Eurographics / IEEE Symposium on Visualization 2011 (EuroVis 2011)
H. Hauser, H. Pfister, and J. J. van Wijk
(Guest Editors)

Volume 30 (2011), Number 3

Exploring Collections of Tagged Text for Literary Scholarship

M. Correll[1], M. Witmore[2], M. Gleicher[1]

[1]University of Wisconsin-Madison Department of Computer Sciences
[2]University of Wisconsin-Madison Department of English

Abstract

Modern literary scholars must combine access to vast collections of text with the traditional close analysis of their field. In this paper, we discuss the design and development of tools to support this work. Based on analysis of the needs of literary scholars, we constructed a suite of visualization tools for the analysis of large collections of tagged text (i.e. text where one or more words have been annotated as belonging to a specific category). These tools unite the aspects of the scholars' work: large scale overview tools help to identify corpus-wide statistical patterns while fine scale analysis tools assist in finding specific details that support these observations. We designed visual tools that support and integrate these levels of analysis. The result is the first tool suite that can support the multi-level text analysis performed by scholars, combining standard visual elements with novel methods for selecting individual texts and identifying represenative passages in them.

Categories and Subject Descriptors (according to ACM CCS): J.5 [Computer Applications]: Arts and Humanities—Literature

1. Introduction

Advances in digital storage, curation, and collaboration are beginning to produce seas of data humanities scholars are currently ill-equipped to handle [Ell05]. Typically when faced with a situation such as this, visualization techniques are applied to aid scholars in making sense of large-scale data. Unfortunately, humanities scholars are not well served by most existing information visualization techniques. Humanities argumentation is traditionally done by reference to specific exemplars. For example, while a chart showing statistical properties in a large dataset might be sufficient proof for an argument in the sciences, literary arguments prefer to reference and analyze specific text passages.

In this paper we describe the creation of a suite of visualization tools designed to meet the needs of literary scholars as they incorporate computational tools that operate over large text corpora, specifically those using tagging schemata to perform "algorithmic criticism." Through a careful design process we have developed a solution tailored to the needs of these scholars. This solution allows for the discovery and examination of patterns of tagged text at the corpus-wide level, but also supports the process of passage analysis for grounding digitally inspired arguments in specific human artifacts.

Our domain collaborators needed to analyze large corpora of texts that had been tagged based on rhetorical content of specifics words and phrases. To assist them in this effort we created two tools: the CorpusSeparator for visualizing corpus-wide rhetorical patterns and trend-specific selections, and the TextViewer for visualizing and selecting salient passages of specific texts. In concert, these tools allow literary scholars to very quickly formulate a hypothesis about a corpus and then provide evidence for this hypothesis by drilling down to the specific passages of text.

1.1. Problem Overview

Rhetorical analysis is a form of literary scholarship that seeks to understand the ways that language is used, rather than the message that is being conveyed. Often, scholars aim for analysis that is *distanced* from the meaning, so that their findings are generalizable beyond the particular content of a text. At the same time, scholars prefer arguments with specific examples of passages that support their analysis. Such arguments are made by *close* reading: careful analysis of specific passages of text, with an eye toward exemplifying some more global pattern.

Performing rhetorical analysis at a large scale, e.g. on

significant corpora of texts, offers the possibility of finding trends and patterns in language usage, for example, to resolve questions of authorship, to see "signatures" of structure in different genres, or to observe the historical development of language. Unfortunately, the traditional approach to analysis (close reading of specific passages) does not scale well due to the sheer amount of text that must be closely read, especially if the scholar attempts to maintain distance. As scanning, curation, and sharing efforts provide scholars access to larger sets of text to analyze, new analysis methods must also be developed that scale appropriately.

Algorithmic criticism is an emerging method for applying rhetorical analysis at a large scale [Ram03]. The approach exploits the disconnect between deep semantic interpretation of text and the surface features of natural language: by focusing on simple, low-level properties that can be discerned algorithmically, these "prosthetic readings" of texts are necessarily distanced. Statistical analysis of low-level properties are viewed to identify patterns and trends. This mode of analysis is sometimes referred to as "iterative criticism" to indicate the need to reincorporate feedback into the process, which emphasizes the need for a fluent workflow that enables iteration.

One form of algorithmic criticism focuses on the roles of words. Thus, each word is replaced by a "tag" corresponding to a rhetorical or semantic category. Automated or semi-automated tagging systems use algorithmically simple means, such as regular expressions, to determine the tags for each word. A text (or a portion of one) can be represented as a vector in n-dimensional space, where there are n different tag types. Each element of the vector is the count of the corresponding tag. Statistical tools such as Principal Component Analysis can then be used to see where different categories of a corpus fall within the n-dimensional space.

While standard text taggers are available and even preferred to ensure consistency in the tag schema, there are few tools to help analyze the tagged data. No existing tools (see below) had been created to support the unique scholarly process that must ultimately connect the distanced reading with the close passage analysis. Scholars have been using standard statistical packages to view corpus-wide data, but this is not a panacea. Even if an interesting pattern does emerge (for instance, one author's rhetorical style results in a very different distribution of tag counts as compared to the corpus at large), providing passages that present literary evidence of this pattern requires manually tagging, and sampling many texts, then looking through interesting passages by hand until a rhetorical pattern is pinpointed. This approach does not scale to large corpora, nor is it particularly efficient.

