Building Customized Text Mining Tools via Shiny Framework: The Future of Data Visualization

Olga Scrivner, Vinita Chakilam, Jivitesh Poojary, Nilima Sahoo, Chandan Uppuluri, Stephan De Spiegeleire

ABSTRACT

With the increasing volume of data, there is a growing need for dynamic data visualization to help reveal instant changes in data patterns. There exist many commercial visualization tools, but traditional scholars are often disengaged from the tool development process; thus, the choice of functionalities is contingent upon tool developers whose choice may not fit the end-users. This collaboration, however, has a potential in bridging the gap between traditional scholars, who are more interested in sense-making of the text than in the tools, and the data scientists, who are more interested in the tools than in the substance, but must still contextualize the outcomes. Until recently, this collaborative process was hindered by the complexity of customization procedures and technological hurdles imposed on users with new installations. With the advent of reactive web frameworks, such as Shiny, the user-driven customization becomes not only feasible, but also essential to advance scientific research. In this paper, we demonstrate a collaborative effort between *learned* scholars and tool developers, allowing for a computational and humanistic fusion.

Keywords

visualization, text mining, Shiny web application

1. INTRODUCTION

In the last decade, the volumes of data collections have grown so “large and complex that it becomes difficult to process using on-hand databases, management tools or traditional data processing applications” [14]. As Jockers [4] points out, these massive digital collections “invite, even demand, a new type of evidence gathering and meaning making”. Consequentially, visual analytics is becoming the cornerstone of scientific analysis by combining “visualization, human factors and data analysis” and contributing to an information synthesis interpretable to the human eye [5]. Furthermore, the recent proliferation of visualization tools, both commercial and open source, has led to an increasing usage of visual analytics among traditional humanities scholars. Since most of these tools have been developed by software engineers, traditional scholars are often disengaged from the tool development process. This collaboration, however, has a potential in bridging the gap between traditional scholars, who are more interested in sense-making of the text than in the tools, and the data scientists, who are more interested in the tools than in the substance, but must still contextualize the outcomes. The insights gained from *learned* scholars would lead to a mutual enrichment, allowing for “synthesis of computational and humanistic modes of inquiry” [6]. A process of collaboration can be achieved through the following steps:

1) Scholars learn from data scientists about analytical tools, techniques, and what they can and cannot achieve

2) They exchange research questions and the implicit or explicit heuristics used in their work

3) They collaborate on how these discoveries can be made and assess the ‘quality’ of developing tools with real data

Until recently, this collaboration was unfeasible. Software not only necessitates a team of software engineers and designers, but also requires installation and consistent updates, which is a technical hurdle for users. Furthermore, the design of collaborative visualization has been commonly described as *a grand challenge* for visualization research [12]. While most visualization research has explored the cognitive and perceptual aspects of design, social interaction has only recently been recognized as a part of visualization system design [13, 3]. For example, some studies examined synchronous and asynchronous collaboration between team players to improve analytical interpretation [2, 3]. In contrast, a collaboration to enhance analytical functionalities and tool design is not common and mostly related to commercial customizable software [15].

With the advent of reactive web applications, such as Shiny, the user-driven tool customization becomes a reality. First, these applications require no installation and are accessible from any web browser, which enables a direct testing of new functionalities by users. Second, the reactive framework allows for a creation of highly dynamic tools with minimum knowledge of web development. Finally, Shiny is a web application developed for R, which is an open source language with a large library for data visualization.

In this paper, we describe our current collaborative research on text mining and visualization customization. Our goal is to assist scholars in their process of ingestion (‘reading’), digestion (analyzing and sense-making), and egestion (through the creation of new learned texts via queries). Our workflow is illustrated in Figure 1.

