new `LayoutPanel` request, the `sentence_id`s that match the slice are stored in under a new `query_id` along with their document IDs. The `query_id` is returned as a response and is is added to the data stored in the `FormValues`. Further requests based on those `FormValue`s – such as the requests for the most frequent nouns, or the most frequent metadata properties, looks up the `query_id` and uses those sentences instead of re-calculating them.

The code automatically clears the sentences for query ID's that are more than 100 less the current ID.

A.2.4 Overviews

Each layout panel shows three types of overviews of the data in a slice. These are implemented as `WordSeer.view.menu.MenuOverlay`s and accessed through the buttons on the top of the `WordSeer.view.windowing.viewport.LayoutPanelLayoutPanel`.

Co-Occurring Terms – an overview of content

This overview shows the most frequent nouns, verbs, adjectives and phrases in the slice. The user accesses it through the 'Co-Occurring Terms' button.

The main views are `WordSeer.view.FrequentWordsList` (for the nouns, verbs, and adjectives), and the `WordSeer.view.PhrasesList` (for the phrases). The data for these views is stored in the `WordSeer.store.FrequentWordsStore` and the `WordSeer.store.`

PhrasesStore. The stores fetch instances of WordSeer.model.WordModel and WordSeer.model.PhraseModel from the URLs src/php/associated-words/get-frequent-words.php and src/php/phrases/get-phrases.php.

The interactions for this overview are managed by the WordSeer.controller.FrequentWordsController and WordSeer.controller.PhrasesController.

Filters – an overview of metadata properties

This overview shows the metadata properties of the sentences in the slice. The user accesses it through the 'Filters' button.

The main views are the WordSeer.view.metadata.facet.StringFacets which display overviews of the string-valued metadata properties, and the WordSeer.view.metadata.facets.RangeFacets that display overviews of number-valued properties in a bar chart. The data for these views is stored in the WordSeer.store.MetadataTreeStore belonging to each WordSeer.model.LayoutPanelModel. The store fetches instances of WordSeer.model.MetadataModel from the url src/php/grammaticalsearch/get-search-results.php?onlyMetadata=true.

The interactions for this overview are managed by the WordSeer.controller.MetadataController.

Sets – an overview of user-created collections

This overview shows the metadata properties of the sentences in the slice. The user accesses it through the 'Sets' button.

The overview shows lists of the type WordSeer.view.collections.SetsList which display lists of sets that the user has created. There are three views, one for each type of sets: Phrase Sets, Sentence Sets, and Document Sets, called WordSeer.view.collections.PhraseSetList SentenceSetList and DocumentSetList respectively. These views call upon data stored in three global stores: WordSeer.store.PhraseSetStore (for the Phrase Sets), WordSeer.store.SentenceSetStore (for the Sentence Sets) and WordSeer.store.DocumentSetStore (for the Document Sets). These stores keep track of the user-created sets for the entire user interface and are updated when the user makes changes.

These stores contain instances of WordSeer.model.PhraseSetModel, WordSeer.model.SentenceSetModel and WordSeer.model.DocumentSetModel respectively. They are Created, Read, Updated, and Deleted (CRUD) through the URL src/php/subsets/crud.php. These models all inherit basic create/read/update/delete functions from the base Set model WordSeer.model.SubsetModel.

The interactions for this overview are managed by the WordSeer.controller.SetsController.

In addition, to show the overlap between the user-created sets and the current slice, each of the SetLists has a link to the LayoutPanelModel's MetadataTreeStore, which has counts of how many sentences and documents in the current slice match different sets.

A.2.5 Widgets

A Widget is a view that displays any kind of content in the main portion of a `LayoutPanel` that the user sees and interacts with. WordSeer currently contains Widgets for the following functionalities:

- Visualizations
- Search
- Reading

The main Widget class is `WordSeer.view.widget.Widget`. All the widgets in the list below extend this Widget class. If you want to add a new visualization or search function to WordSeer, you'll usually want to start by making a new class that is a sub-class of a `WordSeer.view.widget.WidgetWidget`.

The `WordSeer.controller.SearchController` listens for search, filtering and browsing actions on widgets and responds accordingly.

Visualizations

Visualization widgets illustrate different data about words, sentences, and grammatical relationships within collections of data. WordSeer contains the following visualization widgets.