1.2. Solution Overview

The primary insights from our work are an understanding of how the methods of humanities scholars lead to unique needs for their tools. While our study has been specific to algorithmic criticism, we believe these needs, particularly the ability to connect large scale statistics to specific exemplars, exist across many forms of humanities scholarship and beyond. The primary novelty in our system is its overall design that assembles components to support scholars' workflow. However, in realizing this approach, we have developed a number of novel components:

- On-the-fly thresholding and filtering for visualization and re-computation of statistical analysis on text corpora.
- At a glance information about distribution of tags across a corpus, with details on demand.
- Visual links between loadings on principal components and individual passages of tagged text.
- Focus+context techniques for selection of passages.

Contributions: Our work contributes an example of a case study, showing how a process of following task analysis with iterative development can lead to interesting results. We provide specific insights on the work of literary scholarship, and how scholars' needs create demands on tools. The design we propose combines standard visualization techniques with several novel ones (see above) to address these needs Finally, the actual systems themselves, CorpusSeparator and TextViewer, are a contribution as they have shown immediate utility for our collaborators.

2. Related Work

Many existing visualization tools for use with large text corpora are intended mainly for topic clustering and curation. While these tools can generate compelling results for these tasks, they are often abstracted from the actual content of the text, relying on "bag of words" vector representations of texts to generate their results, as with the common terrain or starfield corpus visualization techniques [WTP*95]. Thus while important properties of the text are visualized, there is no clear mapping from these properties to specific text, even in tools such as Docuburst that are otherwise useful for visualizing how different texts differ [CCP09]. Some visualizations, such as implicit shapes [RES98], do not clearly connect visual features to either text or concept, preventing even surface level analysis of properties. This disconnection is inappropriate for our domain. In general, corpus-level overview tools that rely on important words to create groups are not appropriate for specifically literary tasks, even tools such as ThemeRiver [HHN00] that otherwise do a good job of displaying temporal changes in a corpus. Visualizations combining tag clouds with other visualization paradigms provide some connection to the text, but this is still unsuited to passage analysis [CVW09] [CSL*10].

Some tools combine both high level corpus overviews with text views, such as Jigsaw [SGS08]. The generality of these systems creates drawbacks for scholarly applications, including high learning curves and demands for user creation and labeling of salient features. They also do not help

with identifying paradigmatic passages. One might use these tools to be able to see how many times a certain word or pattern appears in a text, but searching by hand through all of these "hits" without the ability to set thresholds of relevance does not scale well.

One method of computer-aided text analysis relies on multiple tools, some of which incorporate views of the text per se. Using a suite of different visualization tools it is possible to conduct intricate text analysis [CPV09]. These tools still rely on specific words, although stemming is often used to remove the effects of changes in tense or number. Rhetorical patterns (and other literary methods of analysis) are concerned less with word choice but more with word meaning, making existing tool suites difficult to adapt to problems of rhetorical analysis.

Wanting to connect visualization of a large corpus with small sections of text is a problem encountered in the wider software visualization community [BE96]. The analogy is not perfect, however, as softVis typically deals with issues such as version control and collaboration that do not have clear parallels in literary analysis, which usually focuses on the final product rather than the production process.

3. Design Setting

Our domain collaborators are literary scholars operating in the DigHum (Digital Humanities) working group at the University of Wisconsin-Madison. We observed their current tagged text analysis process in person, as well as as described in their published works. They had access to a number of corpora of English literary works. Their avenues of research included pinpointing the development of specific genres, differentiating rhetorical style between genres of Renaissance dramas, and using rhetorical "signatures" to back up grounded claims of authorship [Wit10]. This methodology is already successful in identifying how "micro" linguistic level activities, such as word choice, noun forms, syntax, or article and pronoun use, are connected to higher level phenomena, like authorship, genre, and meaning.

Their efforts make use of the Docuscope tagging schema, which categorizes millions of regular expressions (usually on the scale of one or two words) into roughly 100 different tags based on different rhetorical categories (e.g "FirstPerson" or "DirectAddress" language) [CK01]. Since Docuscope was meant to tag modern English speech, modernization techniques were used on the largely antiquated English corpora to increase the percentage of raw text being tagged.

The existing workflow relied on Principal Component Analysis to analyze clusters in a particular corpus. Texts in the corpus were divided into multiple subsections based on an empirically justified word length window. The Principal Components that were sufficient to isolate a sub-corpus of interest were saved, and the subsections that scored particularly well on these axes were examined by hand using traditional literary methods of passage analysis, combined with a subjective appraisal of weights on tags in each Principal Component. This workflow generated results that have been published in mainstream Humanities journals [HW10].

Of note in this workflow is that it differs from traditional (and well-studied) text clustering problems in that it is not about finding clusters in the text (since these clusters are already known, e.g. the genre of texts are known in advance), but rather showing how known a priori groupings are ultimately reflected in low-level properties in specific passages of text. The existing workflow was not optimized for this purpose, and required switching between statistical packages like JMP[†], a proprietary Docuscope reader program, and raw analysis of large matrices of data. This approach was already difficult at the scale of the less than 100 plays of Shakespeare, and could not even in principle scale to larger corpora (such as Google Books' millions of texts).