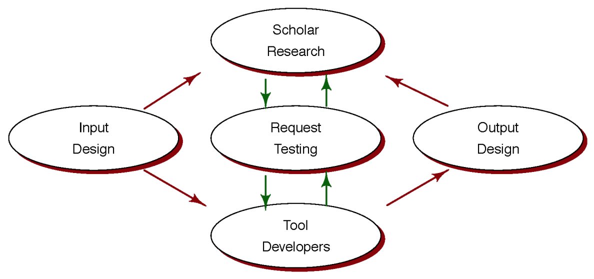


Figure 1: Information Visualization workflow: From the initial stage to a custom-designed stage

Our initial stage begins with the current version of *Interactive Text Mining Suite*,[1] a Shiny web application, developed to test various text mining and visualization techniques for digital humanities scholars [10, 11]. Our second stage comprises a direct collaboration with various scholars via *Rizzoma*, a collaborative social platform for discussions, and by means of various cloud storage platforms. The goal of this social interaction is to 1) identify scholarly research needs, 2) discuss design and functionalities, and, finally 3) develop and embed new functionalities into a web application. This stage also includes bug reports, constant feedback, and suggestions on design improvement directly from scholars. The final stage involves a fully-customized version of web application.

This paper is organized as follows. In section 2 we introduce Shiny, a reactive web framework. We then describe *Interactive Text Mining Suite* and its current functionalities in section 3. Section 4 and 5 will overview the development of customized functionalities for scholarly research, followed by conclusions and future directions presented in section 6.

2. SHINY APPLICATION

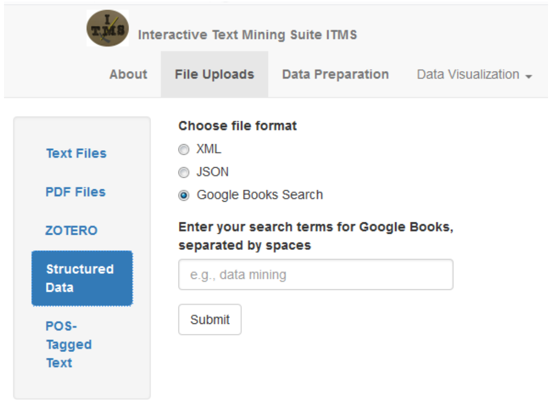
2.1 Shiny Web Framework

Traditional imperative web framework model was developed by Trygve Reenskaug in 1979 and followed a three-component structure: model, view, and control [8]. In this model, the controller plays an *essential* and *explicit* role: "you have to specify what to do when you receive user requests and what resources you are going to mobilize to carry out the necessary tasks outlined in the model" [9]. In contrast, the recent shift toward a reactive web framework has erased such a strict control, thus enabling dynamic systems that are highly responsive to users' input and interaction. Shiny, an R package, is one such application. After its release as an open source software package in 2012, the use of this application has been expanding at an unprecedented rate. This trend can be attributed to the combination of several factors: 1) Shiny web applications do not require a knowledge of web development, 2) web applications are user-friendly and dynamic, allowing for instant feedback to users, 3) web applications are accessible via browser from any device, including mobile devices, which makes it convenient to users, and 4) web applications are highly customizable, allowing for instant modification, as compared to traditional software version releases. In the next section, we will briefly describe our recently developed Shiny application, namely *Interactive Text Mining Suite.*

[1]http://www.interactivetextminingsuite.com

2.2 Interactive Text Mining Suite

Interactive Text Mining Suite (henceforth, ITMS) is designed to assist humanities scholars in the discovery of new insights and patterns within large digital collections, and to provide access to natural language processing techniques with a user-friendly design. Its major strength is the ability to work with data in various formats, PDF and text formats, as well as CSV, JSON, and XML, as shown in Figure 2.



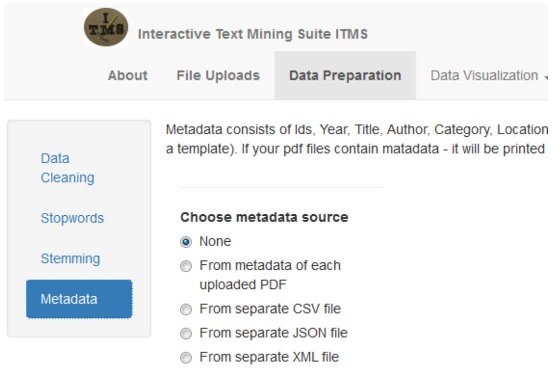
Choose file format
XML
JSON
[✓ Google Books Search]

Enter your search terms for Google Books, separated by spaces

e.g., data mining

Figure 2: Interactive Text Mining Suite: Importing data

In contrast, many existing text mining tools are limited to specific importing formats. Additionally, ITMS performs a wide range of common preprocessing tasks, allowing for maximum flexibility and user control, illustrated in Figure 3 (for a more detailed description, see [10, 11]).