- `WordSeer.view.widget.WordTreeWidget` – view and browse the contexts in which a word or phrase occurs. Interactions are listened for in the `WordSeer.controller.WordTreeController`
- `WordSeer.view.widget.WordFrequencies` – view and filter the frequencies of words that enter into grammatical relationships with each other. Controlled by the `WordSeer.controller.WordFrequenciesController`

For example, the WordTree widget visualizes the contexts that surround every instance of a certain word. For the same word, the Search visualization would list all the sentences in which that word occurs, and the Bar Charts visualization draws bar charts showing the frequencies of other words that enter into grammatical relationships with that word.

Search

- `WordSeer.view.widget.SentenceListWidget` – shows a list of the sentences in the slice. The view is composed of a single `WordSeer.view.sentence.SentenceList` and is controlled by the `WordSeer.controller.SentenceListController`.
- `WordSeer.view.widget.SearchWidget` – Shows the sentences that match a keyword or grammatical search as well as (for grammatical searches) bar charts of the frequencies of the matching words. The view is composed of two items: a `WordSeer.view.`

`sentence.SentenceList` and a `WordSeer.view.visualize.barcharts.BarCharts`, both are controlled by the `WordSeer.controller.SentenceListController`.

- `WordSeer.view.widget.DocumentBrowserWidgetDocumentBrowser` – Shows a list of documents in the slice. Shows a single `WordSeer.view.document.DocumentViewer`, and is controlled by the `WordSeer.view.controller.DocumentsController`.

Reading

`WordSeer.view.widget.DocumentViewerWidget` – Shows a single document. If the view was opened up to show a sentence in context, scrolls to that sentence. The view is composed of a single `WordSeer.view.DocumentViewer` and is controlled by the `WordSeer.controller.DocumentsController`.

A.3 Processing Pipeline

The latest version of the back-end processing pipeline is available at `https://bitbucket.org/silverasm/wordseerbackend/get/default.zip`. It is also bundled with this dissertation as supplementary material.

It has the following files:

- wordseerbackend.jar – the main processing program
- lib/ – a directory containing some files that the program needs
- src/ – the source code.
- example/ – an example data directory
- example/structure.json – the JSON file that gives the structure of the data.
- example/articles/ – three short XML-formatted 'articles' containing some text and metadata.
- ibiblio_shakespeare – another example, all of Shakespeare's plays
- ibliblio_shakespeare/structure.json – the JSON file that gives the structure of the shakespeare XML files.

A.3.1 Input Files

WordSeer's input is a set of XML or HTML-formatted files in a directory. These files do not have to be pure XML or HTML – they do not need to have a DTD or namespace definitions. They simply need to have a hierarchical arrangement of tags. The source code includes examples of simple XML files in the directory `wordseerbackend/example/articles`.

The directory has a short set of 'articles' that each have an author, a title, date, a post number, and a few tags. An example is shown in Listing A.1.

```
<article>
  <author> Mary Smith </author>
  <date>2013-05-12</date>
  <title> My life and other stories </title>
  <number>54</number>
 <tags>
   <tag>Diary</tag>
   <tag>May</tag>
  </tag>
  <text>
     This is the story of my life....
 </text>
</article>
```

Listing A.1: An example of an input data file

JSON Structure File

An input into the data processing pipeline in a JSON file that describes the structure of the XML files. It defines which XML tags contain text, and which contain metadata about the text.

The file must contain a single JSON object with the properties shown in Listings A.2 – A.4. The top level structure is a Document. Documents can have sub-structures (which can have sub-structures and so on), and metadata.

```
{
   structureName: "document",
   xpaths: ["play"], // .. a list of xpaths to the XML tags you want to
      treat as documents
   metadata: [ ... a list of Metadata definitions...], // any metadata
      properties that belong to the document
   units: [ ... a list of Sentence or Unit definitions] // the sub-
      structures within the document
}
```

Listing A.2: The JSON file structure.

Units can have units and metadata.

```
{
   structureName =  "act",  // This can be anything other than 'sentence'.
                            // If you put 'sentence', it means there are
                               no more sub-units.
   belongsTo: "document"  // a string, the parent structure name,
   xpaths: ["./act/"] // a list of xpaths (relative to the parent unit)
      where this structure can be found
   metadata: [.. a list of Metadata definitions]
}
```

Listing A.3: Defining sub-units in a document.