Another problem not supported by manual methods was that it was often possible to find interesting passages (say by using the PCA methods in JMP), but generating "slices" of the texts for this purpose was somewhat arbitrary, generating high frequency, noisy results in PCA space. To minimize this issue our collaborators would use overlapping sliding windows of text a few hundred words long. Changing this window size had the effect of reducing the impact of outlier sections of text, but it was difficult to perform this sort of analysis on the fly without having to recreate the text slices.

3.1. Requirements Analysis

In order to improve (and hopefully supplant) the existing workflow, we compiled an initial list of requirements:

- The tools must scale to corpora with sizes from just a few items to potentially thousands of entries, where each item could be as small as a single passage or as large as an entire novel.
- In order for the results generated to be of use from an argumentation standpoint, there must be a direct link between a visualized item and specific text.

After constructing an initial prototype (see below) we realized that our collaborators had already "learned" the various idiosyncrasies of the Docuscope software, and were reluctant to modify existing knowledge about color mappings or file formats. In addition, other collaborators were reluctant to use new tools at all if they only relied on the Docuscope tagging scheme. Thus we modified our list to include two additional, somewhat contradictory requirements:

- Since the scholars were used to operating with the original Docuscope software, our software ought to mesh well with the esthetics and structure of Docuscope.

[†] A proprietary statistical analysis software suite, www.jmp.com

- Since not all of the scholars thought the Docuscope scheme was equally useful, our software should generalize to different tagging schemata.

In addition, further analysis of the workflow clarified the existing taks, which was more than analyzing tag counts per se. In particular, most of the existing insight-generating work seemed to follow this model:

1. Identify a useful weighting of each tag that distinguishes certain groups of texts from others.
2. Using this weighting, find important texts that typify a group, or are outliers from a group.
3. Within these texts, find important passages that score in important ranges using the current weighting scheme (e.g. passages that "explain" a text's location).

Existing tools were insufficient for this model, and there was no automated way to accomplish the task at different levels of scale. It was not enough to augment this workflow with better tools, since for the most part these tools did not exist. Thus we added a final item to our list of requirements:

- Our tools must either supplant or provide a superset of abilities found in the current workflow, i.e. they must provide the ability to find a salient weighting of tags, find salient texts based on this weighting, and lastly find salient passages within these texts.

4. Design Rationale

For the design of our tools, we followed an iterative model where we would conduct ethnographic observation of workflow, prototype initial tools, and then refine existing tools or create new prototypes based on feedback. This cycle was repeated several times as our understanding of the needs of our collaborators was deepened.

4.1. Initial Prototype

Our initial efforts focused on a specific facet of the existing workflow: our collaborators did not have a good way of visualizing differences in tag counts across the entire corpus other than the manual analysis of a large data matrix. The initial CorpusViewer tool was meant to be a simple prototype that allowed users to visualize outliers between groups.

A stacked bar chart with multiple levels of detail, aggregation, and mouseover annotation was prototyped and released to the domain collaborators. Since Docuscope has more tags than one could conceivably use for a mutually-distinguishable color palette, a palette of low-saturation pastels with alternating bands of small hue difference was used. This proved to be difficult for collaborators used to the rainbow spectrum scheme of the Docuscope program, and so we gave users the option of using our banded saturation palette or using the original rainbow Docuscope palette. The ability to keep the old palette eased the transitions, despite the known deficiencies of the rainbow color scheme [BT07].

While the CorpusViewer was used for gross analysis of tag patterns, we had hoped to supplant what we saw as an inefficient workflow, as well as allow previously impossible capabilities. In particular our domain collaborators had no ready way of visualizing what different tag patterns meant at the textual level. We began prototyping tools that could interface with both a binary list of "interesting" tags generated from the CorpusViewer, as well as the axes in tag space generated by PCA or other embeddings (such as other MDS methods, or a distance from a hyperplane generated by a Support Vector Machine).

4.2. CorpusSeparator

The CorpusSeparator [Fig. 1] is meant to provide an overview of the entire corpus, separating out clusters or patterns of interest at the level of specific texts. Each item in a corpus is represented as a vector of tag counts. After normalization of each item to control for differences in item length and tag density, Principal Component Analysis is used to collapse this potentially high dimensional space into a linear subspace. Users observe the projection of the corpus into a two dimensional subspace of Principal Component space, find clusters of texts that are separated from other groups, and use the relevant Principal Component axes to generate an ad-hoc saliency metric that assigns an importance rating to each tag. Individual texts that are particularly good exemplars of their group, or are particularly troubling outliers, can then be examined using the TextViewer (see below) with the aide of the chosen saliency metric. Problems of occlusion, distance gauging, and general complexity limit the user to viewing only two dimensions of the space at a time, but users can select any arbitrary two dimensional subspace.