Metadata consists of Ids, Year, Title, Author, Category, Location
a template). If your pdf files contain matadata - it will be printed

Choose metadata source
[✓ None]
From metadata of each uploaded PDF
From separate CSV file
From separate JSON file
From separate XML file

Figure 3: Interactive Text Mining Suite: Preprocessing data

3. USER-DRIVEN CUSTOMIZATION

As mentioned in 2.2, ITMS was designed as a digital humanities tool suitable for performing common text mining tasks and visualization methods. That is, it was built for scholars, but not by humanities scholars. However, there exists a gap between scholars, who have been doing more *qualitative* text-based research for public and government sectors, and data scientist/computational linguistics scholars, who work on theoretical text-mining research.[2] To bridge this gap, we have developed a collaborative communication between these two communities (a.k.a. end-users and developers). Instead of a typical *github* environment for reporting progress and issues, we chose a social collaborative platform *rizzoma*.[3] *Rizzoma* is built as knowledge-management and discussion platform allowing for real-time team communication and multimedia support. Figure 4 illustrates our collaborative project structure.

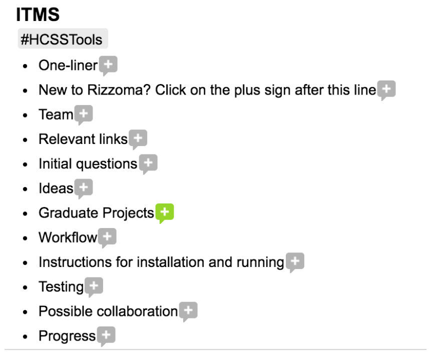

ITMS
#HCSSTools
- One-liner
- New to Rizzoma? Click on the plus sign after this line
- Team
- Relevant links
- Initial questions
- Ideas
- Graduate Projects
- Workflow
- Instructions for installation and running
- Testing
- Possible collaboration
- Progress

Figure 4: A collaborative platform Rizzoma: ITMS project

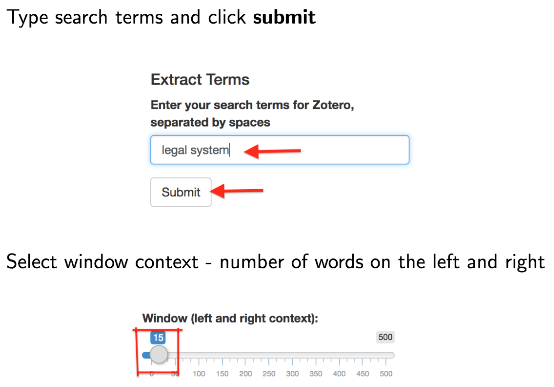
In the following sections, we describe our workflow and design considerations based on this collaboration.

4. DATA INPUT CONSTRAINT

Based on the previous work with existing text mining tools, it was determined that the main *pollutant* for scholarly research is the inability to pre-define text excerpts within the text collection. It appears that stopwords filtering and text preprocessing were not sufficient to obtain intuitive data interpretation for qualitative scholarly studies. Collaboratively, we have developed and tested the following algorithm (see also Figure 5):

1. Parse document collection
2. Scan every document for a specific term defined by the user (e.g., "security") or two terms (e.g., "influenc*" within 10 words of "Europ*")
3. Define a window around these terms (e.g., 10 words to the left and 10 words to the right)
4. Include only the extracted segments into data analysis and visualization

[2]From a personal communication with *The Hague Center for Strategic Studies (www.hcss.nl)*
[3]https://rizzoma.com


Type search terms and click **submit**

Extract Terms
Enter your search terms for Zotero, separated by spaces
legal system
Submit