Units can have metadata, which are properties and values that attach to them. For example, every Act in a play has a title, every speech in a play has a speaker. These would be the properties ‘act title’ and ‘speaker’, which would have different property values.

```
{
   propertyName: "speaker" // the name of the property, no spaces allowed
   xpaths: ["./speaker/text()"] // a list of xpaths containing the value
      of the property
   type: "string"  // Other options are 'number', and 'date_[date format]'
   isCategory: true // Do you want to be able to filter based on this
      information?
   valueIsDisplayed: true // Whether to display this property/value when
      you're viewing the document
   nameIsDisplayed: false // Whether to display the name of the property (
      ie. speaker: Cleo, vs just Cleo),
   displayName: "Speaker" // A pretty name for the property.
}
```

Listing A.4: Defining metadata properties of units.

For a list of date formats, see the D3 date format list: https://github.com/mbostock/d3/wiki/Time-Formatting.

A.3.2 Creating the Database

Before the pipeline can be run, it requires the user to set up the appropriate permissions, as shown below.

```
CREATE USER wordseer@localhost;
SET PASSWORD FOR wordseer@localhost = PASSSWORD(‘wordseer’);
GRANT ALL PRIVILEGES ON  ‘ws_’.* to wordseer@localhost;
```

The processing pipeline is packaged into the `.jar` file `wordseerbackend.jar`. It is run with the following command-line arguments:

```
 -i,--instance <arg>    The short (20-characters or fewer) label to use
                        for this WordSeer instance.
 -d,--data <arg>        The path to the XML data files
 -s,--structure <arg>   The path to the JSON file explaining the structure
                        of the XML files
 -h,--help              Print this message
 -r,--reset             Clear database and restart processing
```

The instance name is a short (20-characters or fewer) name that is used to create a new database and a new front-end access URL.

The program entry point for `wordseerbackend.jar` is the class `edu.berkeley.cs.wordseer.CollectionProcessor`'s `main` method. The pipeline reads the files one by one, uses the JSON file to extract the structures within them, runs the Stanford Parser (available from `http://nlp.stanford.edu` and bundled with this code) on the sentences to extract the words, parts of speech, and grammatical relationships, and records this information in the database.

For example, to process the test collection bundled with the code, the user would execute the following command line from the `wordseerbackend` directory:

```
java -jar -mx2g wordseerbackend.jar --data=example/articles
  --structure=example/structure.json --instance=test
```

To run on a medium-size collection, such as the complete works of Shakespeare, with 2GB of memory, this process takes 5 hours on a 2010 MacBook Pro laptop. Larger collections will take longer.

A.3.3 Database Structure

There are nine content tables and four cross-reference tables that store WordSeer's data. The content tables are:

- `UNIT`,
- `DOCUMENT`,
- `SENTENCE`,
- `WORD`,
- `SEQUENCE`,
- `METADATA`,
- `SET`,
- `RELATIONSHIP`, and
- `DEPENDENCY`

Each of these are responsible for assigning unique ID's to the document sub-structures, sentences, words, n-word sequences, distinct property values, and user-created groups that appear in the text. These tables are linked to each other by five cross-reference tables, which are

- `SENTENCE_IN_UNIT`,
- `WORD_IN_SENTENCE`,

- `SEQUENCE_IN_SENTENCE`,
- `DEPENDENCY_IN_SENTENCE`, and
- `SET_CONTENTS`

The `UNIT` table is the main book-keeper for the structures within the text collection. During database construction, XML input documents are processed one by one, and the `UNIT` table assigns a unique `unit_id` to each document and to each sub-unit within the document. Sub-units are extracted from XML tags that occur within the main document's tag. The table also records each unit's `type` ('document', 'sentence', or a custom value such as 'Act' or 'Scene' for Shakespeare's plays) and `number` (the order of occurrence).

In WordSeer, all documents must contain sentences (which are extracted from plain text inside XML tags), but documents can optionally contain sub-units which can hierarchically contain other sub-units and then sentences. This hierarchy is how WordSeer represents the different acts within a play, the different scenes within an act, and the different speeches within a scene in a single Shakespeare play. Hierarchy is stored using a `parent_id` column. For every unit, the parent units that encompass it can be found by recursively traversing the `parent_id` column. Similarly, the sub-units of a particular unit can be found by recursively querying for units that have that `unit_id` as their `parent_id`. The `parent_id`s of documents are null, because they are top-level units.