One capability that was currently lacking from the existing workflow was to be able to filter out outliers, both in terms of tags and texts. Certain tags were rare enough that they did not noticeably contribute to understanding the corpus, and certain texts were poorly tagged for one reason or another (poor modernization for instance) and so not representative of a group in the corpus. Other texts or tags that would appear to be outliers were actually useful for analysis; filtering decisions had to be made a posteriori. To facilitate this filtering we included "cards" for each tag showing the distribution of normalized tag counts across the entire corpus. Originally users would scroll across each card, setting thresholds for inclusion or exclusion of texts, or binary choices for inclusion or exclusion of the entire tag in the PCA. This approach was useful and allowed a lot of fine control, but did not scale to Docuscope's over 100 tags (and thus over 100 cards). An accordion view was chosen, allowing quick navigation of tag distributions [Fig. 3]. While this accordion view does not scale fully to the level of hundreds of tags, it works well for the number of tags in Docuscope. Since each tag type is meant to encode a different semantic category, the workflow used by our collaborators would

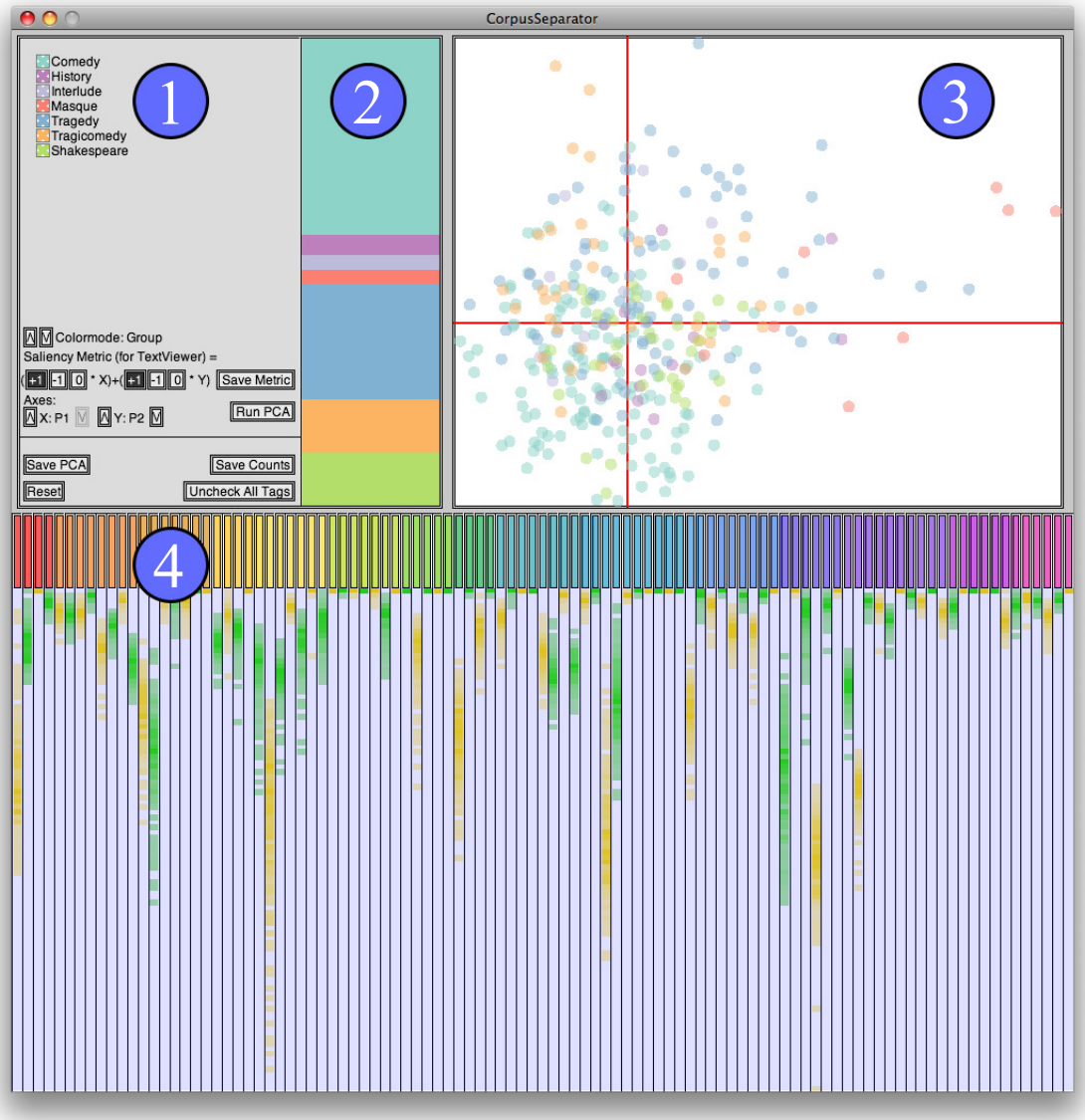

Figure 1: *A view of the CorpusSeparator tool after importing a corpus of 300 16th and 17th century English plays.*
1. The upper left panel displays filtering information and principal component options, as well as options to save a weighted sum of tags as a combination of one or two principal components for later use by the TextViewer. One can scalar multiply each axis' principal component by 0, -1, or +1. Both multiplied axes are then added together to generate an ad-hoc saliency metric.
2. The composition of the corpus (in terms of blocks of different groups). Users can color based on a priori groups like genre, but also metadata such as composition year or author.
3. The upper right panel shows the projection of each text as an n-dimensional vector of tag counts into a two dimensional subspace based on Principal Component Analysis. Conventional axes, in red, orient the user.
4. The lower panel is an accordion view of all tags (detailed in Figure 3), with the distribution of normalized tag counts per text represented as bands of color.