Select window context - number of words on the left and right

Window (left and right context): 15

Figure 5: Development of data segmentation: window constraint

In addition, scholarly research collections are often stored and accessed via bibliographic management systems (e.g., Zotero, Mendeley, and Endnote). While most of these systems do not perform text mining analysis, the Zotero plugin, namely *Paper Machine*,[4] offers a wide range of interactive visualization for document collections. Nevertheless, the user cannot control text segmentation, which yields very broad topic and metadata visualizations. Given that Zotero is the main bibliographic management system in our collaborative project, data import from Zotero into ITMS became one of the most essential primary tasks for our team. Several options exist for exporting library collections from Zotero, namely rdf and csv formats. However, a few issues were discovered during the exploratory phase: 1) csv and rdf files only contain local paths to actual document articles (see Figure 6); 2) local paths cannot be accessed directly from a remote web application; 3) running ITMS locally would require R installation and some programming knowledge, thus generating technological hurdles for end-users.

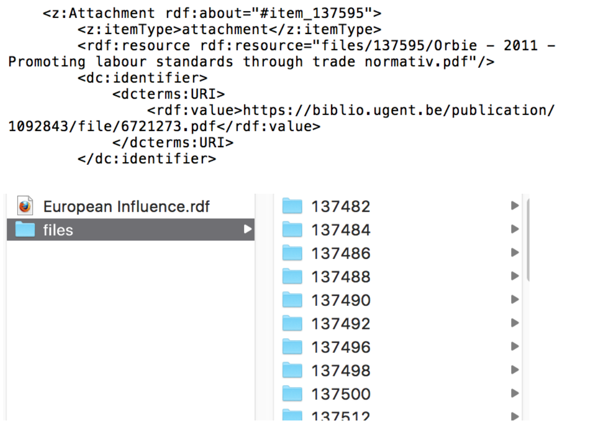
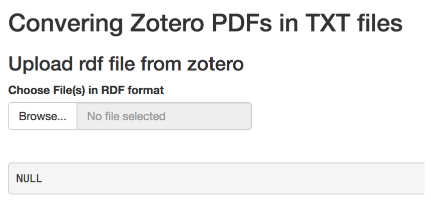

```
    <z:Attachment rdf:about="#item_137595">
        <z:itemType>attachment</z:itemType>
        <rdf:resource rdf:resource="files/137595/Orbie - 2011 -
Promoting labour standards through trade normativ.pdf"/>
        <dc:identifier>
            <dcterms:URI>
                <rdf:value>https://biblio.ugent.be/publication/
1092843/file/6721273.pdf</rdf:value>
            </dcterms:URI>
        </dc:identifier>
```
European Influence.rdf
files
137482
137484
137486
137488
137490
137492
137496
137498
137500
137512

Figure 6: Zotero collection: rdf file format (top) and Zotero internal directory structure (bottom)

Two solutions are being currently tested based on the fol-

[4]http://papermachines.org/

lowing criteria: 1) functionality and 2) the level of complexity. The first approach is the development of a small Shiny application installed locally that would process rdf library collections, export pdf files, convert them into text files, and place them into one directory, which can be accessed from our web application (see Figure 7).



Convering Zotero PDFs in TXT files

Upload rdf file from zotero

Choose File(s) in RDF format

Browse... No file selected

NULL

Figure 7: Small Shiny application: local conversion of Zotero files

This application is used only once and has a low level of complexity, yet the functionality is less user-friendly, as it creates an additional directory with extracted files from Zotero. These files can then be imported into ITMS. The second approach is suggested by the end-users: export zotero library as a csv file, run a local script to extract all pdf files, and add them into the CSV file as a plain text. While the functionality is high, the level of complexity is much higher.

5. DATA VISUALIZATION

There is no doubt that visual analytics facilitates analytical reasoning [12]. For a tool developer, however, it is not always clear whether implemented visualization methods assist the user in their research. The current project proposes to address this issue by a collaborative examination of various visualization types in order to determine their usability for Shiny application and for the end-users. First, we describe n-gram analysis, followed by interactive visualization, and topic modeling visualization.