The `DOCUMENT` table stores the title and source XML file name of each document under the `unit_id` that was assigned by the `UNIT` table. This `unit_id` is stored under the `document_id` column.

The `SENTENCE` table stores the raw text of each sentence under the `unit_id` that was assigned by the `UNIT` table. This `unit_id` is stored under the `sentence_id` column. The text is used whenever a sentence needs to be displayed in the user interface, such as in search results. The document to which the sentence belongs is also recorded under the `document_id` column of this table.

The `SENTENCE_IN_UNIT` table links the `UNIT` and `SENTENCE` tables. This link is not strictly necessary, because the `SENTENCE` table already uses the `unit_id`s from the `UNIT` table, and as previously mentioned, it is possible to compute all the units that contain a given sentence by making recursive calls to the database. However, such calls are time consuming and introduce an inconvenient overhead at query time. The `SENTENCE_IN_UNIT` table therefore records all the larger `unit_id`s that contain every `sentence_id` during pre-processing.

The `METADATA` table stores the metadata property values associated with the different units. It has the fields `unit_id`, `property_name` and `property_value`. For example, if a particular document with `unit_id = 3` has the title 'Hamlet: Prince of Denmark', then the `METADATA` table row representing the information would be:

The `WORD` table assigns each word a distinct `word_id`. It distinguishes between different parts of speech. For example, the verb sense of `cost` ('It cost me a dollar.') and the noun sense of `cost` ('What is the cost?') would have different `word_ids`. The table also stores the aggregate `sentence_count` and `document_count` for each word. The `WORD_IN_SENTENCE`

unit_id	property_name	property_value
3	'title'	'Hamlet: Prince of Denmark'

Table A.1: An example of a table row in the METADATA table storing the title of a play.

table links each word_id with all the sentence_ids in which it appears, and records the surface capitalization, spaces following and preceding, and position within the sentence.

Similarly, the SEQUENCE table assigns each 2-, 3-, and 4-word sequence a distinct sequence_id. For each phrase, variants with and without stop words, and with and without lemmatization are also computed and assigned different sequence_ids. This table also stores the aggregate sentence_count and document_count for each sequence.

The SEQUENCE_IN_SENTENCE table links each sequence_id with all the sentence_ids in which it appears, and records the starting position of the sequence within the sentence.

As the text is processed, the parsers extract the grammatical relationships between the words in each sentence. The RELATIONSHIP table assigns each different type of grammatical relationship a distinct relation_id and the DEPENDENCY table stores each unique combination of two words and a relationship under a distinct dependency_id. Grammatical relationships have two words, called the governor and the dependent. For example, in the phrase " left or right" the relationship is the preposition 'or' (represented as prep_or), the governor is 'left' and the dependent is 'right'. The representation is prep_or(left, right), which is distinct from prep_or(right, left), which would be the extracted from phrase "right or left". The DEPENDENCY table thus has distinct columns for gov_id and dep_id, which store the word_ids of the governor and dependent words respectively. The columns of the table representing the dependency are dependency_id, relation_id, gov_id and dep_id. This table also stores the aggregate sentence_count and document_count for each dependency.

The DEPENDENCY_IN_SENTENCE table links each dependency_id with all the sentence_ids in which it appears, and records the positions of the gov_id word and the dep_id word in the sentence under the columns gov_index and dep_index.

Finally, the SET table gives a name, type ('phrase set', 'sentence set' or 'document set') and unique set_id to each user-created set. Sets are hierarchical, so there is a parent_set_id column. The contents of the set are recorded in the SET_CONTENTS table, which has the columns set_id and content_id. In the case of sentence and document sets, the content_ids are unit_ids. In the case of phrase sets, they are phrase_ids.

A.3.4 Linking the Web App to the Database

To get a new front-end instance of WordSeer, the user can clone the repository https://bitbucket.org/silverasm/wordseer, or download the source code from https://bitbucket.org/silverasm/wordseer/get/new-ui.zip. It is also bundled with this dissertation as supplementary material.

Since WordSeer is a web application, the user must unzip the code and place it in their web server root.

To create a new user interface access point for the newly-created instance, the user must copy and paste one of the existing directories in `wordseer/instances/` (e.g. instances/acm) and rename it to the same `<instance name>` that was chosen during processing, and change the first line in the config.php file to have that instance name as well.

Then, `http://localhost/instances/instance_name/` will take the user to a new front-end instance of WordSeer.