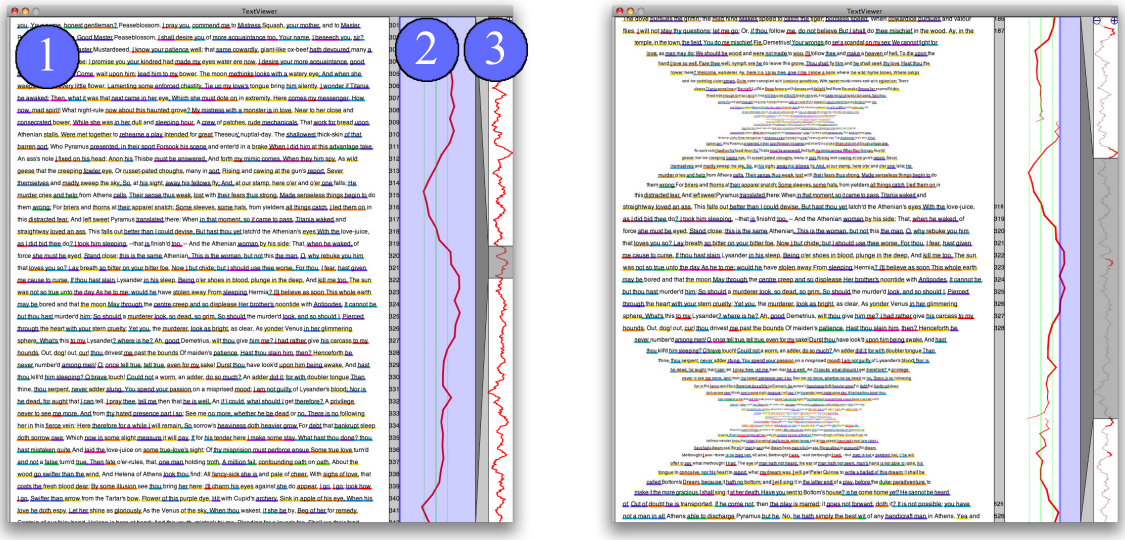

Figure 2: *Two views of the TextViewer tool on Shakespeare's "A Midsummer Night's Dream." The left view shows an initial view of the text, whereas the right view shows the same location in the text after a threshold has been set. Lines with scores at or above the threshold are in focus. This creates bubbles of salient passages that are in focus, while the rest of the text recedes.*
1. The left panel of each image is the raw text, with colored underlines for text that has been tagged.
2. The larger vertical line graph (the local graph) shows the score of a window of text centered on each visible line based on a weight on tags generated in the CorpusSeparator or by hand.
3. The smaller, rightmost graph is the global graph of the scores for the entire play. The gray rectangle of the global graph represents the text currently in view. Note that much more of the text is within the window once a threshold has been set.

not support a tag encoding scheme much larger than Docuscope's even if one did exist.

Using the CorpusViewer, our collaborators were able to develop richer intuitions about the dimensionality reduction process, notice outliers, and see the results of moving to different embedding subspaces on the fly. While other PCA visualization tools exist, many relying on the same scatterplot presentation, our tool unites the ability to quickly filter outliers, gain statistical information about the distributions of values in the original dimensions of the space, and switch to new PCA embeddings.

4.3. TextViewer

The TextViewer [Fig. 2] is meant to tie large scale tag patterns to specific passages of text. It takes in two arguments: a weighted list of tags, and a specific tagged text. This weighted list can be generated using PCA (e.g. using the CorpusSeparator), or can simply be a binary list of inclusion or exclusion of tags. Text is then rendered with colored underlines, with the saturation value of the underline corresponding to the absolute value of the tag weight. These underlines (especially with tensely tagged documents) are distracting to an untrained reader, but are representative of the output of the original Docuscope program and thus our collaborators were acclimatized to reading documents in this form. The goal of a TextViewer session is to visualize how a tag pattern is realized in a particular text, and more specifically pick out and perform close analysis on passages that are exemplars of these patterns. A text might be quite large, but the number of salient passages quite small, and the requirements of saliency fluid. A user would need an overview of the entire text, but also the ability to quickly move between important passages. A focus+context view was then the natural choice [SSTR93]. The raw count (or in the PCA case, the weighted sum of raw counts) of tags in a particular window of text is a one to one mapping of a text to a signal. By setting thresholds of importance, the user creates foci on sections of text that lie within the desired threshold. Virtual lenses placed on these foci causes these passages to come forward, while less important passages recede.

The process of direct analysis in the existing workflow differed in several key areas from the "natural" reading environment, and so we had to adapt TextViewer to more closely resemble this artificial method of reading. Our domain collaborators would strip out information such as line breaks and stage directions from texts, since those vary from edition to edition (and as such do not have explicit, invariant rhetorical content). The lines in the TextViewer are thus not natural units of a text (although it is possible to set canonical