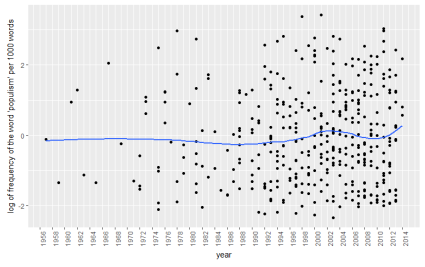


Figure 8: N-gram visualization with JSTORr package: *populism*

5.1 N-gram Visualization

An N-gram allows users to identify the co-occurrence of words within a single text or text collection. The 'n' indicates the number of words being selected to create unigrams, bi-grams, or tri-grams, etc. By using a combination of words, instead of a single occurrence, the user can generate higher quality results for data interpretation. Preferably, the user can also set a 'search window' within which co-occurrence should take place (i.e., within a certain amount of words or certain documents). In addition, our current work is concentrated on interactive and more meaningful n-gram visualization (e.g., tree visualization), as compared to traditional static graphics (Figure 8). Based on our collaborative feedback, the tree N-gram visualization was identified as more meaningful for scholarly interpretation. This tree will share prefixes of N-grams (e.g., "airport"), each connected to the root node. The root node is the set of focus words selected in the query. Every path in the tree, i.e., a path from the root node to a leaf node, corresponds to the N-gram made of the words encountered along the path, and having the score associated with the leaf node. Another possible visualization is a network representation, where the central node is the key word. There exist multiple R libraries that might be used to enhance n-gram interpretation, such as JSTORr, ngram, NSP, WordStat, among many others. In order to identify the best fit for the web application, we address the following criteria: 1) user-friendliness, 2) easy human interpretation, and 3) functionality.

5.2 Interactive Visualization

The ability to perform dynamic and interactive visualization is one of the strengths in reactive applications. While there are many R libraries implementing various types of interactive visualization, we decided to examine two packages, namely *plot.ly* and *googleViz*. Comparison and parallel testing feed our decision to implement their functionalities into ITMS. Table 5.2 presents our current summary.

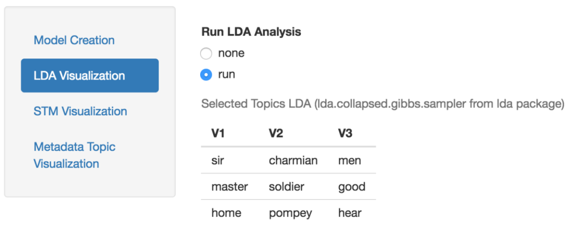
Types	GoogleViz	Plot.ly
Stepped Area chart		
Bubble chart		
Gauge		
Intensity Map		
Geo Chart		
Table with pages		
Tree Map		NA
Annotation chart		
Sankey chart		
Calendar chart		NA
Timeline chart		
Merging charts		NA
Flash charts		NA
Annotated time line chart		
Chord diagram	NA	
Filled Chord diagram	NA	
k-means clustering		
Stream Graph	NA	
PCA	NA	
Hierarchical Clustering	NA	
Doughnut Chart		

Table 1: Functional comparison of 2 R libraries: plot.ly and googleViz

After identifying their functionalities, our next step is to determine the best fit via our collaborative feedback.

5.3 Topic Modeling Story Telling

Topic modeling is a statistical model used in machine learning and natural language processing for discovering abstract topics that occur in a collection of documents. This analysis assists in "classification, novelty detection, summarization, and similarity and relevance judgements" [1]. While topic modeling results can be visualized in different forms, most common form is in a table format (see Figure 9).


Model Creation
LDA Visualization
STM Visualization
Metadata Topic Visualization
Run LDA Analysis
none
run
Selected Topics LDA (lda.collapsed.gibbs.sampler from lda package)

V1	V2	V3
sir	charmian	men
master	soldier	good
home	pompey	hear

Figure 9: Topic visualization in ITMS

By comparing other software with their unique options for topic representation, two candidates for ITMS were identified: *topic bubbling* and *topic coupling* from MALLET, a topic modeling package. The goal of topic bubbling is to compare the relative importance of all the topics; the size of a topic bubble is the accumulated size of all word bubbles within that topic. In contrast, topic coupling reveals the relations between the topics based on their associated words. In this representation, topics are shown as a network of terms (nodes) linked by their interaction with other topics.