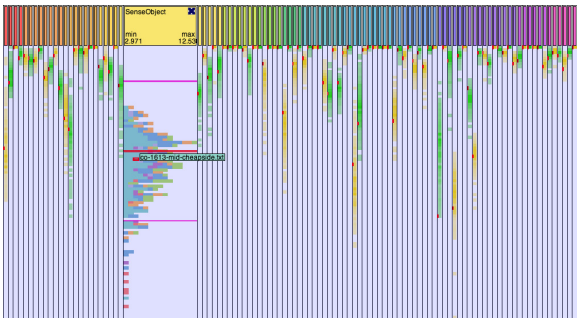

Figure 3: *An example of the accordion view of per tag "cards" in CorpusSeparator. When a mouse is over a card, it expands into a cumulative distribution histogram of a particular tag. The count of a particular tag's appearance in each text is represented by a block. Lines for the mean and the standard deviation in both directions are represented by red lines. To exclude outliers users can drag thresholds up or down, removing texts outside of the threshold from consideration in any future Principal Component Analysis. Collapsed cards represent distribution as saturation values of blocks of color, allowing tags with outliers to be seen at a glance without having to expand every card.*

line breaks). This conflicted with our desire for there to be a one to one relationship between a line of text and a point on the local graph. We also did not want to have to normalize to account for lines with shorter words having more tags (and thus higher absolute values of scores). We decided therefore to treat the score associated with a line as being the score of a window of words centered at that line (e.g. if the window size is 256 words and the line is 56 words, the previous 100 and next 100 words are scored and added to the total). These windows naturally overlap with the windows of other nearby lines. Thus larger window sizes perform a convolution of the high-frequency saliency signal into a smoother, more manageable one. For some texts larger or smaller window sizes are the natural unit of division. Since the "best" window size for a particular text is not known a priori, we gave users the option to adjust this window size on the fly.

Focus+context interfaces in other domains benefit in that the minimum size of an item not in focus may be quite small: one might still glean information from e.g. a tree if sections of it are compressed to one or two pixels [MGT*03]. Unfortunately, in the text domain words must be of a minimum size in order to be legible. Originally, text not within a focus was greeked to provide context information about a passage's location with regards to the rest of the text. Since this greeked text was not useful for the task at hand, we removed this functionality and instead had text recede until it was no longer legible, and then removed the out of focus text entirely, creating "bubbles" of text with discontinuities between them. The user thus still has local context within the passage without having to consider large sections of greeked, meaningless text. Global context is also provided by a small "global graph" overview to the right of the text, with a colored region indicating how much of the document is currently in focus.

Another issue with a focus+context interface in the realm of text is that sentences may begin or conclude in an area currently out of focus, preventing the full context of a passage from being known. To prevent this issue we used a bridge lens as our method of focus: focal lines are in focus, as are lines ε lines above and below. After that point, there is a linear decrease in size for δ additional lines until finally the text is completely out of focus. These parameters are controlled by the user to account for line length and total length of texts. For e.g. a poem important phrases are likely to begin and end on one line and so smaller ε and δ values can be selected.

TextViewer's focus+context interface is a unique way of scanning and visualizing saliency in potentially large text documents, and the ability to load in arbitrary saliency metrics works in concert with this interface to allow fast textual analysis that combines the benefits of distanced readings with close passage analysis.

4.4. Implementation Details

From our requirements analysis we decided to develop our tools as portable Java clients that manipulated the human-readable labeled comma-delimited files already in use by our collaborators. The Docuscope tagging software generated such .csv at the corpus level, and the JMP statistical software output .csv files as well. The Processing library was chosen for its ease of implementation and inherent design as a cross-platform graphical prototyping tool.

5. Results

Our collaborators adopted these tools immediately, sharing them with fellow researchers in and with students (graduate and undergraduate) who are now being trained to use them. Strikingly, they found uses for the tools we had not anticipated, allowing us to see a number of use cases that differ from the "typical" ones described above. These adaptations are worth discussing in detail.

The tool was immediately put to work on an expanded corpus of works. Witmore and Hope had already been working with a collection of 36 Shakespeare plays whose texts have been edited and modernized through centuries of scholarship. (Lacking authoritative autograph manuscripts for Shakespeare's, literary critics must reconsruct them from multiple or variant sources.) Soon, however, they began to work with larger collections: 318 plays written between 1509-1669, including the original 36 by Shakespeare. Analysis of these data did not scale well with existing tools, offering a good use case for the tools and concepts we developed.

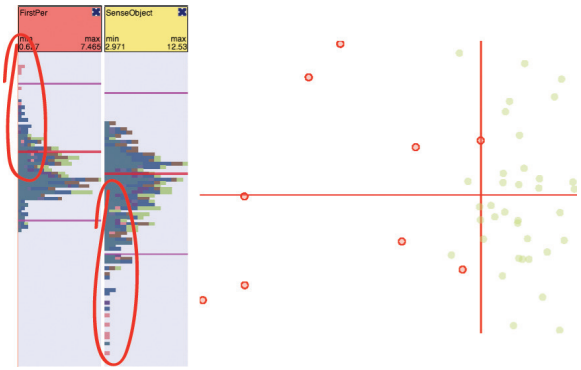

Figure 4: *Court Masques (highlighted in red) vs. Shakespeare, in views from CorpusSeparator. The masques are "pushed" leftward as a result of the current choice of principal components. We can see that certain tags (such as the FirstPer tag) do a good job of distinguishing the two groups, with all of one type clustered at one end or another of the distribution. TextViewer allows users to zoom in and see "masque-like" vs. "Shakespeare-like" passages.*

Witmore and Hope began by looking for distinct types of writing – genres – based on existing scholarly classifications of these texts (corpus-specific metadata). Using the CorpusSeparator's PCA capabilities, they isolated a subgenre known as the court masque, a forerunner of modern opera which includes spoken poetry, music and dance. CorpusSeparator allowed them to identify the most exemplary text from this genre, a masque by Ben Jonson. Accordion view then showed which tags were associated with these plays. In the case of the masque, it turned out to be words or strings of words referring to things in the world that are present in large numbers, whereas these plays are conspicuously lacking in questions. At first it was not obvious why these features would characterize the masque as a genre. Looking at an exemplary passage as indicated in the TextViewer, however, they were able to make sense of the pattern and connect it to observations that have already been made about the masque, but not in linguistic terms: because masques are a courtly genre that requires long, set speeches and little dynamic interaction among characters (they are not particularly dramatic), there are few questions and the dialogue that does exist tends to be descriptive – evoking real or imaginary scenes by describing them slowly and in detail (a practice known as *ekphrasis* in literary criticism). They were also able to take this set of weightings from PCA and apply it to Shakespeare, in effect finding the passages in Shakespeare's works that most resemble a court masque. Since there has been much speculation in the secondary literature on this issue, their ability to make this identification is significant.

Simple PCA indicated that the plays written by Shakespeare were noticeably different from the rest penned by dozens of authors. But this apparent distinctiveness turned out to be a problem: because the plays by Shakespeare had been hand-corrected and modernized in the nineteenth century, whereas the plays by the rest of the authors had been semi-algorithmically "modernized" by one of their collaborators, this difference may have been editorial. The only way to know this would be to find the passages in Shakespeare that were the most dramatically different – or better, representative of Shakespeare's difference – from other authored texts. Using CorpusSeparator, Witmore and Hope identified two principle components that effectively distinguished Shakespeare's plays from other works. This they were able to do by varying displayed components in the CorpusSeparator and looking for significant clusterings. They then saved these components and used them in TextViewer to find exemplary, Shakespearean passages.

In TextViewer, they discovered that curation and modernization were at least partially responsible for the distinctness in Shakespeare's texts: the exemplary passages they viewed in TextViewer revealed certain tokens were being systematically modernized in the Shakespeare corpus but were more variable in the semi-algorithmically modernized corpus. (Renaissance orthography is significantly more variable than contemporary orthography, in part because of the introduction of dictionaries in the eighteenth century.) Knowing this allowed them to correct the modernization algorithm and obtain higher quality texts for analysis. A significant area in their workflow was now better understood and timely, practical interventions became possible.

Having eliminated result-skewing tokens that were an artifact of editorial procedure, they could repeat their survey of corpus-wide variation at different levels of abstraction - finding, for example, patterns that are associated with time of composition (which show up as corpus size increases), but also patterns associated with authorship. With respect to Shakespeare, for example, they were able to see how this writer's "authorial signature" is connected to the ways in which his characters use words that describe properties of people (professions, social status, age, and the like). At certain points in the analysis, they became aware of the need to limit outliers – a difficult issue in literary studies, since there are few "natural" sources of constraint in the production of words that might ensure a Gaussian distribution (as there might be in a biological system). They were able to investigate this phenomenon – outlier status and its significance to literary analysis – because they had a tool that could demonstrate immediately the consequences of excluding outliers. Such exclusions could, for example, bring a particular writers' work into clearer focus. The iterative design and use process, then, produced improvements in the tools, improvements in the data, and a new question for research: what is the nature of variation in a non-physically bounded system like a literary text?

Witmore and Hope offered a presentation at a literature

forum at the St. Louis University in which they presented a large dendrogram representing Shakespeare's plays in the context of the other early modern dramatic texts in their corpus. The dendrogram itself is ungainly visually: when printed it is over 5 feet long. However, individual clusters show interesting patterns: they are produced by a single author, for example, or they are texts that are all written by members of a particular court circle. But in order to understand why the cluster occurs, critics need to be able to see an exemplary passage from the group and so apply their domain knowledge to the results. The TextViewer was used to isolate one of the clusters composed of plays written primarily by Shakespeare. When this image was put before the group of over thirty experts in Renaissance drama, they immediately began producing hypotheses about its Shakespearean character. Thus literary analysis, which is often done by individuals in their solitary reading, can now be done collectively, multiplying the power of their domain knowledge.

6. Conclusion

By careful consideration of design decisions as well as close collaboration with domain experts, we created a suite of tools well suited for use by literary scholars. These tools allow for previously impossible connections between statistical phenomena in large text corpora, and low-level patterns in text. This capability is not desired exclusively by literary scholars. The tools we created are flexible enough to adapt to arbitrary text-tagging schemata; sociologists could examine XML coded ethnographic documents to observe and qualitatively analyze social patterns, web designers could compare the HTML of a large number of websites to determine what "works" and what does not in website design. Even the relatively novice user could compare sections of his or her own written work via tags to pick out error-prone sections of code in software, or overly dense sections of prose in documents.

Outside of the domain of texts, future users could use CorpusSeparator to perform quick analysis of multidimensional scaling results from arbitrary high dimensional vectors. The accordion viewer of distribution provides quick overview of a high dimensional dataset in individual dimensions for outlier selection and exclusion. The visual principles in the TextViewer are also naturally extensible outside of the domain of tagged text. Numerous domains (inlcuding SoftVis, Natural Language Processing, and more generalized rhetorical analysis) are capable of creating complicated saliency metrics. The ability to zoom quickly to sections of text using focus+context techniques is of immediate use whenever there is a one to one mapping from text to saliency.