6. CONCLUSION

In recent years, we have seen growing interest in the use of data visualization tools in the humanities fields. However, many of the existing tools are unable to integrate the humanistic component of exploratory research. Thus, the overarching goal of the current work on ITMS is to bridge the gap between tool-developers and *learned* scholars by adding a user-customization component. In addition, the social interaction between scholars and data scientists has a strong potential to promote text mining methods among humanities as well as to enhance capabilities and functionality of visualization tools. We have also shown that a recent development of reactive Shiny framework has facilitated the task of user-customization: On the one hand, a wide range of open source R libraries and its overall simplicity for deployment made the Shiny framework very accessible to non-experienced programmers. On the other hand, Shiny is user-friendly web application, where users are not constrained by limitations of their local computer memory and platform dependency, as compared to other software tools. While this project only focuses on a collaboration between political/social science scholars, this idea can be extended to other fields. Below we summarize some of the possible implementations for future research:

1. Teaching tool: The web application is developed in the conjunction with the lesson plans, for example statistics modules. The collaboration can also be expanded by including students into the development and testing phases.

2. Digital Humanities: Based on individual research, the web application can be augmented with additional visualization types, for example spatial or chronological maps for literature analysis.

3. Social Science: The user could specify additional media for research and customize their appearance, e.g. tweets, blogs, or photos.

4. Libraries: ITMS is unique in that in addition to its ability to analyze data collections, it can add bibliographic metadata. As Jockers suggests, library metadata “has been largely untapped as a means of exploring literary history” and could “reveal useful information about literary trends” [4].

All these considerations and scholarly collaboration also present new opportunities for the field of data visualization and analytics, advancing our understanding of computation and human nature, namely “synthesis of computational and humanistic modes of inquiry” [6].

7. REFERENCES

[1] N. A. Blei D. and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, pages 993–1022, 2003.

[2] K. Brodlie, D. Duce, J. Gallop, J. Walton, and J. Wood. Distributed and collaborative visualization. *Computer Graphics Forum*, 23:223–251, 2004.

[3] J. Heer and M. Agrawala. Design Considerations for Collaborative Visual Analytics. *Information Visualization*, 7:49–62, 2008.

[4] M. L. Jockers. *Topics in the Digital Humanities: Macroanalysis: Digital Methods and Literary History*. University of Illinois Press, Urbana, IL, USA, 2013.

[5] D. Keim, F. Mansmann, J. Schneidewind, and H. Ziegler. Challenges in Visual Data Analysis. In *Information Visualization (IV 2006), IEEE*, pages 9–16, 2006.

[6] L. Klein and J. Eisenstein. Reading Thomas Jefferson with TopicViz: Towards a Thematic Method for Exploring Large Cultural Archives, 2013.

[7] K. L. and E. J. Reading thomas jefferson with topicviz: Towards a thematic method for exploring large cultural archives. scholarly and research communication. 4(3), 2013.

[8] T. M. H. Reenskaug. Models-views-controllers. http://heim.ifi.uio.no/ trygver/1979/mvc-2/1979-12-MVC.pdf

[9] B. B. Ribeiro. The two frameworks. https://github.com/rstudio/shiny/issues/250, 2017.

[10] O. Scrivner and J. Davis. Topic Modeling of Scholarly Articles: Interactive Text Mining Suite. In *International Conference “Dialogue 2016”*, 2016.

[11] O. Scrivner and J. Davis. Interactive Text Mining Suite: Data Visualization for Literary Studies. In *the Workshop on Corpora in the Digital Humanities*, pages 29–38, 2017.

[12] J. Thomas and K. Cook, editors. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Press, 2005.

[13] F. Viégas and M. Wattenberg. Communication-minded visualization: A call to action. *IBM Systems Journal*, 45, 2006.

[14] T. White. *Hadoop: The Definitive Guide. Storage and Analysis at Internet Scale*. O’Reilly Media/Yahoo Press, Sebastopol, 3rd edition, 2012.

[15] J. Whitehead. Collaboration in Software Engineering: A Roadmap. In *2007 Future of Software Engineering*, pages 214–225, Washington, DC, USA, 2007. IEEE Computer Society.