This work represents a first attempt to provide tools that support literary scholarship. At a basic level, it is limited to a specific type of scholarship (tag-based algorithmic criticism), and is specific to a particular set of tools (PCA). However, the basic ideas emerging from the work can generalize to a broader category of tools (for example, other statistical analysis like clustering or interpolative decompositions), and even to other domains of scholarship. For the specific tasks we considered, the current implementations have a number of limitations. For example, displaying two dimensional projections in PCA space quickly becomes cluttered as corpus size increases. Performing dynamic subsampling of plays into windows is fast at the level of individual text but slow at the corpus level. Our tools are currently optimized for performing PCA on a corpus of hundreds of texts, with up to one hundred tags. Future work will need to scale to larger sizes of corpora, larger sets of domains, and a larger set of statistical analyses. Still, the tools as they currently stand are easy to use and adequate for many immediate lines of inquiry in the current domain.

Natural extensions to the CorpusSeparator include the ability to use a wider range of dimensionality reduction schemes (including more generalized MDS), and using machine learning techniques (such as SVMs) to simplify the process of separating different groups in the corpus in a way that is readable by the TextViewer. The TextViewer tool, for its part, would need to have its colored underline visual encodings modified to deal with tagging schemes where the mapping from tag to text is not one to one.

In addition to the possibility for future exploration with our flexible and easily-deployable tools, learning to work closely with the needs and cultural milieu of humanities researchers has laid the groundwork for future collaboration and results. This collaboration is useful for humanities scholars, some of whom are skeptical of the place of computational techniques in their discipline, as well as visualization scientists, who can overlook the increasing need for compelling visualization tools in humanistic domains.

7. Acknowledgments

We thank the University of Wisconsin-Madison Working Group for Digital Inquiry for their feedback and support. This work was supported in part by NSF award IIS-0946598.

References

[BE96] BALL T., EICK S.: Software visualization in the large. *Computer 29*, 4 (Apr. 1996), 33–43. 3

[BT07] BORLAND D., TAYLOR M. R.: Rainbow color map (still) considered harmful. *IEEE computer graphics and applications 27*, 2 (2007), 14–7. 4

[CCP09] COLLINS C., CARPENDALE S., PENN G.: Docuburst: visualizing document content using language structure. In *Computer Graphics Forum* (2009), vol. 28, John Wiley & Sons, pp. 1039–1046. 2

[CK01] COLLINS J., KAUFER D.: Description of DocuScope. 3

[CPV09] CLEMENT T., PLAISANT C., VUILLEMOT R.: The Story of One: Humanity scholarship with visualization and text analysis. *Relation 10*, 1.43 (2009), 8485. 3

[CSL*10] CAO N., SUN J., LIN Y.-R., GOTZ D., LIU S., QU H.: FacetAtlas: multifaceted visualization for rich text corpora.

IEEE transactions on visualization and computer graphics 16, 6 (2010), 1172–81. 2

[CVW09] COLLINS C., VIEGAS F. B., WATTENBERG M.: Parallel Tag Clouds to explore and analyze faceted text corpora. *2009 IEEE Symposium on Visual Analytics Science and Technology* (Oct. 2009), 91–98. 2

[Ell05] ELLIS D.: The English literature researcher in the age of the Internet. *Journal of Information Science 31*, 1 (Feb. 2005), 29–36. 1

[HHN00] HAVRE S., HETZLER B., NOWELL L.: ThemeRiver: visualizing theme changes over time. *IEEE Symposium on Information Visualization 2000. INFOVIS 2000. Proceedings* (2000), 115–123. 2

[HW10] HOPE J., WITMORE M.: The Hundredth Psalm to the Tune of" Green Sleeves": Digital Approaches to Shakespeare's Language of Genre. *Shakespeare Quarterly 61*, 3 (2010), 357–390. 3

[MGT*03] MUNZNER T., GUIMBRETIÈRE F., TASIRAN S., ZHANG L., ZHOU Y.: TreeJuxtaposer: scalable tree comparison using Focus+ Context with guaranteed visibility. *ACM Transactions on Graphics (TOG) 22*, 3 (2003), 462. 7

[Ram03] RAMSAY S.: Special Section: Reconceiving Text Analysis: Toward an Algorithmic Criticism. *Literary and Linguistic Computing 18*, 2 (June 2003), 167–174. 2

[RES98] ROHRER R., EBERT D., SIBERT J.: The shape of Shakespeare: visualizing text using implicit surfaces. *Proceedings IEEE Symposium on Information Visualization (Cat. No.98TB100258)* (1998), 121–129,. 2

[SGS08] STASKO J., GÖRG C., SPENCE R.: Jigsaw: supporting investigative analysis through interactive visualization. *Information Visualization 7*, 2 (Jan. 2008), 118–132. 2

[SSTR93] SARKAR M., SNIBBE S. S., TVERSKY O. J., REISS S. P.: *Stretching the rubber sheet*. ACM Press, New York, New York, USA, June 1993. 6

[Wit10] WITMORE M.: The Funniest Thing Shakespeare Wrote? 767 Pieces of the Plays, 2010. 3

[WTP*95] WISE J. A., THOMAS J. J., PENNOCK K., LANTRIP D., POTTIER M., SCHUR A., CROW V.: Visualizing the Non-Visual: Spatial ananlysis and interaction with information from text documents. *IEEE Symposium on Information Visualzation 51* (1995), 51–58. 2